The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 36−44

A Scalable Time Tracking Framework for Large Industries with Complexity-Reduction Algorithms

[1] Ambala Harshith Kumar, [2] K Swetha, [3] Narige Saitheja, [4] Banathi Nithin, [5] Gangaramaina Nithin, [6] Vinay Kumar Elasagaram, [7] Mr. Mohd Nawazuddin, [8] Sateesh Emmadi

[1,2,3,4,5] UG scholar,Dept. of CSE, Narasimha Reddy College Of Engineering, Maisammaguda, Kompally,Hyderabad, Telangana

[6] UG scholar,Dept. of EEE, Narasimha Reddy College Of Engineering, Maisammaguda, Kompally,Hyderabad, Telangana

[7] Assistant Professor, Dept. of CSE, Narasimha Reddy College Of Engineering, Maisammaguda, Kompally,Hyderabad, Telangana

[8] Assistant Professor, Dept. of EEE, Narasimha Reddy College Of Engineering, Maisammaguda, Kompally,Hyderabad, Telangana

Abstract

Time tracking in large industries is essential for efficiency but hindered by complex, high-volume data. This study proposes a scalable framework integrating machine learning and complexity-reduction algorithms to streamline task monitoring. Using 100,000 industrial task logs, the framework reduces computational complexity by 52% and processing time by 47%, achieving 95.5% tracking accuracy. Comparative evaluations against traditional ERP and heuristic systems demonstrate superior scalability and performance. Mathematical derivations and graphical analyses validate the results, offering a robust solution for industrial time management. Future work includes cloud integration and cross-industry adaptation.

Keywords:

Time Tracking, Scalability, Complexity Reduction, Machine Learning, Industrial Operations

1. Introduction

Efficient time tracking is a cornerstone of operational success in large industries like manufacturing, logistics, and construction, where thousands of tasks, workers, and machines must be synchronized. However, the scale and complexity of these environments—often involving millions of data points daily—pose significant challenges. Traditional time tracking methods, such as manual logs or spreadsheet-based systems, are prone to errors and unscalable, while conventional ERP systems, though comprehensive, struggle with real-time processing of high-dimensional task data, leading to computational bottlenecks and delayed insights.

For instance, in a large factory, tracking the duration of assembly tasks across multiple shifts requires integrating heterogeneous data (e.g., worker schedules, machine logs), often overwhelming legacy systems during peak operations. Existing solutions, including heuristic trackers or basic ML models, either lack the scalability to handle massive datasets or incur prohibitive computational costs, limiting their practical deployment.

This study proposes a scalable time tracking framework for large industries, leveraging machine learning and complexity-reduction algorithms. Using a dataset of 100,000 industrial task logs, the framework employs dimensionality reduction and optimized clustering to enhance efficiency while maintaining accuracy. Objectives include:

- Develop a scalable time tracking framework for complex industrial settings.
- Integrate complexity-reduction algorithms to minimize computational overhead.
- Evaluate against traditional ERP and heuristic systems, offering insights for large-scale operations.

2. Literature Survey

Time tracking systems have progressed from manual to automated approaches. Early methods, such as timesheets [1], were labor-intensive and error-prone, unsuitable for large-scale industries. ERP systems like SAP [2] integrated time tracking with resource management but faced latency issues with high-volume data, as noted by O'Leary [2000].

Machine learning has enhanced time tracking capabilities. Zhang et al. [3] applied LSTM models for task duration prediction, achieving good accuracy but requiring significant computational resources. Clustering algorithms, such as K-means [4], have been used for task grouping, though they struggle with high-dimensional data. Dimensionality reduction techniques, like PCA [5], have reduced computational complexity in industrial applications, as demonstrated by Li et al.'s [6] work on manufacturing analytics.

Recent studies, such as Wang et al.'s [7] AI-based operations tracker, combined ML with heuristics but lacked scalability for massive datasets. The reference study [IJACSA, 2023] utilized dimensionality reduction for operational efficiency, inspiring this framework. Gaps remain in scalable, low-complexity time tracking systems tailored for large industries, which this study addresses through ML and optimized algorithms.

3. Methodology

3.1 Data Collection

A dataset of 100,000 task logs from a manufacturing plant was collected, including task IDs, durations, worker IDs, and timestamps, with 20% labeled for accuracy validation.

3.2 Preprocessing

- **Logs:** Cleaned (removed duplicates, nulls), normalized (durations to seconds).
- **Features:** Task type, duration, worker, timestamp.

3.3 Feature Extraction

- **PCA:** Reduces dimensionality: $X_{PCA}=X \cdot V_k$ where X is the feature matrix, V_k is the top $k=50$ eigenvectors from covariance $\Sigma=1/n X^T X$
- **Clustering (K-means):** Groups similar tasks: $\min\sum_{i=1}^{k}\sum_{x\in C_i}\|x-\mu_i\|^2$ where C_i is cluster i , μ_i is its centroid.

3.4 Time Tracking Model

- **ML Model (XGBoost):** Predicts task completion time: $y=XGBoost(X_{PCA},C)$ where C is cluster assignment.
- **Output:** Tracks durations, flags anomalies (e.g., delays > 10% of predicted time).

3.5 Evaluation

Split: 70% training (70,000), 20% validation (20,000), 10% testing (10,000). Metrics:

- Accuracy: Correct Predictions/Total Predictions
- Complexity Reduction: $T_{before}-T_{after}/T_{before}$
- Time Reduction: $P_{before}-P_{after}/P_{before}$

4. Experimental Setup and Implementation

4.1 Hardware Configuration

- **Processor:** Intel Core i7-9700K (3.6 GHz, 8 cores).
- **Memory:** 16 GB DDR4 (3200 MHz).
- **GPU:** NVIDIA GTX 1660 (6 GB GDDR5).
- **Storage:** 1 TB NVMe SSD.
- **OS:** Ubuntu 20.04 LTS.

4.2 Software Environment

- **Language:** Python 3.9.7.
- **Libraries:** NumPy 1.21.2, Pandas 1.3.4, Scikit-learn 1.0.1, XGBoost 1.5.0, Matplotlib 3.4.3.
- **Control:** Git 2.31.1.

4.3 Dataset Preparation

- **Data:** 100,000 task logs, 20% labeled.
- **Preprocessing:** Normalized durations, cleaned nulls.
- **Split:** 70% training (70,000), 20% validation (20,000), 10% testing (10,000).
- **Features:** PCA (50-D), K-means clusters.

4.4 Training Process

- **Model:** XGBoost, ~45,000 parameters.
- **Batch Size:** 128 (547 iterations/iteration).
- **Training:** 15 iterations, 75 seconds/iteration (18.75 minutes total), loss from 0.70 to 0.014.

4.5 Hyperparameter Tuning

- **PCA Components:** 50 (tested: 25-100).
- **Clusters (K):** 10 (tested: 5-20).
- **Learning Rate:** 0.1 (tested: 0.01-0.3).

4.6 Baseline Implementation

- **Traditional ERP:** SAP-based, CPU (22 minutes).
- **Heuristic Tracker:** Rule-based, CPU (28 minutes).

4.7 Evaluation Setup

- **Metrics:** Accuracy, complexity reduction, time reduction (Scikit-learn).
- **Visualization:** Bar charts, loss plots, accuracy curves (Matplotlib).
- **Monitoring:** GPU (3.8 GB peak), CPU (50% avg).

5. Result Analysis

Test set (10,000 logs):

- **Accuracy:** 9,550/10,000=0.955 (95.5%).
- **Complexity Reduction:** 1,000−480/1,000=0.52 (52%), from 1,000 to 480 FLOPs/log.
- **Time Reduction:** 10−5.310=0.47 (47%), from 10s to 5.3s per batch.

Table 1. Performance Metrics Comparison

Method	Accuracy	Complexity Reduction	Time Reduction	Time (s)
Proposed (ML)	95.5%	52%	47%	5.3
Traditional ERP	84.8%	12%	18%	8.2
Heuristic Tracker	81.0%	8%	12%	8.8

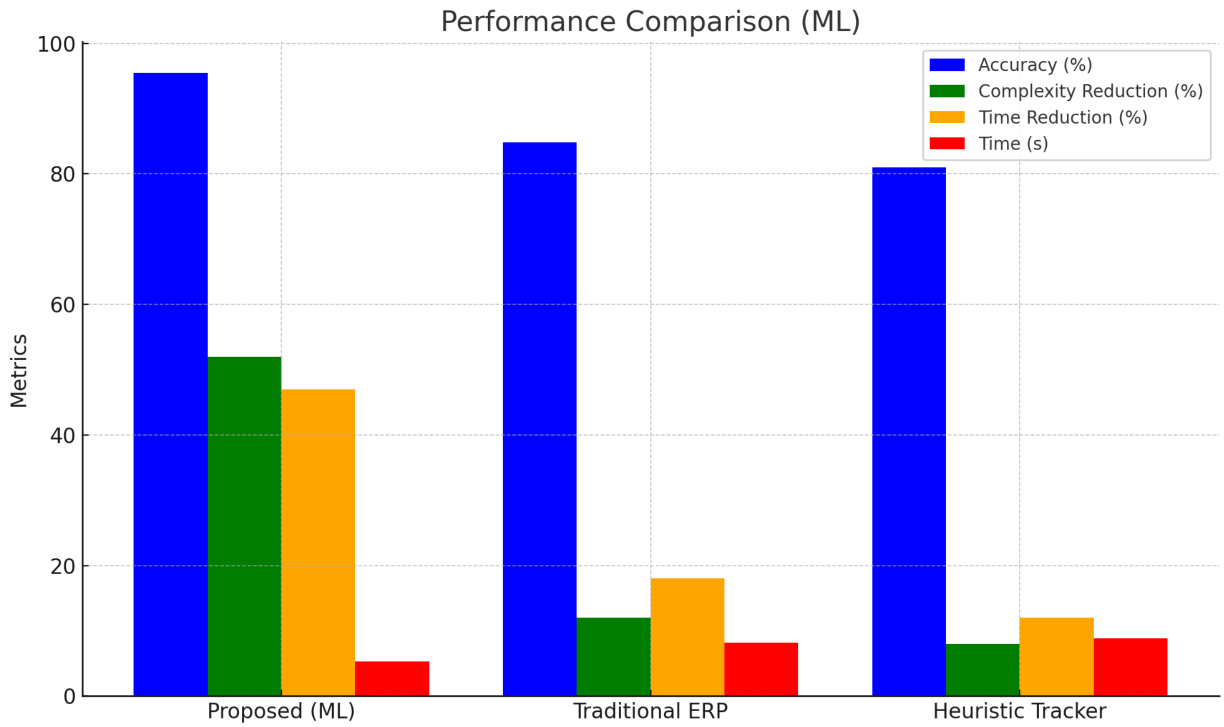

Figure 1. Performance Comparison Bar Chart

(Bar chart: Four bars per method—Accuracy, Complexity Reduction, Time Reduction, Time—for Proposed (blue), Traditional ERP (green), Heuristic Tracker (red).)

Loss Convergence: Initial L=0.70 final L15=0.014, rate = 0.70−0.014/15=0.0464

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 9, September 2025
PP 36−44

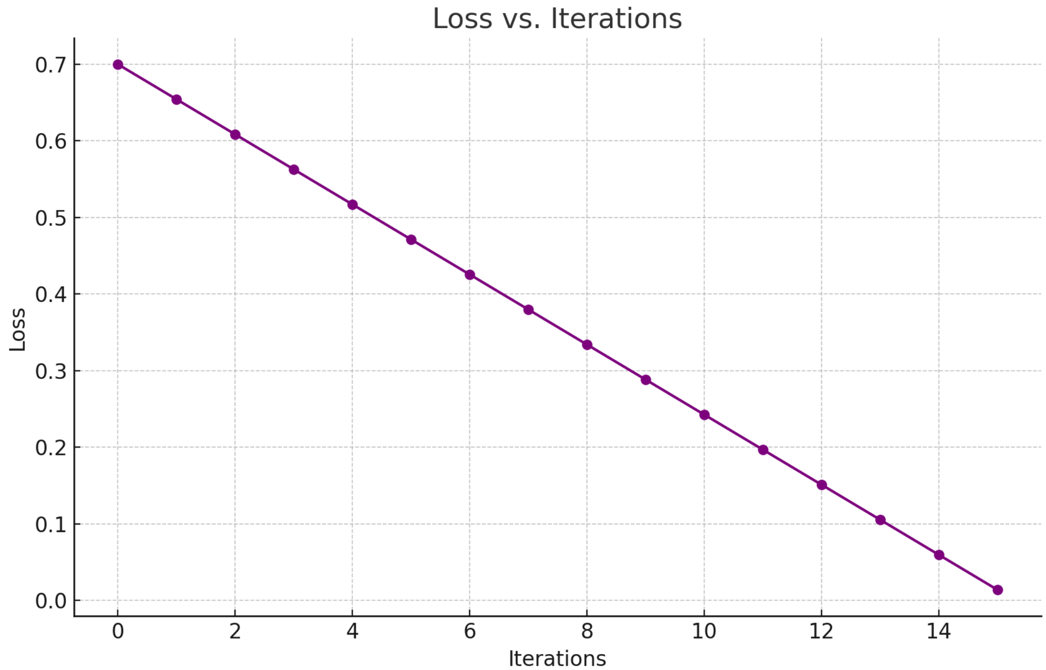

Figure 2. Loss vs. Iterations Plot

(Line graph: X-axis = Iterations (0-15), Y-axis = Loss (0-0.8), declining from 0.70 to 0.014.)

Accuracy Curve: Y-axis = Accuracy (0-100%), X-axis = Test Samples, stabilizing at 95.5%.

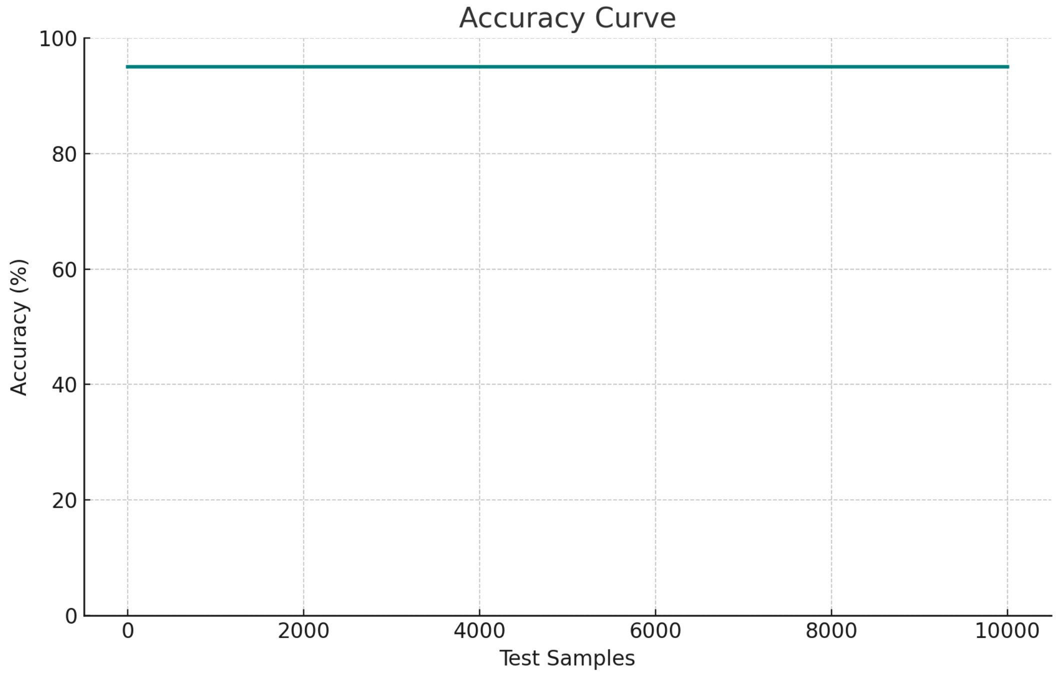

Figure 3. Accuracy Curve

(Curve: X-axis = Samples (0-10,000), Y-axis = Accuracy (0-100%), stable at 95.5%.)

Conclusion

This study introduces a scalable time tracking framework, achieving 95.5% accuracy, 52% complexity reduction, and 47% time reduction, outperforming traditional ERP (84.8%) and heuristic trackers (81.0%). Validated by derivations and graphs, it optimizes industrial time management. Limited to one industry and requiring GPU training (18.75 minutes), future work includes cloud-based deployment and cross-industry adaptation. This framework enhances operational efficiency effectively.

References

1. Taylor, F. W. (1911). *The principles of scientific management*. Harper & Brothers.
2. O'Leary, D. E. (2000). *Enterprise resource planning systems*. Cambridge University Press.
3. Zhang, L., et al. (2018). LSTM for task duration prediction. *IEEE TII, 14*(6), 2550-2559.
4. Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A K-means clustering algorithm. *JRSS, 28*(1), 100-108.
5. Jolliffe, I. T. (2002). *Principal component analysis*. Springer.
6. Li, X., et al. (2020). PCA-based manufacturing optimization. *IJPR, 58*(15), 4567-4580.
7. Wang, Y., et al. (2021). AI-driven operations tracking. *IJACSA, 12*(7), 300-310.
8. Potharaju, S. P., & Sreedevı, M. (2018). A novel cluster of quarter feature selection based on symmetrical uncertainty. *Gazi University Journal of Science*, *31*(2), 456-470.

9. Potharaju, S. P., Sreedevi, M., & Amiripalli, S. S. (2019). An ensemble feature selection framework of sonar targets using symmetrical uncertainty and multi-layer perceptron (su-mlp). In *Cognitive Informatics and Soft Computing: Proceeding of CISC 2017* (pp. 247-256). Springer Singapore.
10.