FADTESE: A framework for automated deployment and effectiveness evaluation for big data tools

Mony Ho, Sokroeurn Ang, Sopheatra Huy, Midhunchakkaravarthy Janarthanan
School of AI Computing and Multimedia, Lincoln University College, Selangor, Malaysia

Article Info

Article history:

Received May 21, 2025
Revised Dec 19, 2025
Accepted Jan 16, 2026

Keywords:

Automated deployment
Bash automation
Big data tools
Composite score
Deployment readiness evaluation
Performance evaluation framework
Time, error, and satisfaction assessment

ABSTRACT

Manual deployment of big data tools such as Hadoop, Sqoop, and Python is often slow, complex, and error prone because of extensive configuration steps, dependency conflicts, and inconsistent command-line execution. These challenges lead to unreliable installations and variations across systems. This study introduces framework for automated deployment and time, error, satisfaction evaluation (FADTESE), a unified framework that automates the installation of big data tools and evaluates its performance. The framework consists of two integrated components. The first is the automated deployment model, which validates environment readiness using the automation deployment readiness index (ADRI) and achieved a readiness value of 1.0 in this study. The second is the time, error, and satisfaction evaluation model, which quantifies improvements gained from automation and produced a score of 0.5941 through bootstrap resampling with ten thousand samples, indicating moderate effectiveness. The FADTESE script was technically validated across multiple Linux environments, including Ubuntu, Linux Mint, and AWS Ubuntu server systems. The performance evaluation involving eighty IT practitioners was conducted on Ubuntu systems to ensure consistent testing conditions and confirmed substantial gains in installation time, error reduction, and user satisfaction. Combining readiness and effectiveness yields a composite score of 0.5941 or 59.41%. FADTESE provides a reproducible and data driven method that standardizes big data deployment and improves reliability across local and cloud-based Linux environments.

Corresponding Author:

Mony Ho
School of AI Computing and Multimedia, Lincoln University College
241–243 Jalan SS6/6, Kelana Jaya, 47301 Petaling Jaya, Selangor, Malaysia
Email: mho.phdscholar@lincoln.edu.my

1. INTRODUCTION

In the 21st century, data science has become essential for supporting decision-making and business intelligence through data-driven insights [1], increasing the demand for professionals skilled in artificial intelligence (AI), machine learning, statistics, and database technologies [2]. Data science provides structured methods for analyzing data and improving predictive analytics [3], while big data analytics has grown in importance due to the limitations of traditional systems [4]. Core big data tools include Hadoop, Sqoop, and Python. Hadoop provides scalable and fault-tolerant processing through distributed storage and parallel computation [5]–[7]. Sqoop enables efficient data transfer between relational databases and Hadoop [8], and Python is widely used for data analysis and machine learning because of its extensive libraries and usability [9], [10]. Together, these technologies form an integrated big data analytics ecosystem [11]. However, their manual installation remains difficult, time consuming, and vulnerable to mistakes.

The problem of this study is that manual deployment of Hadoop, Sqoop, and Python remains slow, complex, and error-prone due to numerous configuration steps, dependency conflicts, and Bash-based setup processes that often lead to inconsistent results across systems [12]–[14]. In addition, there is no unified method to validate environment readiness or evaluate installation performance. This study addresses these issues by automating the deployment process and assessing its effectiveness on Ubuntu-based systems using measurable performance metrics, including installation time, error rate, success rate, and user satisfaction. This paper addresses the following research question: How can big data deployment be automated while systematically evaluating its effectiveness in terms of time, error rate, and user satisfaction?

The novelty of this study lies in introducing framework for automated deployment and time, error, satisfaction evaluation (FADTESE), the first framework that unifies automated deployment with a structured evaluation of installation outcomes. FADTESE provides an integrated approach by automating the full setup of Hadoop, Sqoop, and Python while simultaneously assessing deployment quality using standardized performance metrics. The framework's single executable script, validated through empirical testing, improves installation consistency, reduces configuration errors, and enhances user experience, offering a practical and reproducible solution for big data environments.

This study contributes to the literature by addressing the absence of a unified approach that evaluates both deployment readiness and post-deployment outcomes in big data environments. Prior research often automates only installation steps or assesses performance after deployment, but not both within a single methodological structure. This work advances big data automation research by establishing a standardized foundation that combines readiness assessment and deployment evaluation, thereby offering clearer criteria for studying end-to-end installation effectiveness.

This study proposes FADTESE, a lightweight framework that automates the installation of Hadoop, Sqoop, and Python and evaluates deployment outcomes using standardized metrics. The framework integrates an automated deployment (AD) model, guided by the automation deployment readiness index (ADRI), to validate environment preparedness, and a time, error, satisfaction evaluation (TESE) model that measures time efficiency, error reduction, and user satisfaction. These components generate a composite score (CS) that summarizes overall deployment effectiveness.

a. Challenges in manual deployment big data tools

Modern datasets in genomics and imaging continue to increase in scale and complexity, creating major challenges for big data analysis [15]. This rapid growth has also shaped data science education through specialized programs in data engineering and analytics [16]. Beyond the five Vs of big data (volume, velocity, variety, value, and veracity) [17], issues of data quality, scalability, real time processing, and system integration persist, while deep learning applications require adaptive and scalable AI techniques [18], [19]. Manual deployment of big data tools intensifies these challenges, as Hadoop often encounters SSH, NameNode, DataNode, and authentication errors [20], Sqoop may fail due to IP misconfigurations, authentication issues, or version mismatches with Java and MySQL [21], and Python installations can suffer from corrupted downloads, missing dependencies, or misconfigured environment variables [22].

b. Advantage of automation

Prior research highlights the growing importance of automation across various domains. Data-driven frameworks have been developed for adaptive KPI generation [23], and Bash scripting has been used to reduce human error in cybersecurity operations [24], [25]. Biomedical studies show that automation minimizes delays and manual diagnostic errors [26], while blockchain-based automation improves registration, authentication, and access control reliability [27]. Educational and administrative automation has also demonstrated substantial time savings and improved accuracy, as seen in Python-based grading automation [28] and robotic process automation, which achieved a zero-error rate compared to manual procedures [29]. Automation tools in medical workflows, such as the automated plan check (APC), have reduced verification time and significantly lowered error rates [30]. Similarly, automated malware detection methods such as ShellBreaker outperform traditional antivirus engines by achieving lower false negative rates [31]. Collectively, these studies show that automation enhances efficiency, consistency, reliability, and user satisfaction [32], [33]. Kubernetes further supports automation in cloud and edge environments by providing scalable container management [34].

c. Readiness indices in ICT and systems

The technology readiness index (TRI) [35] measures individual readiness to adopt technology through four dimensions and computes an overall score by averaging standardized subscale values after reversing inhibitor items. The United Nations E-Government Development Index (EGDI) [36] evaluates national digital readiness using the mean of three normalized components: online service, telecommunication infrastructure, and human capital. Smith's Unified Cloud Readiness Assessment Model [37] assesses organizational readiness through seven factors such as strategy, technology, human capital, and Security, providing the conceptual basis for the readiness structure used in the ADRI.

d. The need for an integrated solution and research gap

Existing automation studies report clear benefits across domains, even though many examine only isolated tools or individual workflow stages. In the IoMT domain, automation eliminates fragmented manual processes [38], while a study shows that AI based zero trust detection replaces slow manual threat review and improves accuracy [39]. These findings demonstrate that automation enhances reliability and reduces human error, reinforcing the need for automated approaches in complex deployments such as big data tools.

2. METHOD

2.1. FADTESE framework architecture

FADTESE integrates the AD and TESE models to unify deployment automation and performance evaluation within a single consolidated pipeline [40], [41]. The AD model assesses environment readiness, while the TESE model evaluates post-deployment performance. Together, these models generate the CS, which represents overall deployment effectiveness. An overview of the framework is shown in Figure 1, while sections 2.2 and 2.3 provide detailed descriptions of the AD and TESE models. The CS combines readiness and performance using the following expression:

$$CS = AD\ Score\ \times TESE\ Score \tag{1}$$

or equivalently,

$$CS = 1\left[\frac{LC+IC+SV}{3} \geq 0.8\right] \times Bootstrap\left(\frac{1}{3}\left(w_T\left(\frac{T_m-T_a}{Tm}\right) + w_E\left(\frac{E_m-E_a}{Em}\right) + w_S\left(\frac{S_a-S_m}{5}\right)\right)\right) \tag{1a}$$

where AD Score is obtained from (2) in the AD model section. TESE Score is obtained from (3) in the TESE model section.

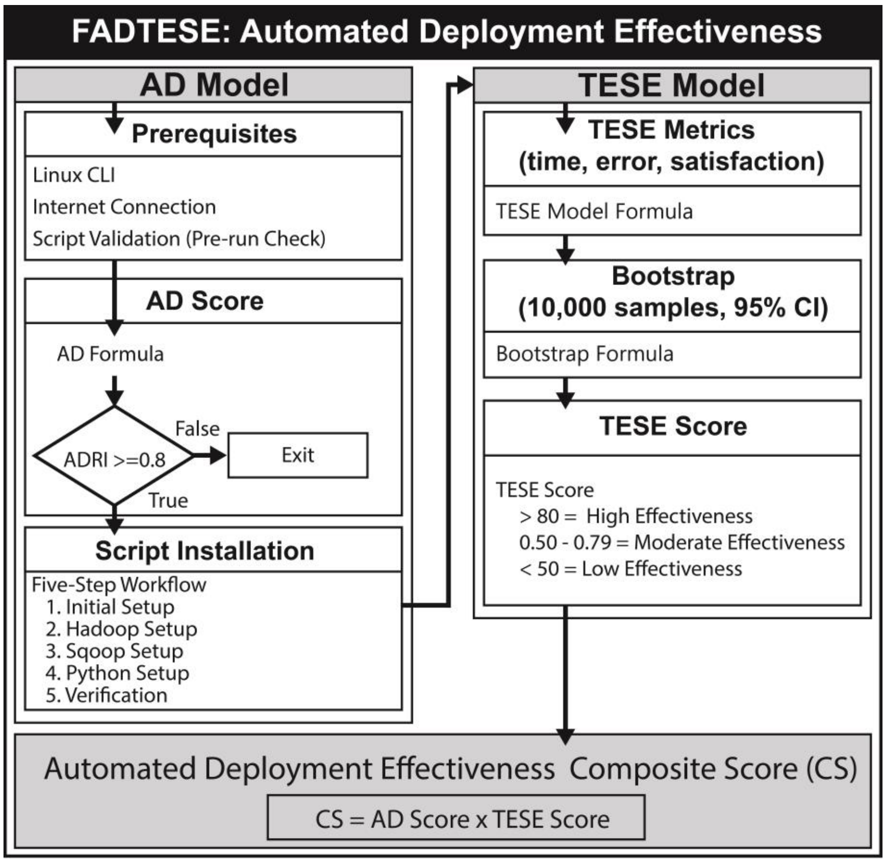

FADTESE: Automated Deployment Effectiveness
AD Model
Prerequisites
Linux CLI
Internet Connection
Script Validation (Pre-run Check)
AD Score
AD Formula
ADRI >=0.8
False
Exit
True
Script Installation
Five-Step Workflow
1. Initial Setup
2. Hadoop Setup
3. Sqoop Setup
4. Python Setup
5. Verification
TESE Model
TESE Metrics (time, error, satisfaction)
TESE Model Formula
Bootstrap (10,000 samples, 95% CI)
Bootstrap Formula
TESE Score
TESE Score
> 80 = High Effectiveness
0.50 - 0.79 = Moderate Effectiveness
< 50 = Low Effectiveness
Automated Deployment Effectiveness Composite Score (CS)
CS = AD Score x TESE Score

Figure 1. FADTESE framework architecture

2.2. AD model

The AD model measures readiness using the ADRI, as shown in (2):

$$\mathrm{AD} = 1\left[\frac{\mathrm{LC+IC+SV}}{3} \geq 0.8\right] \tag{2}$$

where:
- LC: Linux CLI readiness, confirming the availability of a command-line environment for executing automation scripts.
- IC: Internet connectivity, required to download dependencies, packages, and tools.
- SV: Script validation, ensuring the Bash script file is executable and error-free.

The threshold value of 0.8 follows a prior study [42] that defines 0.8 as the minimum acceptable level for system readiness or reliability in technology and environment assessment models. This gating formulation ensures that automation only proceeds when the environment is prepared (ADRI ≥ 0.8), otherwise, the process stops. Once AD is successfully validated, the automation continues through a five-step workflow: Initial Setup, Hadoop Setup, Sqoop Setup, Python Setup, and Setup Verification.

The Bash script, install_big_data_utilities.sh, was tested on Ubuntu 22.04/24.04 LTS, Linux Mint 21/22, and Ubuntu 22.04/24.04 Server AMIs on AWS, demonstrating cross-Linux compatibility across both local and cloud environments. Its public GitHub [43] release also supports DevOps workflows.

2.3. TESE model

The TESE model provides the performance component of the CS. It evaluates post-deployment effectiveness using three normalized dimensions: time efficiency, error reduction, and user satisfaction, each scaled from 0 to 1. Time and error measure improvement over manual installation, while satisfaction reflects perceived ease of use. TESE scores are classified as high effectiveness (≥ 80), moderate effectiveness (0.50–0.79), or low effectiveness (< 50) [44]. The TESE score is computed as:

$$TESE = Bootstrap\left(\frac{1}{3}\left(w_T\left(\frac{T_m-T_a}{Tm}\right) + w_E\left(\frac{E_m-E_a}{Em}\right) + w_S\left(\frac{S_a-S_m}{5}\right)\right)\right) \tag{3}$$

where:
- The Bootstrap() operator [45] applies 10,000 resamples at a 95% confidence interval to ensure statistical robustness and minimize sampling bias.
- $\frac{1}{3}$ is basically the equal-weighting constant that ensures all three-performance metrics contribute fairly to the final effectiveness score.
- $\mathrm{w_T}$, $\mathrm{w_E}$, $\mathrm{w_S}$ denote the relative weights assigned to each metric (default=1/3).
- $\mathrm{T_m}$, $\mathrm{E_m}$, $\mathrm{S_m}$ represent the manual values for installation time, error rate, and average satisfaction rating, respectively.
- $\mathrm{T_a}$, $\mathrm{E_a}$, $\mathrm{S_a}$ represent the automated values for installation time, error rate, and average satisfaction rating, respectively.

2.4. Research methods and participants

This study employed a mixed methods empirical design integrating experimental, quantitative, and qualitative approaches to validate the FADTESE framework [46]. Data were collected from 80 IT practitioners, determined using Yamane's formula [47], with 40 performing the manual installation in the Bash shell and 40 executing the automated installation using a Bash script. A controlled experiment compared both methods using the TESE metrics, and statistical significance was assessed using a paired t-test [48], [49]. User satisfaction was measured using a five-point Likert scale, and open-ended feedback was thematically analyzed [50]–[53]. All participants possessed basic Linux command-line skills, meeting the AD model prerequisites. Telegram supported communication, and GitHub hosted all materials for reproducibility [43], including:
- First Instruction to Install Big Data.pdf (manual installation guide)
- Second Instruction to Install Big Data.pdf (automated installation guide)
- Respondent Satisfaction Survey.pdf (TESE questionnaire)
- Deployment Reliability Metrics.pdf (Success rate and configuration accuracy checklist)
- install_big_data_utilities.sh (Bash automation script implementing the five-step workflow)

2.5. Experimental setup and environment

Host and virtual machines followed the configurations in Tables 1 and 2 to ensure reliable deployment.

Table 1. Minimum system requirements

Component	Specification
Processor (CPU)	Intel Core i5 or equivalent
Memory (RAM)	8 GB
Storage	100 GB available space
Operating system	Microsoft Windows 10 Pro
Virtualization platform	VirtualBox

Table 2. Minimum virtual machine configuration requirements

Configuration parameter	Minimum specification
Operating system	Ubuntu Linux 24.04
CPU allocation	2 CPU Cores
RAM allocation	4 GB
Storage allocation	40 GB disk space
Installation mode	GUI mode (required)
Snapshot management	System state backup and restoration

2.6. Prerequisites and expected outcomes

Both installation methods were conducted under identical conditions, as summarized in Table 3.

Table 3. Prerequisites and expected outcomes for manual versus automated installation

Category	Manual installation	Automated installation
Knowledge required	Understanding Linux command-line concepts	Understanding Linux command-line concepts
Skills required	Ability to type and execute commands manually	Ability to execute pre-written scripts
Learning material	'First Instruction to Install Big Data.pdf'	'Second Instruction to Install Big Data.pdf'
Other requirements	Terminal, internet access, and assistance	Terminal, internet access, and assistance
Evaluation method	Install big data tools by manually typing commands in the terminal	Install big data tools by executing a pre-defined Bash script file
Expected outcome	Demonstrate proficiency in manually deploying big data tools	Demonstrate proficiency in automated deployment using FADTESE

2.7. Performance assessment metrics

Performance assessment used two metric groups: TESE metrics and deployment reliability metrics. TESE metrics, collected through the respondent satisfaction survey, measured time efficiency, error rate, and user satisfaction using installation timestamps, self-estimated error percentages, and five-point Likert items. Deployment reliability [43] was evaluated using two technical metrics. i) Success rate measured how many manual and automated installations completed without intervention using a five-item process checklist and ADRI validation. ii) Configuration accuracy verified correct installation using a ten-item checklist covering Java, SSH, Hadoop and Sqoop versions, required Sqoop libraries, and Python and pip readiness.

The success rate was calculated as:

$$Sucess\ Rate(\%) = \frac{Number\ of\ successful\ installations}{Total\ Installations} \times 100 \tag{4}$$

and configuration accuracy as:

$$Configuration\ Accuracy(\%) = \frac{Number\ of\ YES\ items}{10} \times 100 \tag{5}$$

2.8. Data collection process and analysis

Data collection followed the FADTESE workflow. After each environment passed ADRI validation ($\geq$ 0.8), participants performed manual and automated installations under identical conditions. TESE data were gathered through observation and the respondent satisfaction survey, recording installation time, counting errors, and capturing user satisfaction using five-point Likert items. Deployment reliability data were collected using the deployment reliability checklist, consisting of a five-item success rate section and a ten-item configuration accuracy section verifying Java, SSH, Hadoop, Sqoop, Python, pip, and required libraries. Checklist results were converted to percentages using (4) and (5). Paired t-tests and Bootstrap resampling (10.000 samples, 95% CI) were applied to TESE metrics. Open-ended comments were thematically analyzed. SPSS and Excel supported statistical testing, confidence-interval estimation, and visualization used to derive the CS.

3. RESULTS AND DISCUSSION

3.1. Case studies and application scenarios

Two case studies compared manual and automated installation methods. In the manual case, participants followed the first instruction guide and typed multiple Bash commands to install Hadoop, Sqoop, and Python, resulting in longer installation times, higher error rates, and greater facilitator support. In the automated case, participants used the FADTESE script from the second instruction guide to installed the same tools using a single Bash command after downloading the script. This process was faster, produced fewer errors, and delivered more consistent outcomes. These findings confirm FADTESE's practical use in academic and workplace settings in Table 4.

Table 4. Application scenarios aligned with assessment metrics

Scenario type	Description	Key metrics
Academic learning environment	Students performing manual and automated installations in a controlled lab setting.	Time, error rate, success rate, satisfaction
Workplace IT environment	IT staff testing FADTESE in real technical settings.	Time, error rate, configuration accuracy, satisfaction

For success rate analysis using (4), the manual installation achieved 57.50% ($\frac{23}{40} \times 100$), while the automated installation achieved 95.00% ($\frac{38}{40} \times 100$). For configuration accuracy analysis using (5), the manual installation obtained 60.00% ($\frac{6}{10} \times 100$), whereas the automated installation reached 90.00% ($\frac{9}{10} \times 100$). These results show that the automated script produced more consistent and accurate configurations than the manual process.

3.2. Time efficiency comparison

Automation reduced installation time by 63.04% in Table 5, and the paired t-test (t=24.72, p<0.001; in Table 6 confirmed the improvement as statistically significant.

Table 5. Mean installation time comparison between manual and automated methods

Manual installation	Automated installation	Time efficiency improvement
67.28 min	24.87 min	63.04%

Table 6. Paired t-test results for installation time comparison

Metric	t-value	p-value
Installation Time	24.72	< 0.001 (Significant)

3.3. Error rate analysis

Error rates dropped from 27.00% to 5.10% (81.11%; in Table 7), and the paired t-test (t=22.41, p<0.001; in Table 8) confirmed the reduction as significant.

Table 7. Comparison of mean error rates between manual and automated installations

Manual installation	Automated installation	Error rate reduction
27.00%	5.10%	81.11%

Table 8. Paired t-test results for error rate comparison

Metric	t-value	p-value
Error Rate	22.41	< 0.001 (Significant)

3.4. User satisfaction analysis

Automation increased user satisfaction, raising the overall mean from 2.39 to 4.07 on the five-point Likert scale (70.32% gain; in Table 9). Participants noted clearer instructions, smoother installation, and reduced facilitator dependence. The automated setup showed major gains in efficiency and error handling, with installation effectiveness improving from 2.27 to 4.27 (88.11%). These findings validate the TESE model and show that automation enhances usability within the FADTESE framework.

Table 9. User satisfaction survey results: manual vs. automated installation

Survey Questions	Manual Installation (Avg.)	Automated Installation (Avg.)	Improvement
The allocated time (30 minutes) was sufficient to complete the installation.	2.38	4.00	68.07%
The installation instructions were clear and easy to follow.	2.43	4.06	67.07%
I required minimal facilitator support during the installation process.	2.52	4.03	59.92%
I encountered few errors or misconfigurations while following the steps.	2.36	4.01	69.91%
I found this installation process efficient and effective.	2.27	4.27	88.11%
Overall	2.39	4.07	70.22%

3.5. Thematic analysis of user feedback

Qualitative feedback confirmed the quantitative results, noting clearer steps, greater independence, and reduced reliance on facilitators, as summarized in Table 10. Participants expressed higher confidence and usability, reinforcing TESE findings that automation lowers operational challenges in big data deployment.

Table 10. User-identified challenges and automated solutions

Thematic issue	Challenges in manual installation	Solutions in automated installation
Complex instructions	The instructions were so long, leading to misinterpretation and user errors during execution.	Revised short instructions with step-by-step to execute script to install automatically.
High error rate	Frequent syntax mistakes and missing dependencies made it difficult to complete the setup.	Automated script minimized manual input, ensuring accurate execution and reducing errors.
Dependency issues	Manual installation required manually resolving dependencies, causing delays and compatibility issues.	Pre-installed dependencies in the automated script ensured a smooth and conflict-free installation.
Facilitator dependency	Many steps were unfamiliar, requiring continuous facilitator assistance, especially in configuration settings.	Automated configurations, pre-set environment variables, and guided prompts reduced reliance on facilitators.
Installation time constraints	The process was lengthy, requiring multiple command entries and troubleshooting errors.	Automation reduced installation time through script execution, eliminating manual steps.

3.6. Opportunities for improvement

Although automation improved user experience, additional enhancements were identified after deployment in Table 11. Participants also noted further improvement opportunities in Table 12, including diagnostic, compatibility, and recovery features to increase framework robustness.

Table 11. Improvements after automation

First method's problem (Manual installation)	User experience in second method (Automated installation)
Instructions lacked clarity and detail.	The second practice provided clearer, short, structured instructions with shell automated execution.
Installation required frequent facilitator assistance.	Automation made the process more independent, reducing the need for facilitator support.
Errors occurred frequently due to manual configuration issues.	The script minimized errors by automatically configuring settings and resolving dependencies.
Manual command entry was repetitive and time-consuming.	The automated process optimized installation, reducing typing effort and execution time.

Table 12. Opportunities for improvement in automated installation

Feedback from automated method	Proposed improvement
There is no log record if an error occurs.	Add error logging functionality to the script.
Changes in Hadoop, Sqoop, or Python official download URLs cause failures.	Mirror these tools in a GitHub repository for fallback installation if official URLs fail.
Only works on Linux Ubuntu-based systems.	Expand compatibility to include additional Linux distributions.
No rollback function if installation is interrupted.	Implement a rollback mechanism to restore the previous stable state after failure.

3.7. Key findings comparison with existing studies

Time efficiency improved by 63.04%, with the automated script significantly reducing installation time, consistent with prior automation research in Table 13.

Table 13. Time efficiency improvement compared with prior automation studies

Study	Manual Installation	Automated Installation	Time Efficiency Improvement
This Study	67.28 min	24.87 min	63.04%
Python Script [29]	213 min	22 min	89.67%
RPA [30]	8.33 min	0.83 min	90.04%
APC tool [31]	21.7 min	11.1 min	48.85%

Automation reduced errors by 81.11%, with manual installations averaging 27 errors versus 5.1 for automated. This significant reduction ($p<0.001$) highlights the manual method's susceptibility to misconfigurations in Table 14.

Table 14. Comparison of error rate reduction between this study and prior automation research

Study	Manual Installation	Automated Installation	Error Rate Reduction
This Study	27 min	5.10 min	81.11%
RPA [30]	12%	0%	100%
APC tool [31]	5 plans	1 plan	80.00%
ShellBreaker [32]	34.1 false negative rate	8.30 false negative rate	75.66%

Survey results showed a 70.22% increase in perceived installation ease with automation. Participants reported clearer steps, less reliance on facilitators, and reduced troubleshooting due to guided prompts and preconfigured settings. This aligns with prior studies [15], [33], confirming that reducing manual complexity improves usability and supports IT solution adoption.

3.8. Key findings comparison with framework

The results support FADTESE. From (2), the AD value is:

$$\text{AD} = 1\left[\frac{1+1+1}{3} = 1.0 \geq 0.8\right] = 1$$

Confirming all prerequisites. Using (3), the TESE results were:

Time improvement: $\text{T} = \frac{T_m - T_a}{T_m} = \frac{67.28-24.87}{67.28} = 42.41 - 67.28 = 0.6304$

Error reduction: $\text{E} = \frac{E_m - E_a}{E_m} = \frac{27.00-5.10}{27.00} = 21.90 - 27.00 = 0.8111$

Satisfaction gain: $\text{S} = \frac{S_a - S_m}{5} = \frac{4.070-2.392}{5} = 1.682/5 = 0.3364$

Figures 2 and 3 show the ADRI check and deployment output confirming prerequisites.

Applying Bootstrap resampling (10 000 samples, 95 % CI), the TESE score was:

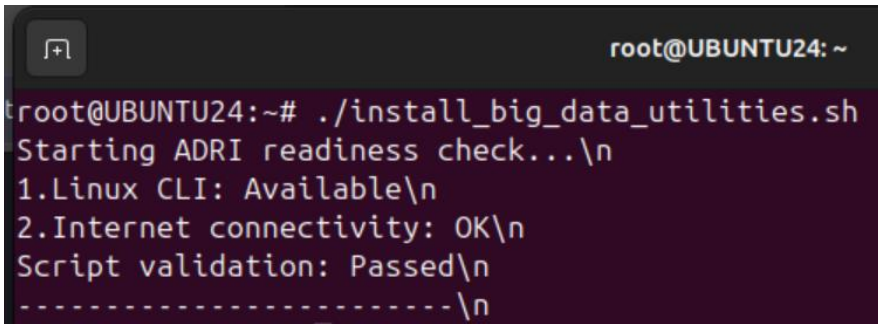
$$\text{TESE} = \text{Bootstrap}\left(\frac{1}{3}(0.6304 + 0.8111 + 0.3364)\right) = \text{Bootstrap}(0.5926) = 0.5941$$

Corresponding to a moderate effectiveness level (59.41%). Substituting the AD and TESE values into (1) confirms moderate overall effectiveness with full readiness.

$$\text{CS} = 1 \times 0.5941 = 0.5941 \text{ or } 59.41\%$$

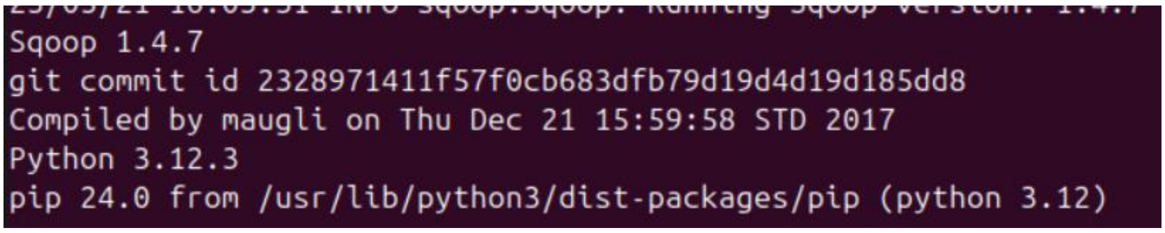
3.9. Practical implications and limitations

Although the FADTESE framework achieved moderate overall effectiveness (CS=59.41%), several limitations remain. The model is platform-dependent and uses equal-weight indices that may not reflect diverse operational contexts. Bootstrap validation is constrained by sample size and controlled conditions, and the framework evaluates deployment but not post-deployment performance. Future work should enable weighted and adaptive readiness scoring, cross-platform support, dynamic error recovery, and broader validation in industrial settings.



```
root@UBUNTU24: ~
root@UBUNTU24:~# ./install_big_data_utilities.sh
Starting ADRI readiness check...\n
1.Linux CLI: Available\n
2.Internet connectivity: OK\n
Script validation: Passed\n
-------------------------\n
```

Figure 2. ADRI readiness verification



```
Sqoop 1.4.7
git commit id 2328971411f57f0cb683dfb79d19d4d19d185dd8
Compiled by maugli on Thu Dec 21 15:59:58 STD 2017
Python 3.12.3
pip 24.0 from /usr/lib/python3/dist-packages/pip (python 3.12)
```

Figure 3. Automated deployment execution output

4. CONCLUSION

This study presented FADTESE as a unified framework that integrates automated deployment with structured effectiveness evaluation for big data tools. By combining the automated deployment model with the time, error, and satisfaction evaluation model, the framework provides a complete approach to addressing the configuration complexity, dependency issues, and inconsistent results typically observed in manual installation of Hadoop, Sqoop, and Python. Results from eighty IT practitioners demonstrated that the environment was fully prepared for automation, as indicated by an ADRI value of 1.0, and that automation led to substantial improvements in installation outcomes. Automation reduced installation time, minimized configuration errors, and increased user satisfaction, leading to a TESE score of 0.5941 and a Composite Score of 59.41%, confirming moderate overall effectiveness with full readiness. Beyond quantitative gains, qualitative feedback showed that automation improved clarity, reduced facilitator dependence, and increased confidence during deployment. These findings validate the framework demonstrate that a combined automation-and-evaluation approach can create more consistent, reliable, and reproducible deployment processes for big data environments. While the framework performed effectively, limitations remain in platform dependency, lack of rollback and logging features, and fixed metric weighting. Future enhancements will include expanding platform support, integrating diagnostic and recovery mechanisms, and validating the approach in broader industrial contexts to further strengthen the robustness and scalability of FADTESE.

ACKNOWLEDGMENTS

The authors thank their co-authors for guidance, peer reviewers for valuable feedback. They also acknowledge the support of the Lincoln University College Doctoral Program.

FUNDING INFORMATION

This research received no external funding.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Mony Ho	✓	✓				✓			✓	✓			✓	
Sokroeurn Ang		✓				✓				✓				
Sopheatra Huy		✓								✓				
Midhunchakkaravarthy Janarthanan										✓		✓		

C	: **C**onceptualization	I	: **I**nvestigation	Vi	: **Vi**sualization
M	: **M**ethodology	R	: **R**esources	Su	: **Su**pervision
So	: **So**ftware	D	: **D**ata Curation	P	: **P**roject administration
Va	: **Va**lidation	O	: Writing - **O**riginal Draft	Fu	: **Fu**nding acquisition
Fo	: **Fo**rmal analysis	E	: Writing - Review & **E**diting		

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are openly available in GitHub [43]. The dataset generated and analyzed during the current study is available from the corresponding author upon reasonable request.

REFERENCES

[1] T. H. Davenport and D. J. Patil, "Data scientist: The sexiest job of the 21st century," *Harvard Business Review*, vol. 90, no. 10, p. 5, 2012.

[2] V. Dhar, "Data science and prediction," *Communications of the ACM*, vol. 56, no. 12, pp. 64–73, 2013, doi: 10.1145/2500499.

[3] F. Provost and T. Fawcett, "Data science and its relationship to big data and data-driven decision making," *Big Data*, vol. 1, no. 1, pp. 51–59, 2013, doi: 10.1089/big.2013.1508.

[4] A. K. Sandhu, "Big data with cloud computing: Discussions and challenges," *Big Data Mining and Analytics*, vol. 5, no. 1, p. 32, 2022, doi: 10.26599/BDMA.2021.9020016.

[5] C. Ma, M. Zhao, and Y. Zhao, "An overview of Hadoop applications in transportation big data," *Journal of Traffic and Transportation Engineering (English Edition)*, vol. 10, no. 5, pp. 900–917, 2023, doi: 10.1016/j.jtte.2023.05.003.

[6] R. C. Taylor, "An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics," *BMC Bioinformatics*, vol. 11, no. SUPPL. 12, 2010, doi: 10.1186/1471-2105-11-S12-S1.

[7] S. Landset, T. M. Khoshgoftaar, A. N. Richter, and T. Hasanin, "A survey of open source tools for machine learning with big data in the Hadoop ecosystem," *Journal of Big Data*, vol. 2, no. 1, pp. 5–6, 2015, doi: 10.1186/s40537-015-0032-1.

[8] D. Geng, C. Zhang, C. Xia, X. Xia, Q. Liu, and X. Fu, "Big data-based improved data acquisition and storage system for designing industrial data platform," *IEEE Access*, vol. 7, pp. 44574–44582, 2019, doi: 10.1109/ACCESS.2019.2909060.

[9] T. J. Pollard, A. E. W. Johnson, J. D. Raffa, and R. G. Mark, "Tableone: An open source Python package for producing summary statistics for research papers," *JAMIA Open*, vol. 1, no. 1, pp. 26–31, 2018, doi: 10.1093/jamiaopen/ooy012.

[10] J. Demšar *et al.*, "Orange: Data mining toolbox in python," *Journal of Machine Learning Research*, vol. 14, pp. 2349–2353, 2013.

[11] I. A. Ajah and H. F. Nweke, "Big data and business analytics: Trends, platforms, success factors and applications," *Big Data and Cognitive Computing*, vol. 3, no. 2, pp. 1–30, Jun. 2019, doi: 10.3390/bdcc3020032.

[12] J. Li *et al.*, "Challenges to error diagnosis in Hadoop ecosystems," in *27th Large Installation System Administration Conference, LISA 2013*, 2013, pp. 145–154.

[13] W. C. Chung *et al.*, "CloudDOE: A user-friendly tool for deploying Hadoop clouds and analyzing high-throughput sequencing data with MapReduce," *PLoS ONE*, vol. 9, no. 6, pp. 2–3, 2014, doi: 10.1371/journal.pone.0098146.

[14] Z. Zhong, S. He, H. Wang, B. Yu, H. Yang, and P. He, "An empirical study on package-level deprecation in Python ecosystem," in *Proceedings - International Conference on Software Engineering*, 2025, pp. 66–77, doi: 10.1109/ICSE55347.2025.00046.

[15] J. Fan, F. Han, and H. Liu, "Challenges of big data analysis," *National Science Review*, vol. 1, no. 2, pp. 293–314, 2014, doi: 10.1093/nsr/nwt032.

[16] B. Kim and G. Henke, "Easy-to-use cloud computing for teaching data science," *Journal of Statistics and Data Science Education*, vol. 29, no. S1, pp. S103–S111, 2021, doi: 10.1080/10691898.2020.1860726.

[17] D. Blazquez and J. Domenech, "Big data sources and methods for social and economic analyses," *Technological Forecasting and Social Change*, vol. 130, pp. 99–113, 2018, doi: 10.1016/j.techfore.2017.07.027.

[18] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, "Deep learning applications and challenges in big data analytics," *Journal of Big Data*, vol. 2, no. 1, pp. 6–7, 2015, doi: 10.1186/s40537-014-0007-7.

[19] X. W. Chen and X. Lin, "Big data deep learning: Challenges and perspectives," *IEEE Access*, vol. 2, pp. 514–525, 2014, doi: 10.1109/ACCESS.2014.2325029.

[20] J. V. Gautam, H. B. Prajapati, V. K. Dabhi, and S. Chaudhary, "Empirical study of job scheduling algorithms in Hadoop mapreduce," *Cybernetics and Information Technologies*, vol. 17, no. 1, pp. 146–163, 2017, doi: 10.1515/cait-2017-0012.

[21] D. K. U. Pavan, D. M. N. Nachappa, and D. S. V. N. Srinivasu, "Sqoop usage in Hadoop distributed file system and observations to handle common errors," *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 9, no. 4, pp. 452–454, 2020, doi: 10.35940/ijrte.d4980.119420.

[22] G. Lindstrom, "Programming with Python," *IT Professional*, vol. 7, no. 5, pp. 10–16, 2005, doi: 10.1109/MITP.2005.120.

[23] M. Sishi and A. Telukdarie, "Adoption of data-driven automation techniques to create smart key performance indicators for business optimization," *Applied System Innovation*, vol. 8, no. 1, p. 1, 2025, doi: 10.3390/asi8010010.

[24] M. Schröder and J. Cito, "An empirical investigation of command-line customization," *Empirical Software Engineering*, vol. 27, no. 2, p. 2, 2022, doi: 10.1007/s10664-021-10036-y.

[25] O. Nyarko-Boateng, I. K. Nti, A. A. Mensah, and E. K. Gyamfi, "Controlling user access with scripting to mitigate cyber-attacks," *Scientific African*, vol. 26, pp. 1–7, 2024, doi: 10.1016/j.sciaf.2024.e02355.

[26] A. A. Khan and others, "A cost-effective approach using generative AI and gamification to enhance biomedical treatment and real-time biosensor monitoring," *Scientific Reports*, vol. 15, no. 17305, p. 13, 2025.

[27] A. A. Khan *et al.*, "A lightweight scalable hybrid authentication framework for Internet of Medical Things (IoMT) using blockchain hyperledger consortium network with edge computing," *Scientific Reports*, vol. 15, no. 1, pp. 6–8, 2025, doi: 10.1038/s41598-025-05130-w.

[28] G. Tierney, "A Python-based automation script to mark computer-aided design assessments," *Applied Sciences (Switzerland)*, vol. 15, no. 3, p. 1,5, 2025, doi: 10.3390/app15031203.

[29] S. A. Mohamed, M. A. Mahmoud, M. N. Mahdi, and S. A. Mostafa, "Improving efficiency and effectiveness of robotic process automation in human resource management," *Sustainability (Switzerland)*, vol. 14, no. 7, pp. 14–15, 2022, doi: 10.3390/su14073920.

[30] S. Liu *et al.*, "Optimizing efficiency and safety in external beam radiotherapy using automated plan check (APC) tool and six sigma methodology," *Journal of Applied Clinical Medical Physics*, vol. 20, no. 8, pp. 56–64, 2019, doi: 10.1002/acm2.12678.

[31] Y. Li, J. Huang, A. Ikusan, M. Mitchell, J. Zhang, and R. Dai, "ShellBreaker: Automatically detecting PHP-based malicious web shells," *Computers and Security*, vol. 87, pp. 10–11, 2019, doi: 10.1016/j.cose.2019.101595.

[32] S. Gavrila Gavrila, C. Blanco González-Tejero, J. A. Gómez Gandía, and A. de Lucas Ancillo, "The impact of automation and optimization on customer experience: a consumer perspective," *Humanities and Social Sciences Communications*, vol. 10, no. 1, pp. 6–8, 2023, doi: 10.1057/s41599-023-02389-0.

[33] M. Y. Ivory and M. A. Hearst, "The state of the art in automating usability evaluation of user interfaces," *ACM Computing Surveys*, vol. 33, no. 4, pp. 470–516, 2001, doi: 10.1145/503112.503114.

[34] K. Senjab, S. Abbas, N. Ahmed, and A. ur R. Khan, "A survey of Kubernetes scheduling algorithms," *Journal of Cloud Computing*, vol. 12, no. 1, pp. 1–2, 2023, doi: 10.1186/s13677-023-00471-1.

[35] A. Parasuraman, "Technology Readiness Index (TRI): A multiple-item scale to measure readiness to embrace new technologies," *Journal of Service Research*, vol. 2, no. 4, pp. 307–320, 2000, doi: 10.1177/109467050024001.

[36] U. Nations, *E-government survey 2020: digital government in the decade of action for sustainable development*. New York, NY, USA: United Nations, 2020.

[37] B. Sauser, D. Verma, J. Ramirez-Marquez, and R. Gove, "From TRL to SRL: The concept of systems readiness levels," in *Proc. Conf. Systems Eng. Research*, 2006, pp. 14–15.

[38] A. A. Khan *et al.*, "BDLT-IoMT—a novel architecture: SVM machine learning for robust and secure data processing in Internet of Medical Things with blockchain cybersecurity," *Journal of Supercomputing*, vol. 81, no. 1, p. 6, 2025, doi: 10.1007/s11227-024-06782-7.

[39] A. A. Laghari *et al.*, "A novel and secure artificial intelligence enabled zero trust intrusion detection in industrial internet of things architecture," *Scientific Reports*, vol. 15, no. 1, pp. 7–12, 2025, doi: 10.1038/s41598-025-11738-9.

[40] A. A. Khan and others, "Cybersecurity, digital forensics, and the IoT for deepfake investigation on social media platforms: a review," *Human-centric Computing and Information Sciences*, vol. 15, pp. 11–15, 2025, doi: 10.22967/HCIS.2025.15.038.

[41] A. A. Khan and others, "Quantum computing empowering blockchain technology with post-quantum resistant cryptography for multimedia data privacy preservation in cloud-enabled public auditing platforms," *Journal of Cloud Computing*, vol. 14, no. 43, pp. 3–8, 2025, doi: 10.1186/s13677-025-00771-8.

[42] M. Tavakol and R. Dennick, "Making sense of Cronbach's alpha," *International Journal of Medical Education*, vol. 2, pp. 53–55, 2011, doi: 10.5116/ijme.4dfb.8dfd.

[43] DataScienceSource, "Bash script for automated big data tool installation." 2025.

[44] A. Bangor, P. Kortum, and J. Miller, "Determining what individual SUS scores mean: Adding an adjective rating scale," *Journal of Usability Studies*, vol. 4, no. 3, pp. 114–123, 2009, doi: 10.5555/2835587.2835589.

[45] B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap*. New York, NY: Chapman & Hall/CRC, 1993.

[46] E. O. George, "Empirical research: A comprehensive guide for academics." 2024.

[47] C. Uakarn, K. Chaokromthong, and N. Sintao, "Sample size estimation using Yamane and Cochran and Krejcie and Morgan and Green formulas and Cohen statistical power analysis by G*Power and comparisons," *APHEIT Int. J.*, vol. 10, no. 2, p. 78, 2021.

[48] S. Saha, "What is experimental research: Definition, types & examples." 2024.

[49] M. Xu, D. Fralick, J. Z. Zheng, B. Wang, X. M. Tu, and C. Feng, "The differences and similarities between two-sample t-test and paired t-test," *Shanghai Arch. Psychiatry*, vol. 29, no. 3, pp. 185–186, 2017, doi: 10.11919/j.issn.1002-0829.217070.

[50] S. Ahmad, S. Wasim, S. Irfan, S. Gogoi, A. Srivastava, and Z. Farheen, "Qualitative v/s. quantitative research – A summarized review," *Journal of Evidence Based Medicine and Healthcare*, vol. 6, no. 43, pp. 2828–2832, 2019, doi: 10.18410/jebmh/2019/587.

[51] H. N. Boone and D. A. Boone, "Analyzing Likert data," *Journal of Extension*, vol. 50, no. 2, pp. 3–6, 2012, doi: 10.34068/joe.50.02.48.

[52] W. M. Lim, "What is qualitative research? An overview and guidelines," *Australasian Marketing Journal*, vol. 33, no. 2, pp. 199–229, May 2025, doi: 10.1177/14413582241264619.

[53] P. Kumwichar, "Enhancing learning about epidemiological data analysis using R for graduate students in medical fields with Jupyter Notebook: Classroom action research," *JMIR Medical Education*, vol. 9, no. e47394, pp. 2–4, 2023, doi: 10.2196/47394.

BIOGRAPHIES OF AUTHORS

Mony Ho is a Ph.D. candidate in Information Technology at Lincoln University College, Malaysia, and holds a master's degree in IT and data science from the European International University, France, and Master Trainer from the Republic of Korea. He serves as a master trainer at the Ministry of Labor and Vocational Training and lectures part-time at several universities in Cambodia. His teaching and research interests include data science and cybersecurity. He can be contacted at email: mho.phdscholar@lincoln.edu.my.

Sokroeurn Ang is a Ph.D. candidate in cybersecurity at Lincoln University College, Malaysia, MSC in cybersecurity at University of London, UK, Micro master in cybersecurity at RIT, USA, Certified with CISSP, CISA, CISM, ECSA, CEH, CyberOps, CCNASec, CCNA, and AWS cloud practitioner. He can be contacted at email: angsokroeurn.phdscholar@lincoln.edu.my.

Sopheatra Huy is a Ph.D. candidate in information technology at Lincoln University College, Malaysia, and holds an M.Sc. in IT from the Royal University of Phnom Penh. With over a decade of part-time lecturing experience, he has taught programming, software engineering, and IT project management. He currently serves as Senior Manager of IT Audit at WB Finance, with prior roles at Phillip Bank and PRASAC MFI. His research interests include IT automation, cybersecurity, and audit technologies. He can be contacted at email: hsopheaktra.phdscholar@lincoln.edu.my.

Midhunchakkaravarthy Janarthanan is the Dean of the School of AI Computing and Multimedia at Lincoln University College, Malaysia. He holds a Ph.D. in Computer Science and has published widely in Big Data, Web Text Mining, Machine Learning, and GPU Computing, with over 1,000 Google Scholar citations. He also supervises postgraduate research in AI and cloud-based systems. He can be contacted at email: midhun@lincoln.edu.my.