Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 07, July - 2025

# Enhancing Deep Learning-Based Software Cost Estimation Through a Hybrid Meta-Heuristic Algorithmic Frame Work

Dr Ritu,
Department of Computer Science, Government PG College,
Sector 1, Panchkula, Haryana, India

Dr. Ashish Jolly,
Department of Computer Science, Government PG College,
Ambala Cantt, Haryana, India

*Abstract*— **Accurate software cost estimation plays a crucial role in informed project decision-making. The increasing adoption of nature-inspired meta-heuristic techniques has shown promise in addressing estimation challenges. While the COCOMO (Constructive Cost Model) remains a widely used regression-based approach, its reliance on fixed parameter values across similar projects limits its adaptability to varying organizational contexts. Deep learning (DL) has recently emerged as a forward-thinking strategy to enhance estimation accuracy. However, DL models often struggle with issues like extensive training time, overfitting, and complex parameter tuning. To address these challenges, meta-heuristic algorithms can assist in identifying optimal solutions with reasonable computational demands. Integrating these methods with deep neural networks can reduce training inefficiencies. This study introduces a hybrid optimization method that combines Ant Colony Optimization (ACO) with the BAT algorithm (BA), referred to as HACO-BA. This approach is evaluated on its ability to fine-tune COCOMO II parameters and enhance DL model training. Results reveal that HACO-BA outperforms standalone ACO and BA approaches in both coefficients tuning and DNN optimization. The proposed deep neural network achieved nearly 98% accuracy, surpassing the 85% accuracy of traditional neural networks on the same datasets.**

*Keywords*— **Estimation, Collaboration, Machine learning, Software tools, Unsupervised learning**

## I. INTRODUCTION

The development of projects encompasses a number of stages—from collection to maintenance—that must be completed within a defined timeline and budget to ensure product reliability. Unlike many engineering disciplines, software development faces unique challenges due to evolving customer demands and rapid technological changes, complicating effective project management. Reports, including those by the Standish Group, indicate that only a small percentage meet expected functionality. A significant number either miss these goals or are abandoned entirely. Similar patterns have been observed in global surveys across various countries, where delays, budget overruns, and high maintenance requirements are commonplace. Project planning, especially effort and cost estimation, is a foundational element of successful software project management. Misjudging these parameters can lead to project failure, resource shortages, or unjustified rejections. Contributing factors include unclear project goals, scheduling issues, budgeting errors, risk mismanagement, and external pressures. Historically, estimation methods have been classified into algorithmic (model-based) and non-algorithmic (analogy or expert-based) approaches. Non-algorithmic methods leverage past project data and expert judgment, while algorithmic models apply mathematical formulas using parameters such as source lines of code or function points. Despite their advantages, fuzzy logic, expert-based methods, and traditional machine learning face challenges such as scalability, training complexity, and data bias. Deep learning, with its capacity to model high-dimensional relationships, offers a compelling alternative—but only when combined with effective optimization strategies.

## II. RELATED WORK

A lower MMRE value indicates that the predicted effort is closer to the actual value, and hence, better model performance. Several studies have explored the use of meta-heuristic algorithms in software cost estimation. These include approaches like Genetic Algorithms (GA), Hybrid GA, ACO, and the Firefly Algorithm (FA), all of which have demonstrated improvements in estimation accuracy. Many of these techniques focus on tuning the parameters of the COCOMO model to enhance its predictive capabilities. For instance, Bee Colony Optimization (BCO)—inspired by the behaviour of bees—has been applied to adjust COCOMO parameters and yielded better results than traditional models like Bailey-Basil and Halstead. Similarly, the Whale–Crow Optimization (WCO) algorithm combines Whale Optimization and Crow Search to fine-tune regression coefficients, leading to reduced MMRE values across several datasets.

Other notable works include:

- Use of the Harmony Search Algorithm (HSA), which outperformed basic COCOMO using NASA data.
- Implementation of Particle Swarm Optimization (PSO) and ACO for estimating software reliability.
- Application of the BAT algorithm, which showed improvements when benchmarked against Grey Relational Analysis (GRA).
- Introduction of hybrid models like BATGSA, combining BAT and Gravitational Search Algorithm for enhanced optimization.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 07, July - 2025

Additional hybrid approaches have included:

- Combining ACO with the Chaos Optimization Algorithm (COA) to enhance training-test accuracy.
- Merging Genetic Algorithms with Artificial Bee Colony techniques for better convergence and lower error rates.
- Integrating PSO and Differential Evolution (DE) for handling incomplete data effectively.
- Using Cuckoo Search with Harmony Search to tune COCOMO-II parameters.

Machine learning techniques have also been applied, including:

- Random Forest (RF) for software effort prediction, whose accuracy was found to depend heavily on parameter tuning.

Deep learning models like Deep Belief Networks (DBN) and advanced variants (e.g., cost-sensitive DBN) have been proposed to improve classification in imbalanced datasets. Optimization algorithms such as Grey Wolf have also been integrated with deep learning to refine model weights and improve performance. Several authors have further proposed secure and lightweight optimization models for IoT and cybersecurity domains using methods like identity-based proxy sign cryption and modified reptile search algorithms, demonstrating the broader applicability of meta-heuristic and DL-based optimization strategies.

## III. ALGORITHMIC APPROACH

These algorithms often draw inspiration from biological systems, evolutionary behavior, animal swarm dynamics, and other natural phenomena.

### 3.1 Meta-Heuristic Algorithms

They are particularly effective in handling nonlinear, multidimensional, and large search spaces—common traits of real-world engineering problems. These algorithms are iterative and strategically combine various techniques to explore and exploit search spaces efficiently.

### 3.1.1 Ant Colony Optimization (ACO)

Originally introduced by Marco Dorigo in 1992, ACO mimics the foraging behavior of real ants. Ants release pheromones on paths they travel, which helps other ants follow the most efficient route to a food source. The collective behavior eventually reinforces the shortest and most successful paths.
In ACO, each ant represents a potential solution. The intensity of pheromones influences the likelihood of a path being chosen. Over time, better solutions emerge as pheromone trails are updated based on solution quality. ACO has been effectively applied in software estimation due to its distributed and adaptable nature.

The core pheromone update formula is:

τi,j(t+1)=τi,j(t)+∑k=1nδτi,jk(t)\tau_{i,j}(t + 1) = \tau_{i,j}(t) + \sum_{k=1}^{n} \delta\tau^k_{i,j}(t)τi,j(t+1)=τi,j(t)+k=1∑nδτi,jk(t)

### a) 3.1.2 BAT Algorithm

Proposed by Xin-She Yang, the BAT algorithm draws inspiration from the echolocation behavior of bats. Bats emit sound waves and listen to the echoes that bounce back from objects, enabling them to locate prey and navigate in darkness. Each bat in the algorithm represents a solution and is characterized by properties such as position, velocity, frequency, loudness, and pulse rate. These values are updated in each iteration based on formulas that guide global and local search.

Key equations include:

- Frequency:

  fk=fmin+(fmax−fmin)·βf_k = f_{\text{min}} + (f_{\text{max}} - f_{\text{min}}) \cdot \betafk=fmin+(fmax−fmin)·β

- Velocity:

  vkt=vkt−1+(xkt−1−xk∗)·fkv_k^t = v_k^{t-1} + (x_k^{t-1} - x_k^*) \cdot f_kvkt=vkt−1+(xkt−1−xk∗)·fk

- Position:

  xkt=xkt−1+vktx_k^t = x_k^{t-1} + v_k^txkt=xkt−1+vkt

Local search is conducted using:

xnew=xold+δ·L(t)x_{\text{new}} = x_{\text{old}} + \delta \cdot L(t)xnew=xold+δ·L(t)

where δ\deltaδ is a random factor and L(t)L(t)L(t) represents the average loudness.

### b) 3.1.3 Hybrid ACO and BAT Algorithm (HACO-BA)

The proposed hybrid algorithm, HACO-BA, combines the global search capability of ACO with the fine-tuning strengths of the BAT algorithm. By merging the two, the hybrid model capitalizes on ACO's path exploration strengths and BAT's local search efficiency. This makes HACO-BA particularly effective for multi-variable optimization problems such as software cost estimation.

### 3.2 Performance Evaluation and Evaluation

This section presents the experimental framework, datasets, and performance metrics used to evaluate various meta-heuristic algorithms for software cost and effort estimation.

### c) 3.2.1 Experimental Framework

To compare performance, six nature-inspired meta-heuristic algorithms are tested:
These are evaluated alongside the original COCOMO model. MATLAB is chosen as the development platform due to its matrix-based architecture and extensive toolbox support. The experiments are performed on three standard datasets sourced from the PROMISE repository.

d) 3.2.2 Datasets Used

Three datasets were selected, each widely recognized in software estimation research:

NASA Dataset
Contains 93 project records from various NASA canters, with 15 effort multipliers and 5 scale factors for each project.

COCOMO 81 Dataset
Based on the original COCOMO model, it includes standard effort multipliers and project-related metrics.

KEMERER Dataset
Contains 8 attributes per project, measured in KLOC. To align it with other datasets, additional attributes are assumed to have default (nominal) values.

Effort multipliers from these datasets are grouped into three categories:

- Positively correlated with effort
- Negatively correlated with effort
- Related to project scheduling

e) 3.2.3 Evaluation Metrics

Several widely accepted evaluation measures are used to determine estimation accuracy:

- MRE (Magnitude of Relative Error):

  MRE=∣Actual−EstimateActual∣\text{MRE}= \left|\frac{\text{Actual}-\text{Estimate}}{\text{Actual}}\right|MRE= ActualActual−Estimate

- MMRE (Mean Magnitude of Relative Error): The average of MRE across all projects:

  MMRE=1N∑i=1NMREi\text{MMRE}= \frac{1}{N}\sum_{i=1}^{N} \text{MRE}_iMMRE=N1 i=1∑NMREi

- MBRE (Mean Balanced Relative Error): A symmetric error metric that penalizes overestimation and underestimation equally:

  MBRE=1N∑i=1N∣Actual−Estimate∣min⁡(Actual,Estimate)\text{MBRE} = \frac{1}{N} \sum_{i=1}^{N} \frac{|\text{Actual} - \text{Estimate}|}{\min(\text{Actual}, \text{Estimate})}MBRE=N1i=1∑N min(Actual,Estimate)∣Actual−Estimate∣

- PRED(x): Percentage of projects where MRE ≤ x (commonly x = 0.25):

  PRED(x)=kN\text{PRED}(x)= \frac{k}{N}PRED(x)=Nk

where kkk is the number of projects with acceptable MRE.

f) 3.2.4 Results Overview

Three experiments were conducted—each on a different dataset:

Experiment 1 (NASA Dataset):
All algorithms outperform basic COCOMO. HACO-BA achieves the lowest MMRE, showing its superior optimization capability.

Experiment 2 (COCOMO 81 Dataset):
MMRE is generally higher compared to NASA data, but again, HACO-BA provides the best results among tested methods.

Experiment 3 (KEMERER Dataset):
Due to limited attributes (with some assumed as normal), MMRE reduction is not as pronounced. Nevertheless, HACO-BA still outperforms other approaches.

The findings, summarized in Table 4 of the paper, confirm that the hybrid HACO-BA algorithm consistently produces more accurate and reliable estimates across all datasets.

## IV. PROPOSED SOFTWARE EFFORT ESTIMATION MODEL USING A HYBRID META HEURISTIC AND DEEO LEARNING APPROACH

This section introduces a software cost estimation system that integrates deep learning with a hybrid optimization technique—HACO-BA—to improve accuracy and efficiency.

a) 4.1 Deep Learning Overview

Deep learning (DL), a subset of artificial neural networks (ANN), has shown substantial potential in solving complex prediction and classification problems across domains such as healthcare, finance, and transportation. Despite this, traditional neural networks suffer from limitations like local minima entrapment and overfitting. To overcome these issues, researchers often combine ANN with meta-heuristic optimization techniques. Popular algorithms like Genetic Algorithm (GA), Artificial Bee Colony (ABC), Cuckoo Search Algorithm (CSA), and Particle Swarm Optimization (PSO) have been successfully used to optimize neural networks, improving their convergence speed and prediction accuracy. DL models are built upon layered architectures where each node (neuron) processes weighted inputs and applies an activation function. These models are particularly useful for handling large datasets and extracting meaningful patterns.

b) 4.2 Limitations of Traditional Deep Learning

Although deep neural networks (DNNs) are powerful, their performance is limited by:

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 07, July - 2025

- Manual tuning of hyperparameters and architectures
- Lack of standardized training procedures
- High computational demands during training

Nature-inspired optimization algorithms can address these challenges by automatically finding optimal configurations for DL models. However, integrating these two approaches effectively remains a relatively unexplored area.

### c) 4.3 Hybrid Approach with HACO-BA

Based on earlier results showing that HACO-BA consistently yields the best estimation accuracy, this hybrid algorithm is chosen to optimize the deep learning model. Specifically, HACO-BA is used to determine the initial weights and configurations for a DNN to accelerate training and improve estimation accuracy. The deep learning model is trained using input from three datasets: NASA, COCOMO 81, and KEMERER. Fifteen key variables (effort multipliers and scale factors) are selected from these datasets. The output layer of the network is designed to predict the total development effort.

### d) 4.4 Data Preprocessing

Before training, all feature values are normalized using MinMax scaling to bring them into the [0, 1] range. This ensures consistent learning behavior across input variables with different value ranges.

### e) 4.5 Model Architecture

The DNN architecture consists of:

- **Input Layer:** Accepts 15 preprocessed features.
- **Hidden Layers:** Three layers with 50, 100, and 50 neurons respectively.
- **Output Layer:** Produces the estimated software effort.

The network is trained over 100 epochs using supervised learning principles, where each input is associated with a known output (effort value).

The model incorporates:

- Activation functions to introduce non-linearity
- A loss function to evaluate prediction error
- An optimizer (based on HACO-BA) to update weights
- Bias parameters initialized to zero

### f) 4.6 Model Training and Optimization

The training process involves evaluating the network's predictions against actual effort values and updating weights to minimize the loss. HACO-BA significantly enhances this process by:

- Providing optimized initial weights
- Accelerating convergence
- Avoiding local minima
- Reducing training time

The model's performance is assessed using the same four metrics: MRE, MMRE, MBRE, and PRED. The proposed HACO-BA optimized DNN is compared with conventional neural networks and other hybrid models from existing literature.

### g) 4.7 Results Summary

Experiments confirm that the proposed model achieves:

- **Higher accuracy**: Nearly 98% estimation accuracy vs. 85% for traditional NNs
- **Reduced training time**: Especially when fewer data instances are used
- **Better error metrics**: Lower MRE, MMRE, MBRE; higher PRED values

Although training time increases with more data (e.g., 50+ instances), the overall model still outperforms others in terms of accuracy and robustness.

## V. CONCLUSION

This research presents an innovative framework for enhancing software cost estimation by integrating deep learning with a hybrid meta-heuristic optimization strategy, specifically the HACO-BA algorithm. Traditional estimation models like COCOMO suffer from fixed coefficient values, which limit adaptability across diverse projects and organizational environments. Deep learning offers an advanced alternative but faces its own challenges, including high training time, overfitting, and the need for optimal parameter tuning. To address these issues, this study applied the HACO-BA algorithm—a combination of Ant Colony Optimization and the BAT algorithm—for two key purposes:

1. Tuning the coefficients of the COCOMO II model.
2. Optimizing the training process of a Deep Neural Network (DNN) for software effort estimation.

The hybrid HACO-BA method consistently outperformed individual ACO and BA methods across multiple datasets (NASA, COCOMO 81, and KEMERER). Experimental evaluations showed that the proposed model significantly reduced error rates such as MRE, MMRE, and MBRE, while increasing prediction accuracy (PRED). In particular, the HACO-BA-optimized DNN achieved an accuracy close to 98%, a notable improvement over the 85% obtained from conventional neural networks

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 07, July - 2025

## VI. FUTURE WORK

Although the proposed model delivered strong results, there is still room for further enhancement. Upcoming research could explore the following directions:

- **Real-time Model Training:** Develop mechanisms for dynamic or online training, where the model continuously updates as new data becomes available.
- **Incorporation of Additional Meta-Heuristics:** Combine more recent or domain-specific meta-heuristic techniques with deep learning to further improve estimation accuracy.
- **Automation of Feature Engineering:** Integrate automated feature selection techniques to reduce dependency on manual preprocessing and scaling.
- **Scalability Testing:** Test the system's performance on larger and more complex industrial datasets to assess generalization and robustness.
- **Hybrid Deep Architectures:** Investigate combining Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), or Transformers with meta-heuristic algorithms for specialized software prediction tasks.

The results affirm that blending deep learning with meta-heuristic optimization offers a powerful tool for addressing the persistent challenges in software cost estimation, setting a strong foundation for future enhancements in both research and practical applications..

## REFERENCES

[1] ul Hassan, C. A., Khan, M. S., Irfan, R., Iqbal, J., Hussain, S., Sajid Ullah, S., ... & Umar, F. (2022). Optimizing Deep Learning Model for Software Cost Estimation Using Hybrid Meta-Heuristic Algorithmic Approach. *Computational Intelligence and Neuroscience*, *2022*(1), 3145956.

[2] Khan, M. S., Jabeen, F., Ghouzali, S., Rehman, Z., Naz, S., & Abdul, W. (2021). Metaheuristic algorithms in optimizing deep neural network model for software effort estimation. *Ieee Access*, *9*, 60309-60327.

[3] Ritu. (2023, December). Preserving Information Integrity: A Novel Machine Learning Approach for Fake News Detection. In 2023 3rd International Conference on Mobile Networks and Wireless Communications (ICMNWC) (pp. 1-6). IEEE.

[4] Shreya, S., Kumar, S., Tirkey, S., Agarwala, S. K., & Gupta, L. K. (2023, November). Decentralized Social Media Application Based on Blockchain and NFT Technology. In 2023 International Conference on Integrated Intelligence and Communication Systems (ICIICS) (pp. 1-7). IEEE.

[5] Pandey, S., Chauhan, A., & Sharma, S. (2023, November). Advanced EdTech Strategies for Predicting Software Reliability: Harnessing ALM and SVM. In 2023 International Conference on Sustainable Communication Networks and Application (ICSCNA) (pp. 1395-1399). IEEE.

[6] Singh, N. T., Ritu, R., Kaur, C., & Chaudhary, A. (2023, July). Comparative Analysis of Traditional Machine Learning and Deep Learning Techniques for Facial Expression Recognition. In 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT) (pp. 1-7). IEEE.

[7] Singh, N. T., Rana, S., & Kumari, S. (2023, August). Facial Emotion Detection Using Haar Cascade and CNN Algorithm. In 2023 International Conference on Circuit Power and Computing Technologies (ICCPCT) (pp. 931-935). IEEE.

[8] Ritu, & Bhambri, P. (2023). Software effort estimation with machine learning–A systematic literature review. Agile software development: Trends, challenges and applications, 291-308.

[9] Ritu and P. Bhambri, "A CAD System for Software Effort Estimation," 2022 2nd International Conference on Technological Advancements in Computational Sciences (ICTACS), Tashkent, Uzbekistan, 2022, pp. 140-146, doi: 10.1109/ICTACS56270.2022.9988123.

[10] Garg, Y. (2022, May). Comparative analysis of machine learning techniques in effort estimation. In 2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON) (Vol. 1, pp. 401-405). IEEE.

[11] Bhambri, P., & Tripathy, B. Role of AI and IoT Based Medical Diagnostics Smart Health Care System for Post-Covid-19 World. In Smart Healthcare Systems (pp. 135-145). CRC Press.

[12] Bhambri, P., & Tripathy, B. (2024). Internet of Medical Things for Abnormality Detection in Infants using Mobile Phone App with Cry Signal Analysis. In Smart Healthcare Systems (pp. 236-248). CRC Press.

[13] Ritu. (2023, December). Safeguarding Ecosystems and Efficiency in Peer-to-Peer File Sharing Systems: An IoT-Inspired Approach to Pollution Mitigation. In International Conference on Deep Learning, Artificial Intelligence and Robotics (pp. 237-245). Cham: Springer Nature Switzerland.

[14] Ritu. (2023, December). Preserving Information Integrity: A Novel Machine Learning Approach for Fake News Detection. In 2023 3rd International Conference on Mobile Networks and Wireless Communications (ICMNWC) (pp. 1-6). IEEE.

[15] Pandey, S., Chauhan, A., & Sharma, S. (2023, November). Advanced EdTech Strategies for Predicting Software Reliability: Harnessing ALM and SVM. In 2023 International Conference on Sustainable Communication Networks and Application (ICSCNA) (pp. 1395-1399). IEEE.