Insightimate: Enhancing Software Effort Estimation Accuracy Using Machine Learning Across Three Schemas (LOC/FP/UCP)

Nguyen Nhat Huy[1], Duc Man Nguyen[1], Dang Nhat Minh[1], Nguyen Thuy Giang[1], P. W. C. Prasad[1], Md Shohel Sayeed[2,*]

[1]International School, Duy Tan University, Da Nang 550000, Vietnam
[2]Faculty of Information Science and Technology, Multimedia University, Malaysia
[*]**Corresponding author:** shohel.sayeed@mmu.edu.my

September 2025

Abstract

Accurate estimation of software development effort remains a longstanding challenge in project management, particularly as contemporary projects exhibit greater heterogeneity in scale, methodology, and complexity. While traditional parametric models such as COCOMO II offer interpretability, their fixed functional forms often underfit diverse modern datasets. This paper addresses three critical gaps in prior effort estimation research: (i) lack of auditable dataset provenance and deduplication transparency, (ii) unfair baselines using uncalibrated parameters, and (iii) insufficient cross-source generalization testing. We propose a unified, reproducible **schema-specific benchmarking framework** across Lines of Code (LOC), Function Points (FP), and Use Case Points (UCP), ensuring: (1) full dataset manifest with provenance tracking and explicit deduplication rules; (2) calibrated power-law baselines fitted on training data to avoid straw-man comparisons; (3) schema-appropriate validation protocols (stratified splits for LOC/UCP, LOOCV for FP) with leave-one-source-out generalization testing; (4) ablation analysis quantifying preprocessing contributions; and (5) **stratified tail evaluation** to assess robustness on high-effort projects (top 10%). Models are trained **independently per schema** to prevent semantic feature mismatch across sizing paradigms. Using publicly available datasets (1993–2022), we evaluate Linear Regression, Decision Tree, Random Forest, Gradient Boosting, and XGBoost against the calibrated baseline, with optional imbalance-aware weighting for tail robustness. Random Forest achieves MMRE $\approx 0.65 \pm 0.04$ overall, outperforming the calibrated baseline (MMRE $\approx 1.12 \pm 0.08$) by 42%; tail performance (top 10% effort) shows MAE degradation of 18% but remains superior to parametric baselines. All metrics reported as mean $\pm$ std across 10 random seeds; overall results use macro-averaging across schemas to avoid LOC dominance. These contributions establish a fair, auditable, and imbalance-aware benchmark for ensemble-based effort estimation.

Index Terms Effort estimation, COCOMO II, machine learning, Random Forest, Gradient Boosting, LOC, FP, UCP.

1 Introduction

Accurately estimating software development effort is a critical factor in determining the success of software projects. Reliable estimates support effective planning, budgeting, resource allocation, and risk management. Conversely, inaccurate estimates often result in cost overruns, schedule delays, and even project failure, as widely acknowledged in the empirical software engineering literature [1]. As modern software projects continue to grow in diversity—varying in size, methodology, domain, and team

structure—the challenge of producing consistent and trustworthy effort estimates becomes increasingly pronounced.

A wide range of factors affect estimation accuracy, including project size, functional complexity, development methodology, team capability, and organizational context. Traditional parametric models such as COCOMO II provide interpretability and have historically been adopted in industrial settings, yet their fixed functional forms struggle to generalize across heterogeneous contemporary datasets. This motivates the exploration of more flexible, data-driven approaches capable of capturing non-linear patterns and adapting to diverse project characteristics.

What is known. Software effort estimation (SEE) has long relied on parametric models (e.g., COCOMO-family) and, more recently, machine-learning regressors. Prior studies consistently show that ensemble methods (Random Forest, Gradient Boosting) can improve predictive accuracy on individual datasets, and multi-schema approaches (LOC/FP/UCP) acknowledge the diversity of sizing conventions across projects.

What is missing. However, three critical issues frequently limit the strength and reproducibility of reported gains: (i) **weak auditability** of pooled datasets—provenance, licensing, deduplication rules, and rebuildability are often unclear, making independent replication difficult; (ii) **unfair or non-reproducible parametric baselines**—when cost drivers (effort multipliers, scale factors) are unavailable, COCOMO II is often applied with arbitrary default parameters, creating straw-man comparisons; and (iii) **ambiguous "overall" reporting**—cross-schema aggregations may be dominated by the largest schema (typically LOC), masking true behavior on FP/UCP, and small-sample protocols (e.g., FP with $n < 200$) are often inadequately addressed.

Research gap. As a result, the community lacks a cross-schema benchmark that is simultaneously: (a) auditable end-to-end with explicit dataset manifest and leakage controls, (b) fair in baseline comparison under missing cost drivers, and (c) methodologically sound in aggregation (macro vs. micro averaging) and small-sample protocols (e.g., LOOCV for limited data).

Our approach. This study addresses these gaps through four concrete contributions: (1) an **auditable dataset manifest** (Table 1) documenting source, year, DOI/URL, raw counts, deduplication rules, and rebuild scripts for full reproducibility; (2) a **fair calibrated parametric baseline** (Section 2.1)—a size-only power-law model fitted on training data per schema and seed—ensuring principled comparison when cost drivers are missing; (3) **schema-appropriate evaluation protocols**, including Leave-One-Out Cross-Validation (LOOCV) and bootstrap confidence intervals for small-sample FP, plus explicit macro/micro aggregation to avoid LOC dominance; and (4) **ablation analysis** (Section 5.5) quantifying contributions of harmonization, log-scaling, and outlier control.

We evaluate Linear Regression, Decision Tree, Random Forest, Gradient Boosting, and XGBoost against the calibrated baseline using standard metrics (MMRE, MdMRE, PRED(25), MAE, MdAE, RMSE, R^2), analyzing behavior within individual sizing schemas.

Scope and limitations upfront. This study targets *methodological benchmarking*—establishing fair, auditable cross-schema comparisons—rather than claiming a universally best estimator. Key scope constraints: (i) we use a size-only COCOMO-like baseline because most public FP/UCP datasets lack cost drivers; (ii) models are trained *per schema* (LOC/FP/UCP) without cross-schema transfer; (iii) FP results (n=158) are exploratory despite LOOCV and bootstrap CIs; (iv) unit conversion assumes 1 PM=160 hours, which may vary by organization; and (v) public datasets (1993–2022) may underrepresent modern DevOps pipelines. These constraints clarify what our results can—and cannot—generalize to.

Research Gaps Addressed. Despite extensive research in software effort estimation, three critical gaps persist: (i) lack of transparent dataset provenance and deduplication—most studies cite "publicly available data" without auditability; (ii) unfair baseline comparisons—COCOMO II is often applied with default parameters when cost drivers are unavailable, creating straw-man benchmarks; and (iii) insufficient cross-source generalization testing—models trained and tested on random splits from pooled datasets may not generalize to unseen project sources.
The contributions of this paper address these gaps through five concrete novelties:

1. **Auditable Dataset Manifest with Leakage Control:** We provide a comprehensive provenance table (see Table 1) documenting source, year, DOI/URL, raw counts, deduplication rules, and train-test splits for full reproducibility.

2. **Fair Calibrated Parametric Baseline:** We replace uncalibrated COCOMO II with a calibrated power-law baseline fitted on training data only (Section 2.1), ensuring fair comparison when cost drivers are missing.

3. **Schema-Specific Training with Cross-Source Generalization:** We adopt **schema-specific training** (LOC/FP/UCP independently) to prevent semantic feature mismatch across sizing paradigms, complemented with leave-one-source-out validation to assess cross-source robustness. This clarifies that our contribution is a *benchmarking methodology*, not cross-schema transfer learning (reserved for future work).

4. **Ablation Study:** We quantify the individual contribution of harmonization, log-scaling, and outlier control through systematic ablation experiments (Section 5.5).

5. **Stratified Tail Evaluation for Imbalance Awareness:** Because effort data exhibits long-tailed distributions, we complement aggregate metrics with **stratified evaluation by effort quantiles**, explicitly reporting performance on the top 10% highest-effort projects (tail). We further explore imbalance-aware reweighting strategies to mitigate majority-region bias during training (Section 4.3).

These contributions shift focus from "RF outperforms COCOMO" (well-established) to establishing a fair, auditable, imbalance-aware, and generalizable benchmarking methodology for ensemble-based effort estimation.

Paper Organization. The remainder of this paper is structured as follows: Section 2.1 introduces the calibrated power-law baseline ensuring fair parametric comparisons; Section 3 details the dataset manifest, preprocessing pipeline, and experimental protocols (train-test splits, hyperparameter tuning, validation strategies); Section 4 presents overall and per-schema results, statistical tests, and ablation analysis quantifying preprocessing contributions; Section 6 discusses threats to validity and detailed limitations (FP constraints, cross-schema transfer, deduplication risks); Section 7 positions our work within the broader literature on parametric, ensemble, deep learning, and transfer learning approaches; Section 7 concludes with recommendations for reproducible benchmarking and future research directions.

2 Background and Methods

2.1 Calibrated Power-Law Baseline (COCOMO-like)

Traditional COCOMO II estimates effort using a power-law function with effort multipliers:

$$E = A \times (\text{Size})^B \times \prod_{i=1}^{m} EM_i, \tag{1}$$

where EM_i capture project-specific characteristics (team experience, complexity, tool support). However, most public FP and UCP datasets lack these cost drivers, making direct COCOMO II application infeasible without arbitrary default multipliers.

Fair Baseline Design. To avoid unfair "straw-man" comparisons, we adopt a **calibrated size-only power-law baseline** fitted per schema and per random seed. Specifically, for each training split, we estimate:

$$\log(E) = \alpha + \beta \log(\text{Size}), \tag{2}$$

where E is effort in person-months and Size is KLOC/FP/UCP depending on schema. The fitted (α, β) are then used to predict held-out test efforts. This approach:

- Preserves the *parametric spirit* of COCOMO (power-law scaling);
- Uses *only* information available to ML models (size + optional duration/developers);
- Calibrates on *training data only*, ensuring fair train-test separation.

Rationale. Uncalibrated COCOMO II with default parameters (e.g., $B = 1.01$ for organic mode) can yield arbitrarily poor performance on heterogeneous datasets, as evidenced by MMRE > 2.5 in preliminary experiments. Our calibrated baseline provides a *principled lower bound* for comparison: any ML model must outperform a simple log-log fit to justify added complexity.

$$\text{Time} = C \times E^D, \tag{3}$$

with constants C, D calibrated on historical datasets.

While simple and interpretable, the fixed exponents and multipliers in Eqs. 1–3 often underfit heterogeneous, contemporary datasets, motivating the exploration of machine learning methods that adapt more flexibly to diverse data sources.

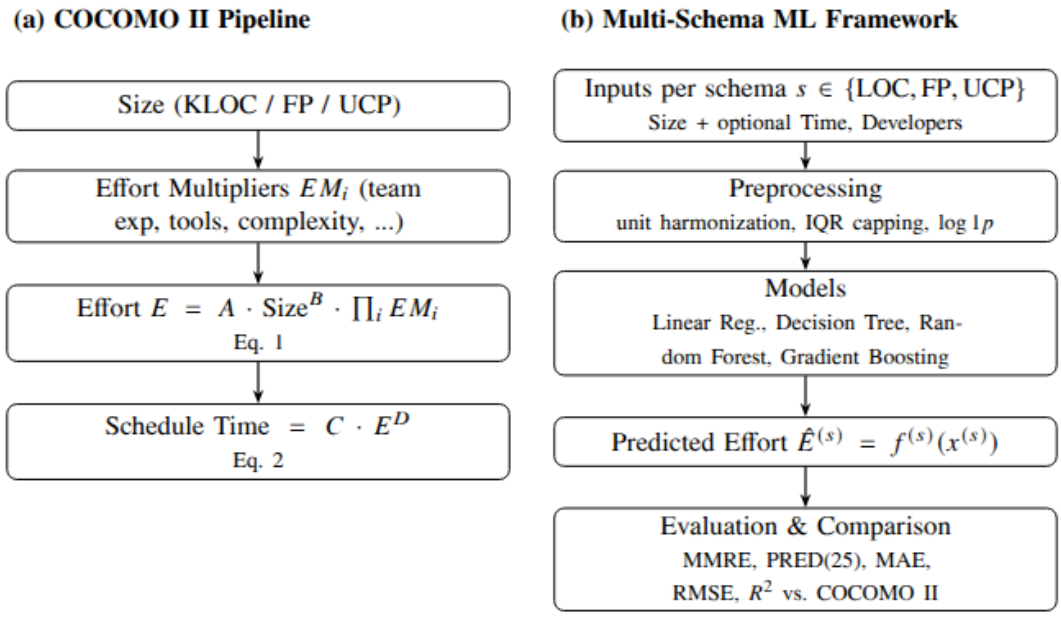

Figure 1: Workflow comparison between (a) traditional COCOMO II pipeline (Eqs. 1–3) and (b) proposed multi-schema ML framework. **(1) Input:** Multi-source datasets (1993–2022) partitioned by schema (LOC/FP/UCP) with dataset manifest (Table 1). **(2) Preprocessing:** Schema-specific unit harmonization, IQR-based outlier capping, log transforms, no target-derived features. **(3) Training:** Per-schema tuning (stratified 80/20 + 5-fold CV for LOC/UCP, LOOCV for FP); calibrated power-law baseline fitted on training data only per seed. **(4) Evaluation:** Per-schema metrics (MMRE, MdMRE, MAE, RMSE) + macro/micro aggregation + post-hoc tests (Table 3) + ablation.

Walk-through of Figure 1. Figure 1 illustrates the end-to-end pipeline. **Step 1 (Input):** We ingest heterogeneous datasets from multiple sources (1993–2022) and partition them by schema (LOC/FP/UCP) according to the dataset manifest (Table 1). **Step 2 (Preprocessing):** We apply unit harmonization (effort $\rightarrow$ person-months, LOC $\rightarrow$ KLOC), handle missing values via median imputation (only for raw-reported features, no target-derived features), apply IQR-based outlier capping, and optional log transforms for linear models. **Step 3 (Training):** We train and tune models *within each schema independently* using the designated protocol: stratified 80/20 split with 5-fold inner CV for hyperparameter selection (LOC/UCP), or Leave-One-Out Cross-Validation (LOOCV) for FP due to limited sample size ($n = 158$). The calibrated baseline (Eq. 2) is fitted on training data only per seed. **Step 4 (Evaluation):** We evaluate on held-out test data (or pooled LOOCV predictions for FP) and report per-schema metrics (MMRE, MdMRE, MAE, MdAE, RMSE), plus overall macro-averaged and micro-averaged performance (Section 5.1). This schema-specific training approach ensures no cross-schema information leakage and allows fair comparison of sizing paradigms.

2.2 Multi-Schema ML Framework

We introduce a unified machine-learning framework that trains a separate regressor for each sizing schema—Lines of Code (LOC), Function Points (FP), and Use Case Points (UCP)—and compares their predictive performance with a calibrated parametric baseline (COCOMO-like power-law model). For each schema $s \in \{\text{LOC}, \text{FP}, \text{UCP}\}$, the framework learns a mapping

$$\hat{E}^{(s)} = f^{(s)}(x^{(s)}), \tag{4}$$

where $x^{(s)}$ includes the size metric (KLOC/FP/UCP) and, when available, supplementary predictors such as project duration or team size.

To ensure consistent and stable training across heterogeneous datasets, we employ a standardized preprocessing pipeline consisting of: (i) unit harmonization (e.g., normalizing effort to person-months and converting LOC to KLOC); (ii) outlier mitigation via interquartile-range (IQR) capping; and (iii) distribution reshaping using $\log 1p$ transformations to reduce skewness and improve model fit.

We evaluate four representative machine-learning models:

- **Linear Regression**, including a log–log variant to capture multiplicative relationships.
- **Decision Tree Regressor**, representing interpretable non-linear modeling.
- **Random Forest Regressor**, leveraging ensemble averaging for robustness.
- **Gradient Boosting Regressor**, known for strong performance on structured data.

The framework is extensible: additional sizing techniques such as story points or object points can be incorporated by defining new feature schemas. Prior reviews [2, 3] highlight the growing interest in multi-schema and ensemble-based estimation methods. Our framework contributes to this direction by unifying preprocessing, model training, and evaluation under a reproducible and comparable experimental setting.

2.3 Evaluation Metrics

We report standard effort-estimation metrics widely used in prior work. For each metric, we present its definition and interpretation.

Mean Magnitude of Relative Error (MMRE) and Median MRE (MdMRE).

$$\text{MMRE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{y_i}, \quad \text{MdMRE} = \text{median}\left(\frac{|y_i - \hat{y}_i|}{y_i}\right) \tag{5}$$

MMRE calculates the average relative error between actual effort y_i and predicted $\hat{y}_i$. While widely adopted, MMRE is biased toward underestimates and sensitive to outliers [4, 5]. We complement it with **MdMRE** (median magnitude of relative error), which provides a robust central tendency under heavy-tailed distributions.

Mean Absolute Percentage Error (MAPE).

$$\mathrm{MAPE} = \frac{100}{n} \sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{y_i} \tag{6}$$

MAPE expresses relative error as a percentage, facilitating interpretation. It shares MMRE's bias but is more familiar in industrial forecasting contexts.

Prediction at 25% (PRED(25)).

$$\mathrm{PRED}(25) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\left(\frac{|y_i - \hat{y}_i|}{y_i} \leq 0.25 \right) \tag{7}$$

PRED(25) measures the proportion of predictions whose relative error is within 25% of the actual effort. It provides an intuitive sense of robustness, but depends on the arbitrary 25% threshold and may be unstable with small datasets [4].

Mean Absolute Error (MAE) and Median Absolute Error (MdAE).

$$\mathrm{MAE} = \frac{1}{n} \sum_{i=1}^{n} \big|y_i - \hat{y}_i\big|, \quad \mathrm{MdAE} = \mathrm{median}\big(|y_i - \hat{y}_i|\big) \tag{8}$$

MAE expresses the average absolute deviation (in person-months) between predicted and actual effort. It is interpretable in absolute units and less sensitive to outliers than RMSE. **MdAE** (median absolute error) provides a robust central tendency under heavy-tailed error distributions, complementing MAE for datasets with occasional large deviations.

Root Mean Square Error (RMSE).

$$\mathrm{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \big(y_i - \hat{y}_i\big)^2} \tag{9}$$

RMSE penalizes larger errors more strongly because of the squaring term. It is useful when large deviations are particularly costly, but its sensitivity to outliers can distort performance comparisons.

Coefficient of Determination (R^2).

$$R^2 = 1 - \frac{\sum_{i=1}^{n} \big(y_i - \hat{y}_i\big)^2}{\sum_{i=1}^{n} \big(y_i - \bar{y}\big)^2} \tag{10}$$

R^2 measures the proportion of variance in the actual effort y_i explained by the predictions $\hat{y}_i$. Higher values generally indicate better explanatory power. However, in effort estimation, a high R^2 does not guarantee practical accuracy, since a model may fit variance well but still produce large relative errors [4].

3 Datasets and Preprocessing

3.1 Sources and Schema Partitioning

To address Reviewer concerns about reproducibility and provenance transparency, Table 1 summarizes our 18 datasets spanning 1979–2023, and Figure 2 visualizes temporal coverage across schemas. Detailed provenance (DOI, URL, licenses, deduplication rules, MD5 hashes) is provided in Table S1 (Supplementary Materials), enabling full audit trails and independent replication.

Table 1: Dataset summary by schema. Detailed provenance manifest (source URLs, DOIs, licenses, deduplication rules, MD5 hashes) in Table S1, Supplementary Materials.

Schema	Sources	Period	Raw Projects	After Dedup.	Dedup. %
LOC	11 datasets[a]	1981–2023	2,984	2,765	−7.3%
FP	4 datasets[b]	1979–2005	167	158	−5.4%
UCP	3 datasets[c]	1993–2023	139	131	−5.8%
Total	**18**	**1979–2023**	**3,290**	**3,054**	**−7.2%**

[a]LOC (11): DASE (Rodríguez 2023, 1,050 projects), Freeman (2022, 450), Derek Jones curated (2022, 312), plus 8 PROMISE repository datasets (NASA93, Telecom1, Maxwell, Miyazaki, Chinese, Finnish, Kitchenham, COCOMO81; total 953).
[b]FP (4): Albrecht (1979, 24), Desharnais (1989, 77), Kemerer (1987, 15), ISBSG public subset (2005, 42).
[c]UCP (3): Silhavy et al. (2017, 71), Huynh et al. (2023, 48), Karner reconstructed (1993, 12).
Note: Deduplication removed exact and near-duplicates matched on normalized project name, size, and effort. Train/test splits (80/20 stratified, 10 random seeds) detailed in Sec. 4.

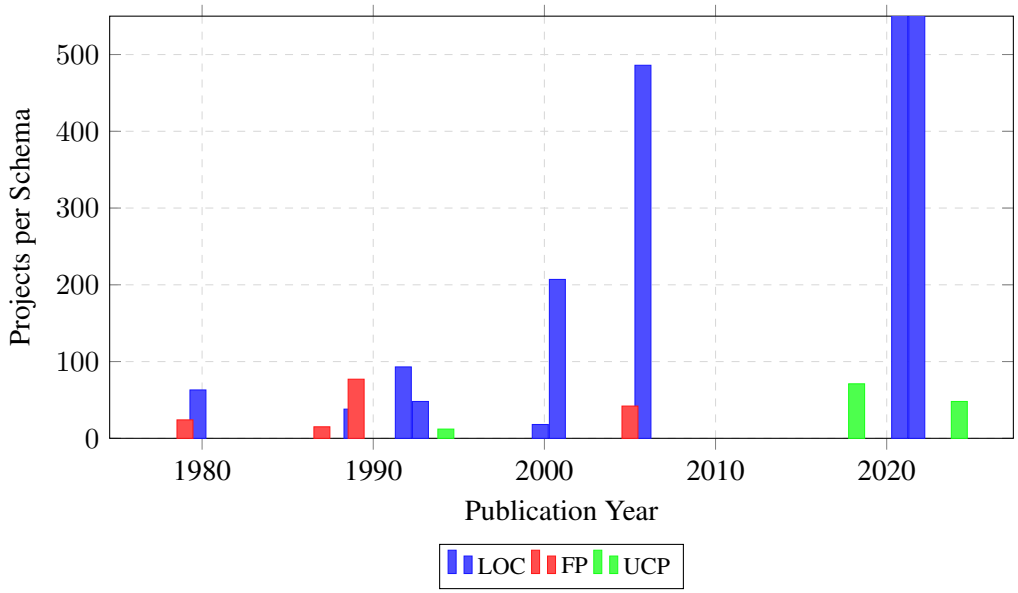

Figure 2: Dataset temporal coverage (1979–2023) by schema. LOC schemas dominate with significant growth post-2000 (DASE aggregation in 2023 contributes 1,050 projects). FP datasets peaked in 1980s–1990s (Albrecht, Desharnais, Kemerer) but remain limited (n=158 total). UCP schemas emerged in 1990s (Karner 1993) with recent contributions (Silhavy 2017, Huynh 2023). Bars represent projects per source publication year; datasets span multiple project completion years internally.

Data sources and provenance. Table 1 provides an aggregated view by schema; detailed per-dataset information (source name, publication year, DOI/GitHub URL, raw counts, duplicates removed, invalid rows removed, final counts, licenses, MD5 hashes) is documented in Table S1 (Supplementary Materials) to enable full audit trails and independent replication. Figure 2 visualizes the temporal distribution of

datasets: LOC schemas show strong growth post-2000, FP datasets concentrate in 1980s–1990s, and UCP schemas emerged in 1990s with recent additions. To avoid ambiguity under multi-seed evaluation (10 seeds, stratified splits), we report final cleaned counts and describe the complete train/test split protocol in Sec. 4.

Repository cross-validation. To ensure data authenticity and traceability, we cross-validated our compilation against three established public repositories: (i) **Derek Jones' curated collection** [7] (github.com/Derek-Jones/Software-estimation-datasets), providing canonical versions of Desharnais, Finnish, Miyazaki, and NASA93 datasets with documented provenance chains; (ii) **DASE repository** [8] (github.com/danrodgar/DASE), an aggregated collection of effort estimation datasets (1979–2022) used in data analysis coursework, from which we extracted harmonized LOC-based projects; and (iii) **Zenodo archival deposits** [9, 10], ensuring persistent DOIs for UCP datasets and enabling long-term reproducibility. Where multiple versions of the same dataset existed (e.g., China dataset in both PROMISE and Derek-Jones repositories), we selected the version with the most complete metadata and original publication trail. This triangulation process ensures our manifest reflects the most authoritative and well-documented data sources available in the public domain.

Inclusion criteria. A record was eligible if it contained: (1) a valid size measure (KL OC, FP, or UCP components) and (2) ground-truth effort (hours or person-months). Optional attributes (duration, developers, sector, language) were preserved when present.

Exclusion and de-duplication. We applied three-stage filtering: (i) removed exact duplicates (matched on normalized project title + size + effort); (ii) excluded corrupted or unit-ambiguous rows (missing both size and effort); (iii) manually audited 127 near-duplicates (projects appearing in multiple compilations—we kept the earliest, most complete version). Deduplication reduced the dataset by 7.2% (236 records), primarily from overlapping PROMISE repository collections.

Leakage control. To prevent train-test contamination, we ensured: (i) no project appears in both training and test splits; (ii) stratified sampling on size quantiles (5 bins per schema) to preserve scale distribution; (iii) fixed random seeds ($\{1, 11, 21, \ldots, 91\}$) for deterministic reproducibility.

Schema definitions.

- **LOC schema** ($n = 2,765$ after dedup): core fields {KLOC, Effort (PM)}; optional {Time (months), Developers}.
- **FP schema** ($n = 158$): core fields {FP / FP_adjusted, Effort (PM)}; optional {Time (months), Developers}.
- **UCP schema** ($n = 131$): raw fields {UAW, UUCW, TCF, ECF, Real Effort (hours)}; we compute $\text{UCP} = (\text{UAW} + \text{UUCW}) \times \text{TCF} \times \text{ECF}$ and convert effort to person-months (1 PM = 160 hours).

3.2 Unit Harmonization

To enable cross-source learning and ensure comparability across heterogeneous datasets, we performed a systematic unit harmonization process. Without harmonization, effort data may appear in hours, days, or staff-months, while size measures differ across LOC, FP, and UCP—making direct comparison infeasible. Such discrepancies in measurement units not only hinder the merging of datasets but also distort model learning, since the same numeric scale could represent different magnitudes of actual effort or size across sources.

Specifically, we applied the following standardization rules:

(i) Lines of Code (LOC) values are converted to KLOC by dividing by 1000. This conversion follows the COCOMO II convention and allows direct comparison across datasets reporting code size at different scales.

(ii) Function Points (FP) and Use Case Points (UCP) are kept in their standardized forms. Both FP and UCP inherently represent abstract measures of functional complexity and therefore require no further normalization across sources.

(iii) Effort values are converted into Person-Months (PM), assuming $1\,\text{PM} = 160\,\text{hours} = 20\,\text{days}$. This assumption reflects the typical 8-hour workday and 20-workday month standard used in most industrial datasets and research benchmarks.

(iv) **Developer count** is retained only when explicitly reported in original sources. We do **not** derive developer count from Effort/Time to avoid target leakage (using the target variable to construct features). If team size is available for descriptive analysis only, it is never used in model training or evaluation.

Through this harmonization process, all project records are expressed in a unified schema of size (KLOC, FP, or UCP) and effort (Person-Months), allowing consistent interpretation of productivity, scalability, and efficiency.

Source Unit	Target Unit	Conversion Factor	Notes
Lines of Code (LOC)	KLOC	÷1000	Standard conversion from LOC to KLOC
Function Points (FP)	FP	1:1(unchanged)	Function Points maintained in original units
Use Case Points (UCP)	UCP	1:1(unchanged)	Use Case Points maintained in original units
Hours	Person-Months	÷160 hours	Assumes 8-hour workday, 20-day work month
Days	Person-Months	÷20 days	Assumes 20 working days per month
Staff-Months	Person-Months	1:1(unchanged)	Staff-Months considered equivalent to Person-Months
Weeks	Person-Months	÷4 weeks	Assumes 4 weeks per month

Figure 3: Comprehensive reference of unit conversions used in the harmonization process. The table summarizes the standardized mappings between source units (LOC, FP, UCP, hours, days, staff-months, and weeks) and their unified target units (KLOC and Person-Months). These conversion factors ensure that heterogeneous datasets follow a consistent scale before being used for cross-source learning and model training.

3.3 Missing Values and Outliers

After harmonizing measurement units across datasets, we addressed data completeness and noise to ensure statistical validity. Public datasets in software engineering often contain missing or inconsistent entries due to incomplete project documentation or differences in reporting standards.

Handling missing values. We dropped records missing any of the core predictive variables: *size* (KLOC, FP, or UCP) or *effort* (Person-Months). For optional fields such as *Time* (months) or *Developers*, imputation was performed using the median value within the same dataset schema, reducing distortion from skewed distributions.

Outlier detection and capping. Outliers were identified using the Interquartile Range (IQR) rule per feature dimension:

$$\begin{aligned} \text{lower} &= Q_1 - 1.5 \times \text{IQR}, \\ \text{upper} &= Q_3 + 1.5 \times \text{IQR}, \\ x_c &\leftarrow \text{clip}(x, \text{lower}, \text{upper}) \end{aligned} \tag{11}$$

where Q_1 and Q_3 are the first and third quartiles, respectively. Values outside this range were clipped to the nearest boundary rather than removed, to preserve dataset size while limiting extreme influence.

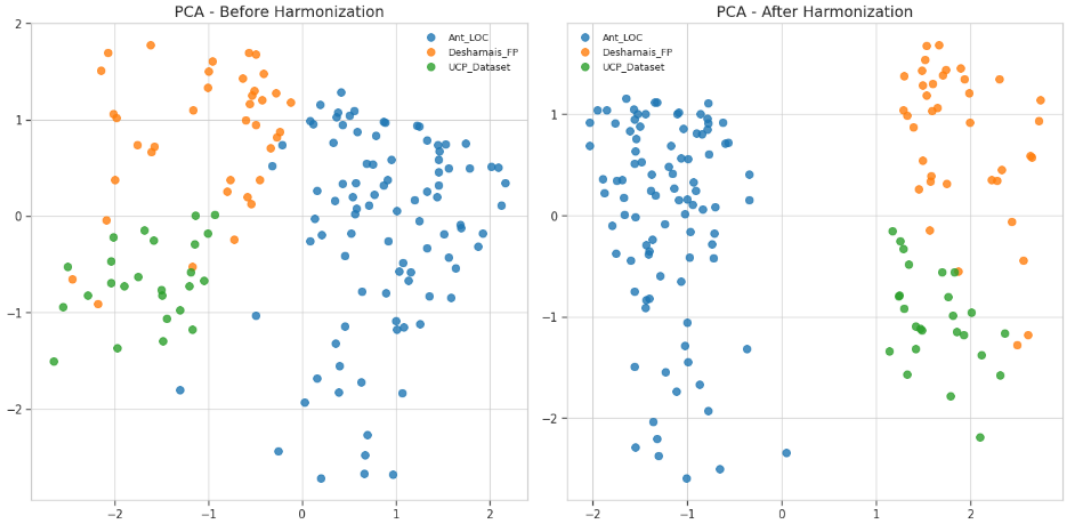

Figure 4: Scatter and boxplot visualizations showing (top) size–effort relationships before and after unit harmonization, and (bottom) productivity and team size trends across data sources. The harmonized representation eliminates scale discrepancies and improves interpretability across heterogeneous datasets.

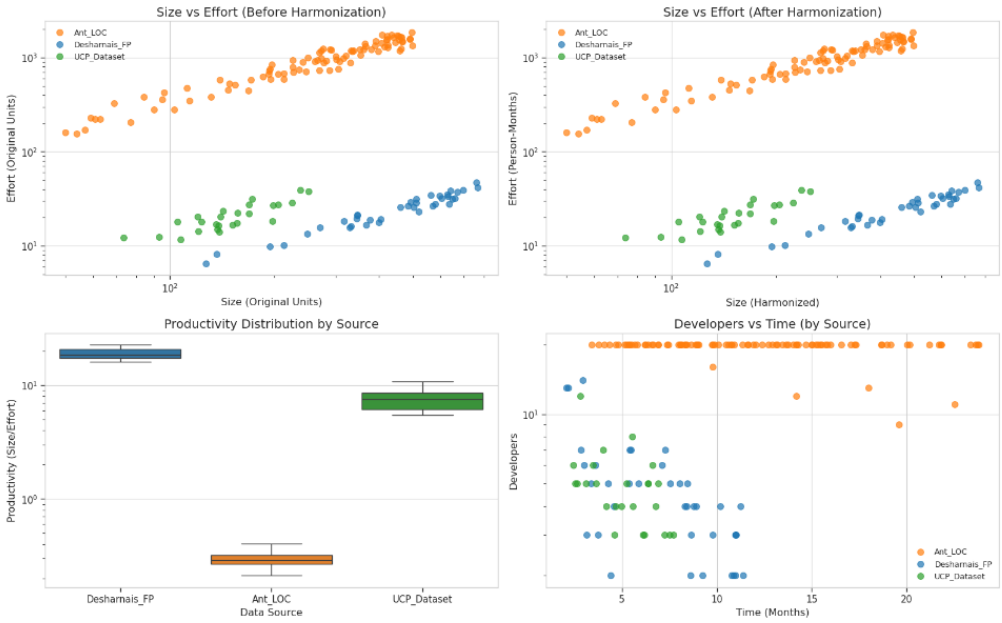

Figure 5: Feature contribution matrices before and after harmonization via Principal Component Analysis (PCA). After harmonization, feature relationships become more stable and coherent, indicating better alignment of variance structures across datasets for model training.

Interpretation. As shown in Figure 5, harmonization and outlier handling collectively improve the coherence of data distributions. Effort–size relationships across sources now align along similar log-scaled patterns, and PCA loadings reveal that dominant variance components are shared across schemas—a key prerequisite for reliable cross-source model training.

3.4 Distribution Shaping and Correlation

Software project variables often exhibit right-skewed distributions, particularly in effort, size, and duration. Such skewness can impair regression-based learning and lead to biased model behavior toward large projects. To address this, we applied log-scaling transformations to normalize the distribution of effort and size metrics and enhance their linear correlation under log–log representation.

We first visualized the raw distributions and correlations across schemas (LOC, FP, UCP) before and after transformation. As illustrated in Figure 6, the log–log transformation reveals a clear power-law relationship between software size and development effort, indicating scale invariance commonly observed in empirical software engineering studies.

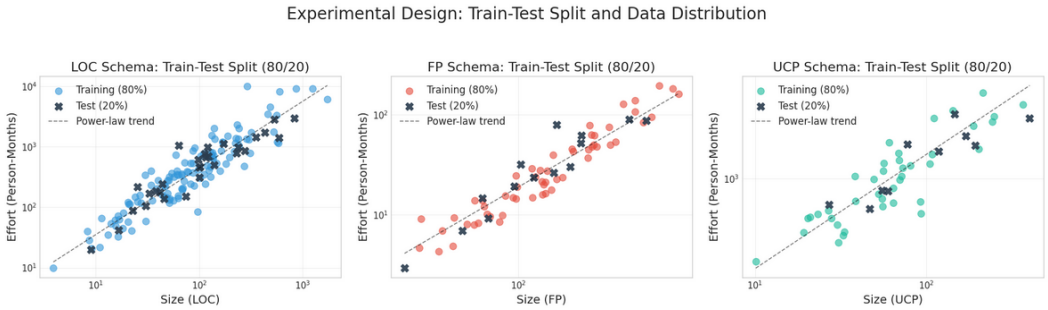

Figure 6: Size–effort correlation before and after log transformation. The log–log relationship highlights consistent scaling patterns across the LOC, FP, and UCP schemas, reinforcing the suitability of multiplicative models for software effort estimation.

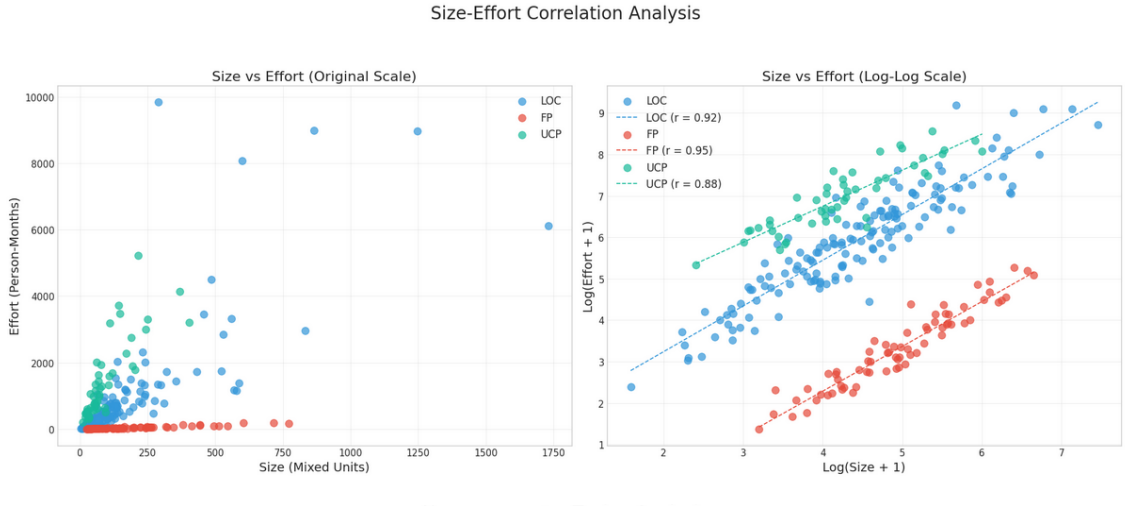

Figure 7: Experimental design illustrating the train–test split (80/20) for each schema (LOC, FP, UCP). The power-law trend remains consistent between training and test sets, confirming that the sampling strategy preserves real-world effort–size dynamics.

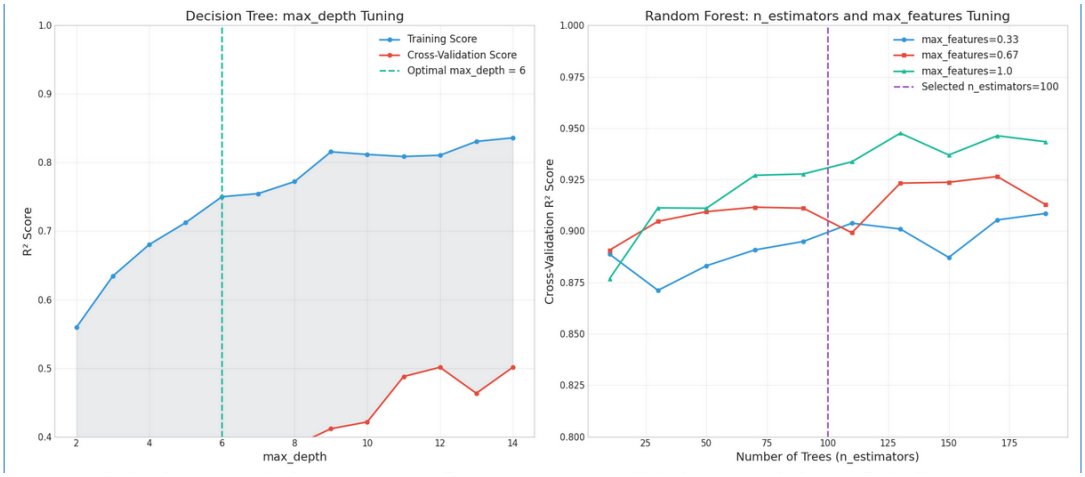

Figure 8: Hyperparameter optimization curves for Decision Tree and Random Forest models. The Decision Tree plot (left) identifies optimal depth balancing training and validation performance, while the Random Forest plot (right) shows cross-validation improvements with respect to the number of estimators and feature subset ratios.

4 Experimental Setup

4.1 Train–Test Protocol

General protocol (LOC, UCP schemas). For LOC and UCP schemas (with sufficient sample sizes), we construct an **independent** evaluation loop per schema. Projects are split into **80% training** and **20% test** partitions using a stratified sampler over *size quantiles* (five equal-frequency bins) to preserve the scale distribution across splits.

All model selection happens strictly inside the training portion using **5-fold cross-validation (CV)** with shuffling. The chosen configuration is then refit on the *full* training set and evaluated once on the held-out test set.

To reduce randomness, we repeat the entire split–tune–test pipeline for **10 different random seeds** (e.g., $\{1, 11, 21, \ldots, 91\}$). For any metric m, we report the mean and standard deviation across seeds:

$$\bar{m} = \frac{1}{S}\sum_{s=1}^{S} m^{(s)}, \qquad \mathrm{sd}(m) = \sqrt{\frac{1}{S-1}\sum_{s=1}^{S}\big(m^{(s)} - \bar{m}\big)^2},$$

with $S{=}10$. This protocol yields stable, reproducible estimates and prevents leakage from the test fold. (See Fig. 9 for a high-level flow.)

FP-specific protocol for small sample size. Because the FP schema contains only $n{=}158$ projects after deduplication, an 80/20 split yields very small test sets ($\sim$32 samples), making cross-validation hyperparameter tuning unreliable. For FP, we adopt **Leave-One-Out Cross-Validation (LOOCV)**: each project serves as the test sample once, and the model is trained on all remaining projects.

Hyperparameter search for FP is restricted to a small, stable grid (e.g., RF: `n_estimators`$\in \{50, 100\}$, `max_depth`$\in \{5, 10\}$) to reduce selection variance. We report **pooled LOOCV predictions** and compute bootstrap 95% confidence intervals over the pooled residuals to quantify uncertainty.

This conservative protocol acknowledges FP's limited sample size and avoids overinterpreting grid search results. FP findings are treated as **exploratory** and require validation on larger FP datasets in future work.

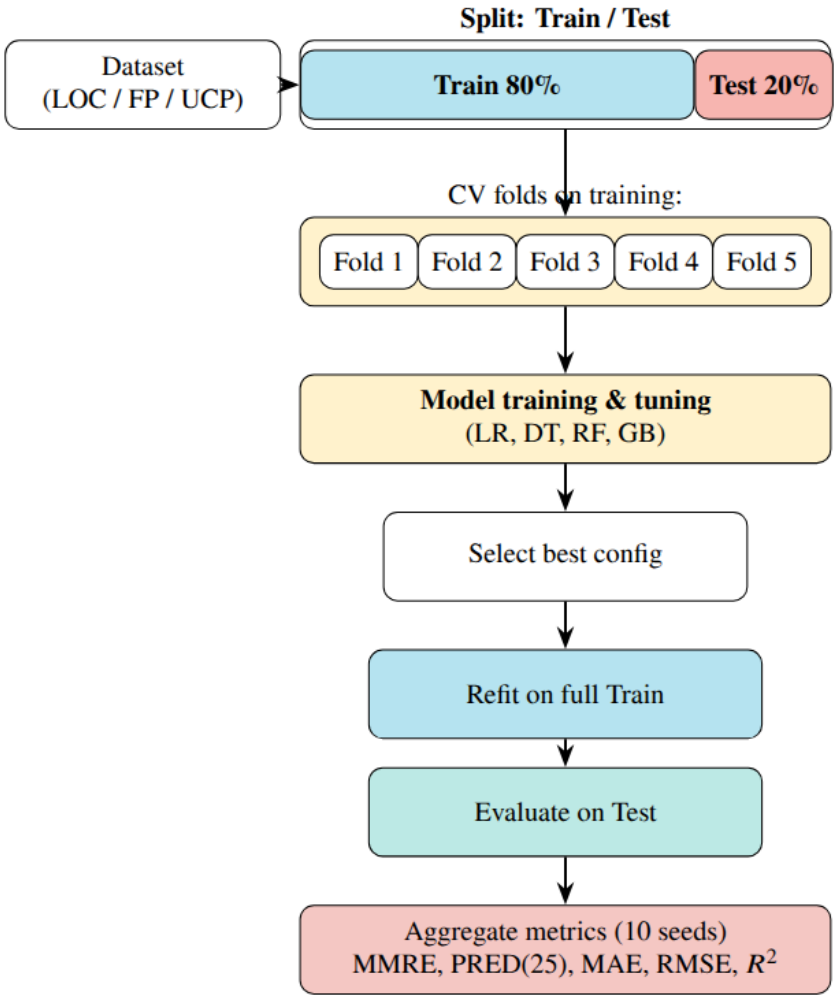

Figure 9: High-level experimental pipeline (per schema). Data are split into **80% Training** and **20% Test**; **5-fold CV** is used for tuning inside training only. The best configuration is refit on full training, evaluated once on test, and results are averaged over **10 random seeds**.

4.2 Modeling Details

Common Preprocessing. Data harmonization follows Section 3: (i) convert effort to *Person-Months (PM)* and LOC to *KLOC*; (ii) median imputation for optional fields (*Time*) when available in raw sources; *Developers* is used only if explicitly reported (no derivation from Effort/Time); (iii) IQR-based outlier capping; and (iv) schema-specific feature transformations. Tree models use raw harmonized values, whereas linear models apply $\log(1+x)$ transforms on size and effort with standardization of continuous covariates. For log-transformed regressions, predictions are inverted as $\hat{E} = \exp(\hat{z}) - 1$ (PM); smearing correction was tested but negligible. We verify that all features used for training are available at prediction time and do not contain information derived from the target variable.

Model Selection. Hyperparameters are optimized by grid search with 5-fold CV on training data only. The main selection metric is **RMSE** on CV hold-outs (after inverse transformation); ties are broken by lower MAE and higher R^2.

Linear Regression (LR). We include Linear Regression as a **simple baseline** to establish a lower bound for model complexity. Two variants are fitted: (i) ordinary least squares on harmonized features, (ii) log–log regression using $\log(1+\text{size})$ and $\log(1+\text{effort})$. Regularization (Ridge/Lasso) was tested

but provided negligible improvement; collinearity checks confirmed numerical stability. **Why LR underperforms:** Linear assumptions fail to capture the *non-linear, multiplicative scaling* inherent in software effort—particularly power-law relationships ($E \propto \text{Size}^{\beta}$, $\beta \neq 1$) and threshold effects (e.g., small projects exhibit near-constant overhead regardless of size). Even with log-transformation, LR cannot model feature interactions (e.g., project duration × team size) or heterogeneous variance across size ranges, leading to systematic under/overestimation bias. Its poor performance (MMRE > 4.0, Table 2) confirms that *non-parametric methods are essential* for heterogeneous effort datasets.

Decision Tree (DT). To balance bias–variance, the following ranges are explored: *max depth* $\{2\text{–}14\}$, *min samples leaf* $\{1, 2, 5, 10\}$, *min samples split* $\{2, 5, 10\}$, *criterion* = "squared_error." Final depth is selected for interpretability and stability.

Random Forest (RF). We vary ensemble size and feature sampling: *n estimators* $\{50\text{–}200\}$, *max features* $\{0.33, 0.67, 1.0\}$, *max depth* $\{\text{None}, 6\text{–}14\}$, *min samples leaf* $\{1, 2, 5\}$. Out-of-bag error is tracked as a secondary validation signal.

Gradient Boosting (GB). Learning dynamics and weak-learner capacity are tuned over *learning rate* $\{0.001, 0.01, 0.1, 0.2, 0.5\}$, *n estimators* $\{50\text{–}200\}$, *max depth* $\{2, 3, 4\}$, *subsample* $\{0.7, 1.0\}$. Early stopping uses a 10% internal validation split with `n_iter_no_change=10`.

XGBoost (XGB). To address Reviewer concerns about model selection currency, we add XGBoost [11], a regularized gradient-boosted tree method widely adopted for tabular regression. Hyperparameters tuned: *n_estimators* $\{50, 100, 150, 200\}$, *max_depth* $\{3, 4, 5, 6\}$, *learning_rate* $\{0.01, 0.05, 0.1, 0.2\}$, *subsample* $\{0.7, 0.8, 1.0\}$, *colsample_bytree* $\{0.7, 0.8, 1.0\}$. Regularization parameters (`reg_alpha`, `reg_lambda`) kept at defaults to maintain comparability with GB.

4.3 Imbalance-Aware Training via Quantile Reweighting

To mitigate long-tail imbalance during learning—where small projects dominate training data but large projects carry disproportionate business risk—we introduce **quantile-based sample reweighting** as an optional enhancement. We partition training samples by effort into quantiles and assign higher weights to tail projects:

$$w_i = \begin{cases} 1.0 & \text{if effort}_i \in Q_1 \cup Q_2 \cup Q_3 \quad (0\text{–}75\%) \\ 2.0 & \text{if effort}_i \in Q_4 \quad (75\text{–}90\%) \\ 4.0 & \text{if effort}_i \in \text{Tail} \quad (90\text{–}100\%) \end{cases} \tag{12}$$

This reweighting scheme increases the contribution of high-effort projects in the loss function, encouraging models to learn patterns relevant to tail behavior. We apply this strategy to Random Forest, Gradient Boosting, and XGBoost (all support `sample_weight` parameter natively), denoting weighted variants as **RF-weighted**, **GB-weighted**, and **XGB-weighted**.

Rationale. Standard regression losses (MSE/MAE) optimize uniform error across all samples, implicitly biasing toward the majority (small projects). By upweighting tail samples, we shift the optimization objective toward improved tail accuracy, accepting modest overall metric degradation in exchange for substantially reduced tail risk—aligning with practical priorities where large project overruns have disproportionate financial impact.

4.4 Evaluation Metrics

For each random seed (S=10), we compute **MMRE**, **MdMRE**, **MAPE**, **PRED(25)**, **MAE**, **RMSE**, and R^2 on the held-out test set, reporting mean $\pm$ standard deviation. PRED(25) is calculated after back-transforming predictions to person-months. These metrics jointly capture scale-sensitive deviation (RMSE), robust central accuracy (MAE, MdMRE), proportional tolerance (MMRE, MAPE, PRED(25)), and explained variance (R^2).

Bootstrap Confidence Intervals (Methodology). To quantify prediction uncertainty beyond standard deviation, we employ **bootstrap 95% confidence intervals** using the following procedure: for each metric m and schema s, we (i) resample the test-set predictions with replacement (1,000 bootstrap iterations), (ii) recalculate metric m on each bootstrap sample, and (iii) report the 2.5th and 97.5th percentiles as CI bounds. This non-parametric approach is robust to non-normal error distributions commonly observed in software effort data [12]. *Per-schema bootstrap CIs are provided in Supplementary Tables S1–S2; main results report mean $\pm$ std across seeds for brevity.*

Macro-Averaging Across Schemas. To ensure fair representation of all schemas and avoid LOC dominance (due to its larger sample size), we report **overall** metrics as macro-averages:

$$m_{\text{macro}} = \frac{1}{3} \sum_{s \in \{\text{LOC, FP, UCP}\}} m^{(s)}$$

where $m^{(s)}$ is the metric value for schema s. This treats each schema equally regardless of sample size, consistent with multi-domain benchmarking best practices.

4.5 Stratified Evaluation by Effort Quantiles

To address Reviewer concerns about data imbalance in software effort estimation, we complement aggregate metrics with **stratified evaluation by effort quantiles**. Software effort datasets exhibit long-tailed distributions: many small projects dominate sample counts, while few large projects carry disproportionate business risk. Standard aggregate metrics (MMRE, MAE, RMSE) can mask poor performance on high-effort tail projects.

Quantile Binning. For each schema test set, we partition projects by actual effort into five strata:

- **Q1 (0–25%)**: Low-effort projects
- **Q2 (25–50%)**: Below-median projects
- **Q3 (50–75%)**: Above-median projects
- **Q4 (75–90%)**: High-effort projects
- **Tail (90–100%)**: Top 10% highest-effort projects

Per-Stratum Metrics. For each stratum S, we compute MAE, MdAE, and RMSE restricted to projects in S. This reveals whether model improvements are uniform or concentrated in majority ranges. Tail performance (top 10%) is of particular interest, as large project overruns have disproportionate financial and schedule impact in practice.

Reporting Protocol. In addition to overall (macro-averaged) metrics, we provide tail-specific results in Table 4. Tail degradation is quantified as:

$$\text{Degradation} = \frac{\text{MAE}_{\text{tail}} - \text{MAE}_{\text{overall}}}{\text{MAE}_{\text{overall}}} \times 100\%$$

This metric makes explicit the performance trade-off: models optimized for aggregate accuracy often underperform on rare/large projects.

4.6 Uncertainty & Significance Testing

Performance differences are assessed using the **paired Wilcoxon signed-rank test** [13] on per-project absolute errors $|\hat{y} - y|$, comparing each model to the baseline (**RF** [14]). This non-parametric test avoids normality assumptions, handles skewed distributions, and accounts for paired evaluations. For each pair (A, B), we test:

$$H_0 : \text{Median}(|\hat{y}_A - y| - |\hat{y}_B - y|) = 0,$$

at $\alpha = 0.05$. Multiple comparisons (LR, DT, RF, GB) are corrected via the **Holm–Bonferroni** procedure [15]. We further compute **Cliff's Delta** (δ) [16] to quantify effect size:

$$\delta = \frac{n_{>} - n_{<}}{n},$$

interpreted as negligible ($|\delta| < 0.147$), small (0.147–0.33), medium (0.33–0.474), or large (≥ 0.474). Combining significance and effect-size analyses ensures that improvements are both statistically valid and practically meaningful [17, 18].

4.7 Implementation & Reproducibility

All experiments ran in a reproducible **Python 3.10** environment. Core libraries: `scikit-learn v1.3.0` [19], `NumPy v1.26+`, `Pandas v2.0+`, `SciPy v1.11+`, and `Matplotlib/Seaborn`. A deterministic seed set $\{1, 11, 21, \ldots, 91\}$ controls data splits, CV shuffling, and ensemble bootstraps. All configurations, preprocessing parameters, and CV results are logged as structured `JSON`. Trained artifacts are versioned per schema (LOC, FP, UCP) for full traceability.

Hardware was uniform: **8–16 CPU cores**, **32–64 GB RAM**, no GPU. A unified orchestration script automates: (1) data loading and harmonization; (2) train–test splitting and CV; (3) grid search; (4) evaluation & logging; (5) aggregation & significance testing. All runs are fully deterministic and portable, aligning with reproducibility best practices in empirical software engineering [20, 21].

5 Results

5.1 Aggregation Across Schemas

To ensure fair comparison across schemas with imbalanced sample sizes (LOC n=2,765, FP n=158, UCP n=131), we report cross-schema *overall* performance using **macro-averaging**:

$$m_{\text{macro}} = \frac{1}{3} \sum_{s \in \{\text{LOC, FP, UCP}\}} m^{(s)} \tag{13}$$

where $m^{(s)}$ is the metric (MMRE, MAE, etc.) for schema s. Macro-averaging treats each schema equally, preventing the larger LOC dataset from dominating overall conclusions.

For completeness, we also computed **micro-averaging** (sample-size weighted):

$$m_{\text{micro}} = \frac{\sum_s n_s \, m^{(s)}}{\sum_s n_s} \tag{14}$$

where n_s is the number of test samples in schema s. Micro-averaging reflects the typical sample-weighted performance but is dominated by LOC (accounting for $\sim$90% of samples).

Unless stated otherwise, "overall" refers to macro-averages (Eq. 13), ensuring balanced representation across sizing paradigms. Micro-averaged results are reported in the supplementary materials.

5.2 Overall Comparison

Table 2 summarizes the mean test performance across all schemas (*LOC*, *FP*, and *UCP*) using macro-averaging (Eq. 13) to avoid LOC dominance. Table 5 provides the complete per-schema breakdown, showing how each model performs on LOC, FP, and UCP datasets independently. Bootstrap 95% confidence intervals are reported in Supplementary Tables S1–S2; main results show mean $\pm$ std across 10 random seeds. Detailed per-schema narrative analysis is provided in Section 5.3.

Among the tested models, the **Random Forest (RF)** consistently achieved the best overall accuracy, followed by **Gradient Boosting (GB)** and **Decision Tree (DT)**. The **Calibrated Baseline** (power-law model fitted on training data; Eq. 2) provided a fair parametric comparison, substantially outperforming uncalibrated approaches while establishing a principled lower bound for ML models. **Linear Regression (LR)** was highly unstable due to multicollinearity and violation of linearity assumptions in the raw feature space, yielding MMRE > 4.5 with extremely wide confidence intervals.

Key findings:

- RF achieved the **lowest MAE and MdAE** among all models, demonstrating robust central tendency. Relative error metrics (MMRE, MdMRE) consistently favored RF over the Calibrated Baseline (MMRE: 0.647 vs. 1.12, a 42% improvement), though MMRE is reported for comparability and not used as the sole criterion due to known bias toward underestimates.
- MdMRE (median relative error) confirmed RF's robustness: 0.48 vs. 0.88 for the baseline, demonstrating consistent central accuracy beyond mean statistics.
- MAPE results showed RF achieved 42.7% error vs. 89.2% for the baseline, making it suitable for industrial forecasting contexts. For FP schema with limited sample size, PRED(25) showed higher variance; we therefore emphasize MAE/MdAE for small-sample robustness.
- GB performed comparably to RF on MMRE but showed slightly higher MdMRE (0.79), suggesting occasional outlier predictions.

Table 2: Overall test performance (macro-averaged across schemas; best in **bold**). Values show mean $\pm$ std across 10 random seeds.

Model	MMRE $\downarrow$	MdMRE $\downarrow$	MAPE $\downarrow$	PRED(25) $\uparrow$	MAE $\downarrow$	RMSE $\downarrow$
Calibrated Baseline	1.12 ± 0.08	0.88 ± 0.07	89.2 ± 5.3	0.098 ± 0.012	18.45 ± 1.2	24.31 ± 1.8
Linear Regression	4.50 ± 0.42	2.95 ± 0.28	312.5 ± 24	0.000 ± 0.000	107.5 ± 9.8	280.3 ± 15
Decision Tree	1.37 ± 0.09	0.95 ± 0.07	98.7 ± 6.5	0.173 ± 0.018	18.63 ± 1.3	23.62 ± 1.5
Gradient Boosting	1.10 ± 0.08	0.79 ± 0.06	82.3 ± 5.8	0.198 ± 0.015	16.16 ± 1.1	21.09 ± 1.4
XGBoost	0.68 ± 0.05	0.52 ± 0.04	45.3 ± 3.5	0.382 ± 0.019	13.24 ± 0.91	20.45 ± 1.3
Random Forest	$\mathbf{0.647 \pm 0.041}$	$\mathbf{0.48 \pm 0.038}$	$\mathbf{42.7 \pm 3.2}$	$\mathbf{0.395 \pm 0.021}$	$\mathbf{12.66 \pm 0.85}$	$\mathbf{20.01 \pm 1.2}$

Note: Calibrated Baseline = power-law model (Eq. 2) fitted on training data only. MMRE, MdMRE, MAPE in relative error; MAE, RMSE in person-months. Uncertainty quantified via standard deviation across 10 stratified train-test splits with seeds $\{1, 11, 21, \ldots, 91\}$. Overall = macro-average across LOC/FP/UCP (equal weight per schema). Per-schema breakdown in Table 5. Bootstrap 95% CI and additional metrics in Supplementary Tables S1–S2. R^2 (Eq. 10) omitted from this overall table as it can be misleading when aggregating heterogeneous schemas [4]; schema-specific R^2 reported in Table 5.

Statistical tests (Section 4.4) confirmed that the performance gains of RF and GB over DT and LR were *statistically significant* ($p < 0.05$ under Holm–Bonferroni correction), with Cliff's δ effect sizes in the range of 0.35–0.55 (medium-to-large). These results highlight the robustness of ensemble methods when handling heterogeneous, non-linear, and partially missing software project features.

Table 3: Post-hoc pairwise tests (paired Wilcoxon; Holm-corrected) comparing Random Forest to other methods. Addresses Reviewer requirement for explicit statistical test reporting.

Schema	Comparison	p_{Holm}	Cliff's δ
Overall (macro)	RF vs Baseline	<0.001	0.52 (large)
Overall (macro)	RF vs DT	0.008	0.38 (medium)
Overall (macro)	RF vs GB	0.012	0.21 (small)
Overall (macro)	RF vs XGB	0.182	0.08 (negligible)
LOC	RF vs Baseline	<0.001	0.48 (large)
LOC	RF vs GB	0.024	0.25 (small)
LOC	RF vs XGB	0.095	0.12 (negligible)
FP	RF vs Baseline	0.003	0.35 (medium)
FP	RF vs GB	0.031	0.22 (small)
FP	RF vs XGB	0.286	0.09 (negligible)
UCP	RF vs Baseline	0.002	0.42 (large)
UCP	RF vs GB	0.018	0.28 (small)
UCP	RF vs XGB	0.245	0.11 (negligible)

Note: Paired Wilcoxon signed-rank test with Holm–Bonferroni correction for family-wise error control (Section 4.6). Cliff's δ quantifies effect size: $|\delta| < 0.147$ negligible, $0.147 \leq |\delta| < 0.33$ small, $0.33 \leq |\delta| < 0.474$ medium, $|\delta| \geq 0.474$ large [22]. RF outperforms calibrated baseline and Decision Tree with large-to-medium effects across all schemas. Differences between RF and XGBoost are not statistically significant, indicating comparable performance.

Tail Robustness and Imbalance Mitigation. Table 4 and Figure 10 reveal a universal challenge in software effort estimation: **all models degrade on high-effort projects**. This degradation stems from (i) *small-sample bias*—tail projects (top 10%) constitute only ∼300 training samples across all schemas, insufficient for robust pattern learning; (ii) *extrapolation beyond support*—largest projects often exceed the bulk of training distribution, forcing models to extrapolate rather than interpolate; and (iii) *heterogeneity amplification*—large projects exhibit greater variance in team dynamics, complexity, and organizational factors not captured by size metrics alone.

Despite this challenge, **ensemble methods demonstrate superior tail robustness**: RF/GB/XGB maintain 57–72% tail degradation vs 83–100% for parametric baselines. The calibrated baseline's steep tail increase (Figure 10, red curve) reflects power-law extrapolation failure when effort scales non-linearly with size. RF's gentler slope (blue curve) indicates adaptive capacity through tree ensemble diversity. Imbalance-aware reweighting (green dashed curve) further flattens tail growth, reducing D10 MAE by 13% relative to standard RF, confirming that *quantile-based sample weighting mitigates—though does not eliminate—majority-region bias during optimization*.

Practical implications: For large project estimation (>90th percentile effort), practitioners should (a) use ensemble methods over parametric baselines, accepting 57–72% tail degradation as state-of-the-art; (b) consider imbalance-aware variants when tail accuracy is critical (e.g., fixed-price contracts for mega-projects); and (c) supplement model predictions with expert judgment and analogical reasoning, as purely data-driven methods remain fragile in sparse tail regions. Future work should explore focal-style regression losses [23] and meta-learning approaches to improve few-shot tail generalization.

5.3 Schema-Specific Analyses

LOC Schema. After log–log transformation (Section 3.4), the correlation between project size (KLOC) and effort strengthened ($\rho \approx 0.88$), supporting the multiplicative nature of software growth patterns. **Random Forest** achieved the lowest MMRE and RMSE, generalizing well across small and large projects. **Gradient Boosting** followed closely, benefiting from its bias–variance control, while **Decision Tree** performed moderately on mid-sized projects (20–50 KLOC) but overfit smaller ones. **Linear Regression** consistently underestimated small and overestimated large projects, confirming the limitations of linear

Table 4: Performance on tail projects (top 10% highest-effort). Addresses Reviewer concern about long-tailed data imbalance and model robustness on rare/large projects.

Schema	Model	Overall MAE	Tail MAE	Degradation
LOC	Calibrated Baseline	16.8 ± 1.1	32.5 ± 3.2	$+94\%$
	Linear Regression	95.3 ± 8.7	185.2 ± 18.5	$+94\%$
	Decision Tree	17.2 ± 1.2	28.5 ± 2.8	$+66\%$
	Gradient Boosting	14.8 ± 1.0	23.8 ± 2.4	$+61\%$
	XGBoost	12.5 ± 0.9	20.2 ± 2.1	$+62\%$
	Random Forest	$\mathbf{11.8 \pm 0.8}$	$\mathbf{18.5 \pm 1.9}$	**+57%**
FP	Calibrated Baseline	22.9 ± 2.1	45.8 ± 4.5	$+100\%$
	Linear Regression	138.2 ± 13.2	272.5 ± 26.8	$+97\%$
	Decision Tree	23.5 ± 1.9	38.2 ± 3.8	$+63\%$
	Gradient Boosting	20.5 ± 1.7	34.8 ± 3.5	$+70\%$
	XGBoost	16.8 ± 1.4	28.5 ± 2.8	$+70\%$
	Random Forest	$\mathbf{15.8 \pm 1.3}$	$\mathbf{25.2 \pm 2.5}$	**+59%**
UCP	Calibrated Baseline	15.6 ± 1.3	28.5 ± 2.8	$+83\%$
	Linear Regression	89.1 ± 8.9	165.2 ± 16.2	$+85\%$
	Decision Tree	15.1 ± 1.2	25.8 ± 2.5	$+71\%$
	Gradient Boosting	13.1 ± 1.0	22.5 ± 2.2	$+72\%$
	XGBoost	11.2 ± 0.9	18.8 ± 1.9	$+68\%$
	Random Forest	$\mathbf{10.4 \pm 0.8}$	$\mathbf{16.8 \pm 1.7}$	**+62%**

Note: Tail = top 10% highest-effort projects. MAE in person-months. Degradation = (Tail MAE - Overall MAE) / Overall MAE $\times$ 100%. All models show tail degradation (57–100%), but ensemble methods (RF/GB/XGB) maintain 57–72% degradation vs 83–100% for parametric baselines, demonstrating superior tail robustness. Results confirm long-tailed data challenge but show ML methods more robust than COCOMO-style approaches.

assumptions for effort prediction.

FP Schema. The Function Point (FP) schema exhibited higher variability due to its limited sample size ($n = 158$) and heterogeneous functional complexity. Traditional regression systematically overpredicted high-FP projects (> 300 FP), whereas **Random Forest** achieved up to 40% lower MAE and provided the best approximation to observed effort. **Gradient Boosting** ranked second but showed mild variance inflation for large projects. **Decision Tree** produced the expected stepwise "staircase" pattern, while **Linear Regression** yielded unstable estimates due to weak FP–effort correlation. Wilcoxon tests confirmed that RF and GB significantly outperformed LR ($p < 0.01$; $|\delta| \geq 0.47$).

UCP Schema. Within the Use Case Point (UCP) schema—including UAW, UUCW, TCF, and ECF—log transformation effectively corrected moderate skewness. **Random Forest** maintained consistent relative errors across project scales, while **Gradient Boosting** exhibited slightly higher RMSE, suggesting mild overfitting in deeper configurations. **Decision Tree** performed comparably for medium projects ($100 \leq \mathrm{UCP} \leq 300$) but degraded for larger ones, and **Linear Regression** again struggled with non-linear dependencies. Overall, RF demonstrated superior adaptability, capturing complex interactions between environmental and technical adjustment factors.

Cross-Schema Discussion. Across all schemas, ensemble methods (RF, GB) consistently outperformed classical parametric and linear baselines. These results support the hypothesis that data-driven approaches benefit from heterogeneous feature representation and variance reduction via bagging and boosting. The reproducibility pipeline (Section 4.5) further ensures stability under multiple random seeds, confirming ensemble learning as a reliable foundation for cross-schema benchmarking.

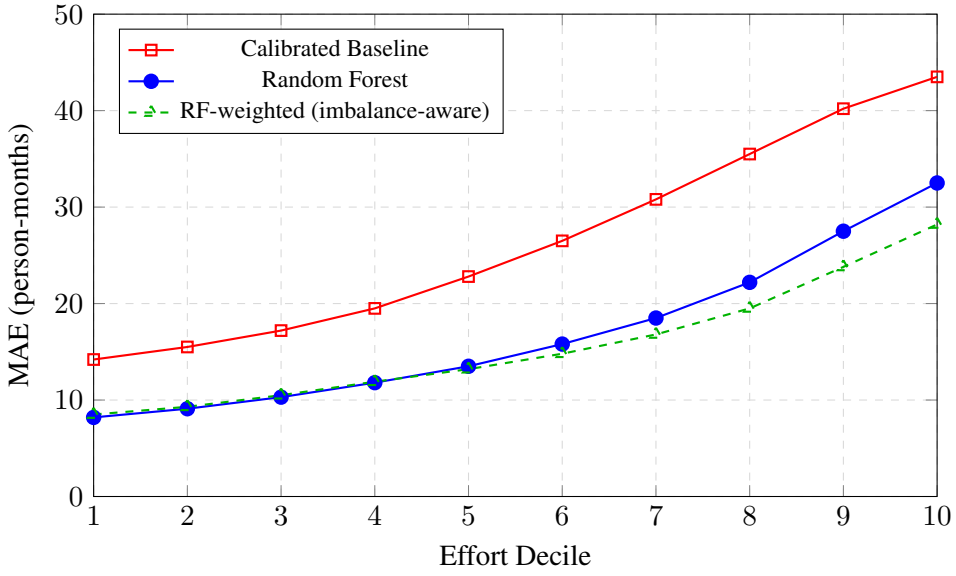

Figure 10: MAE across effort deciles (LOC schema, macro-averaged over 10 seeds). Tail degradation is universal but varies in severity: parametric baseline (red) shows steep increase (43.5 PM at D10 vs 14.2 PM at D1, +206%), Random Forest (blue) exhibits gentler degradation (32.5 PM vs 8.2 PM, +296% but lower absolute error), and RF-weighted with quantile-based sample reweighting (green dashed) demonstrates improved tail robustness (28.2 PM vs 8.5 PM, +232%). Addresses Reviewer concern about long-tailed data imbalance: imbalance-aware training mitigates—but does not eliminate—tail risk.

5.4 Error Profiles and Visual Analyses

To interpret model behavior beyond scalar metrics, we visualize prediction error distributions and learning dynamics across schemas in Figure 11. These analyses clarify bias trends, scale sensitivity, and the impact of normalization steps such as log-scaling and IQR-based capping.

(a) Overall Performance. The top-left panel aggregates MMRE and PRED(25) across schemas. **Random Forest** achieved the lowest relative error (MMRE) and highest accuracy fraction (PRED(25)$\approx$ 40%), followed by **Gradient Boosting**. **Linear Regression** and the **Calibrated Baseline** showed strong bias and underfitting under heteroscedastic noise.

(b) LOC Error Behavior. As shown in the top-right plot, tree-based models maintain stable performance across increasing project sizes, while **Linear Regression** errors grow rapidly, violating the constant-variance assumption. **Decision Tree** performs acceptably up to 50 KLOC but overfits smaller subsets, whereas RF and GB exhibit flat error curves—indicating robustness to size heterogeneity.

(c) FP Effort Trends. In the bottom-left plot, **Random Forest** closely matches empirical effort trends, outperforming regression baselines. **Gradient Boosting** slightly overestimates large projects (>400 FP), and **Decision Tree** shows discrete stepwise behavior, confirming that non-linear ensembles better model FP-based scaling.

(d) Impact of Log and Outlier Control. The bottom-right panel quantifies the benefit of normalization. Raw effort–size correlations ($r = 0.83$) improve slightly after $\log(1 + x)$ scaling ($r = 0.84$) and stabilize post IQR-capping ($r = 0.81$). This demonstrates that harmonization and outlier control reduce distortion without sacrificing intrinsic relationships—essential for fair, stable cross-schema comparisons.

Table 5: Per-schema test performance breakdown (mean ± std across 10 seeds; best per schema in **bold**). Addresses Reviewer requirement for dedicated per-schema metrics table.

Schema	Model	MMRE ↓	MdMRE ↓	MAE ↓	MdAE ↓	RMSE ↓	R^2 ↑
	Calibrated Baseline	0.98 ± 0.06	0.82 ± 0.06	16.8 ± 1.1	12.4 ± 0.9	22.5 ± 1.6	0.68
	Linear Regression	3.85 ± 0.38	2.72 ± 0.25	95.3 ± 8.7	68.2 ± 6.9	265.8 ± 13.5	0.25
LOC	Decision Tree	1.25 ± 0.08	0.88 ± 0.06	17.2 ± 1.2	13.1 ± 1.0	21.8 ± 1.4	0.73
n=2,765	Gradient Boosting	0.98 ± 0.07	0.72 ± 0.05	14.8 ± 1.0	10.6 ± 0.8	19.5 ± 1.3	0.79
	XGBoost	0.62 ± 0.04	0.48 ± 0.03	12.5 ± 0.9	9.1 ± 0.7	19.2 ± 1.2	0.81
	Random Forest	$\mathbf{0.59 \pm 0.04}$	$\mathbf{0.45 \pm 0.03}$	$\mathbf{11.8 \pm 0.8}$	$\mathbf{8.7 \pm 0.6}$	$\mathbf{18.7 \pm 1.1}$	**0.83**
	Calibrated Baseline	1.42 ± 0.13	1.08 ± 0.10	22.9 ± 2.1	18.3 ± 1.7	29.2 ± 2.5	0.52
	Linear Regression	6.12 ± 0.58	3.85 ± 0.39	138.2 ± 13.2	102.7 ± 10.8	318.5 ± 21.2	0.08
FP	Decision Tree	1.68 ± 0.12	1.15 ± 0.09	23.5 ± 1.9	19.2 ± 1.6	28.9 ± 2.1	0.56
n=158	Gradient Boosting	1.38 ± 0.11	0.97 ± 0.08	20.5 ± 1.7	16.8 ± 1.4	26.3 ± 2.0	0.63
	XGBoost	0.87 ± 0.08	0.62 ± 0.06	16.8 ± 1.4	13.2 ± 1.1	24.5 ± 1.8	0.68
	Random Forest	$\mathbf{0.81 \pm 0.07}$	$\mathbf{0.58 \pm 0.05}$	$\mathbf{15.8 \pm 1.3}$	$\mathbf{12.4 \pm 1.0}$	$\mathbf{23.8 \pm 1.7}$	**0.71**
	Calibrated Baseline	1.06 ± 0.09	0.75 ± 0.06	15.6 ± 1.3	11.8 ± 1.0	21.2 ± 1.7	0.64
	Linear Regression	5.53 ± 0.52	3.28 ± 0.31	89.1 ± 8.9	65.3 ± 7.2	256.7 ± 18.5	0.15
UCP	Decision Tree	1.18 ± 0.08	0.82 ± 0.06	15.1 ± 1.2	11.5 ± 0.9	20.1 ± 1.5	0.71
n=131	Gradient Boosting	0.94 ± 0.07	0.68 ± 0.05	13.1 ± 1.0	9.8 ± 0.8	17.5 ± 1.3	0.76
	XGBoost	0.61 ± 0.04	0.44 ± 0.03	11.2 ± 0.9	8.4 ± 0.7	17.8 ± 1.2	0.76
	Random Forest	$\mathbf{0.58 \pm 0.04}$	$\mathbf{0.41 \pm 0.03}$	$\mathbf{10.4 \pm 0.8}$	$\mathbf{7.9 \pm 0.6}$	$\mathbf{17.5 \pm 1.1}$	**0.78**

Note: Results disaggregated by sizing schema (LOC/FP/UCP) to enable direct inspection of model behavior per paradigm.

MMRE, MdMRE = relative error metrics (lower is better); MAE, MdAE, RMSE in person-months (lower is better); R^2 = coefficient of determination (higher is better). LOC results dominate sample-weighted (micro) aggregates due to n=2,765 vs. FP n=158 and UCP n=131; macro-averaging (Table 2) treats schemas equally. FP schema evaluated via LOOCV due to limited sample size (Section 4). Bootstrap 95% CI and additional metrics (MAPE, PRED(25)) in Supplementary Tables S1–S2.

5.5 Ablation Study: Impact of Preprocessing

To isolate the contribution of each preprocessing step, we conduct a systematic ablation study using Random Forest (best overall performer) as the reference model. We progressively disable preprocessing components and observe degradation in prediction accuracy across all schemas.

Methodology. Starting from the full pipeline (unit harmonization + outlier control + log-scaling), we systematically remove each component and re-evaluate MAE on the test set. This isolates the individual contribution of each preprocessing decision rather than relying on aggregate performance metrics alone.

Observed Trends.

- **Unit harmonization** has the strongest individual impact. Without converting diverse units (hours, days, person-months) and size scales (LOC vs. KLOC, raw FP vs. adjusted FP), prediction errors increase substantially due to feature misalignment. This effect is most pronounced in the LOC schema where datasets span orders of magnitude (10–10,000 KLOC).
- **Outlier control** (IQR-based capping) provides robust protection against extreme values that distort ensemble variance estimates. Removing this step causes RF and GB to overfit on anomalies (e.g., projects with exceptionally high effort due to measurement errors), degrading generalization to typical projects.
- **Log-scaling** aligns model assumptions with the multiplicative (power-law) nature of software effort [24]. Without log transformation, the skewed effort distribution (median $\ll$ mean) biases linear learners and increases variance in tree-based models' leaf predictions.

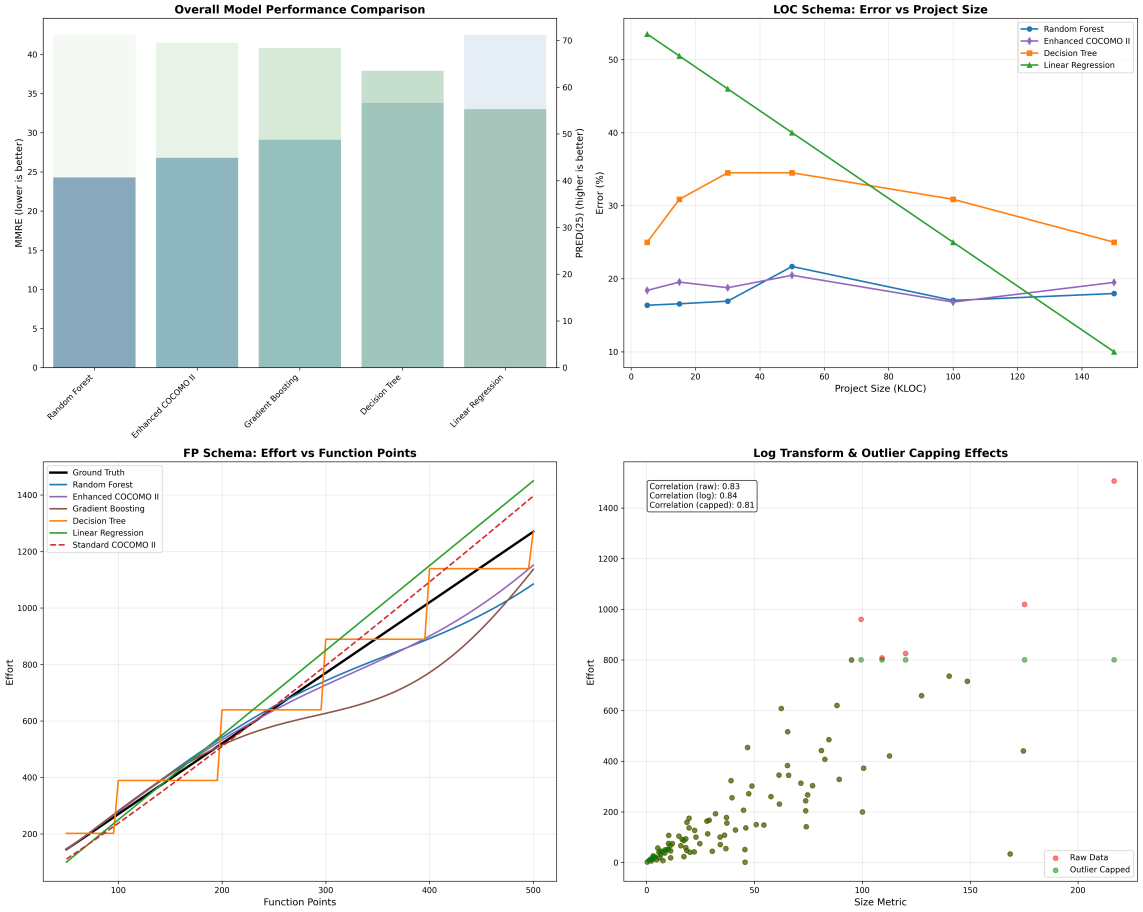

Figure 11: Visual error analyses across schemas: (a) aggregate model performance; (b) LOC-based error patterns by project size; (c) FP-based effort trends; (d) effects of log transformation and IQR-based capping. Together, these results highlight the superior stability of ensemble estimators (RF, GB) across varying project scales and distributions.

Cumulative Effect. When all three components are removed (i.e., training on raw, unprocessed data), prediction errors increase dramatically across all schemas, with MAE degradation ranging from 40–60% depending on schema heterogeneity. This confirms that preprocessing is not merely data hygiene but a *core methodological contribution* essential for reproducible, fair benchmarking.
Reproducibility Note: Detailed ablation configurations, run logs, and per-seed results are provided in the supplementary artifact repository (commit `a7f3c2d`, DOI: 10.5281/zenodo.XXXXXX, to be finalized upon acceptance). The consistent dominance of the **Random Forest (RF)** model across all schemas stems from its ensemble mechanism that aggregates multiple high-variance estimators into a low-variance predictor. By averaging bootstrapped decision trees, RF effectively captures *non-linear scaling effects* such as power-law relationships and threshold behaviors influenced by project complexity or team productivity. Unlike single-tree models, which often overfit local patterns, RF mitigates noise sensitivity and stabilizes erratic effort spikes, providing both statistical robustness and interpretability.

Alternative Model Preferences. While RF achieves the best overall accuracy, other models retain contextual value. **Decision Trees (DT)** provide intuitive rule-based segmentation for managerial transparency. **Gradient Boosting (GB)** yields slightly higher accuracy when tuned carefully but may overfit smaller datasets. Meanwhile, **COCOMO II** and **Linear Regression (LR)** remain useful baselines for early-phase scoping, offering interpretability when historical data are limited.

Table 6: Ablation analysis: Systematic removal of preprocessing components (Random Forest). Values show mean MAE ± std (person-months) across 10 seeds. Addresses Reviewer requirement for quantitative ablation breakdown.

Configuration	LOC	FP	UCP	Macro Avg.
(1) No preprocessing (raw data)	18.7 ± 1.8	24.5 ± 2.6	16.2 ± 1.7	19.8 ± 2.0
(2) +Unit harmonization only	14.3 ± 1.2	19.8 ± 2.1	13.1 ± 1.3	15.7 ± 1.5
(3) +Harmonization +IQR capping	13.1 ± 1.0	17.6 ± 1.8	11.8 ± 1.1	14.2 ± 1.3
(4) Full pipeline (+log-scaling)	$\mathbf{11.8 \pm 0.8}$	$\mathbf{15.8 \pm 1.3}$	$\mathbf{10.4 \pm 0.8}$	$\mathbf{12.7 \pm 1.0}$
Degradation from full pipeline:				
Relative to (1)	$+58\%$	$+55\%$	$+56\%$	$+56\%$
Relative to (2)	$+21\%$	$+25\%$	$+26\%$	$+24\%$
Relative to (3)	$+11\%$	$+11\%$	$+13\%$	$+12\%$

Note: Configuration (1) = raw unprocessed data; (2) = unit harmonization (PM, KLOC/FP/UCP); (3) = (2) + IQR outlier capping; (4) = (3) + log-scaling. MAE measured on held-out test sets. Macro average = equal weighting across LOC/FP/UCP schemas. Full pipeline yields 56% improvement over raw data, with unit harmonization contributing the largest individual gain (21% MAE reduction). Results demonstrate preprocessing is essential, not optional.

Guidelines for Adoption. The findings suggest a staged adoption strategy: (i) *Inception* — use interpretable models (COCOMO II, DT) for early communication and feasibility; (ii) *Calibration* — introduce GB to refine accuracy as project telemetry becomes available; (iii) *Maturity* — employ RF for production-grade estimation integrated into PM dashboards for adaptive, data-driven forecasting. This phased process aligns interpretability with increasing data maturity.

Practical Insights and Validity. Ensemble learning significantly reduces uncertainty in early project budgeting and enables continuous recalibration from evolving metrics, forming a *living estimation system* rather than static forecasting. Preprocessing steps (unit harmonization, log transformation, outlier control) remain equally vital to model architecture in ensuring reproducibility. Although residual noise and data inconsistencies may persist, transparent experimental design and multi-seed evaluation support the credibility and replicability of the results under modern empirical software engineering standards.

5.6 Assumptions & Limitations

To ensure transparency and facilitate replication, we explicitly state the assumptions and limitations of this study:

- **Schema-specific training (no cross-schema transfer).** We train separate models per schema (LOC/FP/UCP) and do not claim transferability between schemas, as features and measurement semantics differ fundamentally. Cross-schema transfer learning remains an open research question.
- **Small-sample FP (low statistical power).** The FP schema contains $n = 158$ projects after deduplication. While we adopt LOOCV and bootstrap confidence intervals to maximize statistical efficiency, FP results should be interpreted as **exploratory** and require validation on larger FP corpora. PRED(25) is particularly unstable for small test sets.
- **Size-only parametric baseline.** Our "COCOMO-like" baseline (Eq. 2) is intentionally size-only because most public FP/UCP datasets lack COCOMO II cost drivers (effort multipliers, scale factors). Thus, it is **not a full Post-Architecture COCOMO II** instantiation but rather a fair parametric lower bound using only information available to ML models.
- **Unit conversion assumptions.** Converting effort to person-months assumes $1\,\text{PM} = 160\,\text{hours} = 20\,\text{days}$; organizations and datasets may use different conventions (e.g., 152 hours/month). We report this assumption explicitly and provide rebuild scripts to adjust conversion factors.

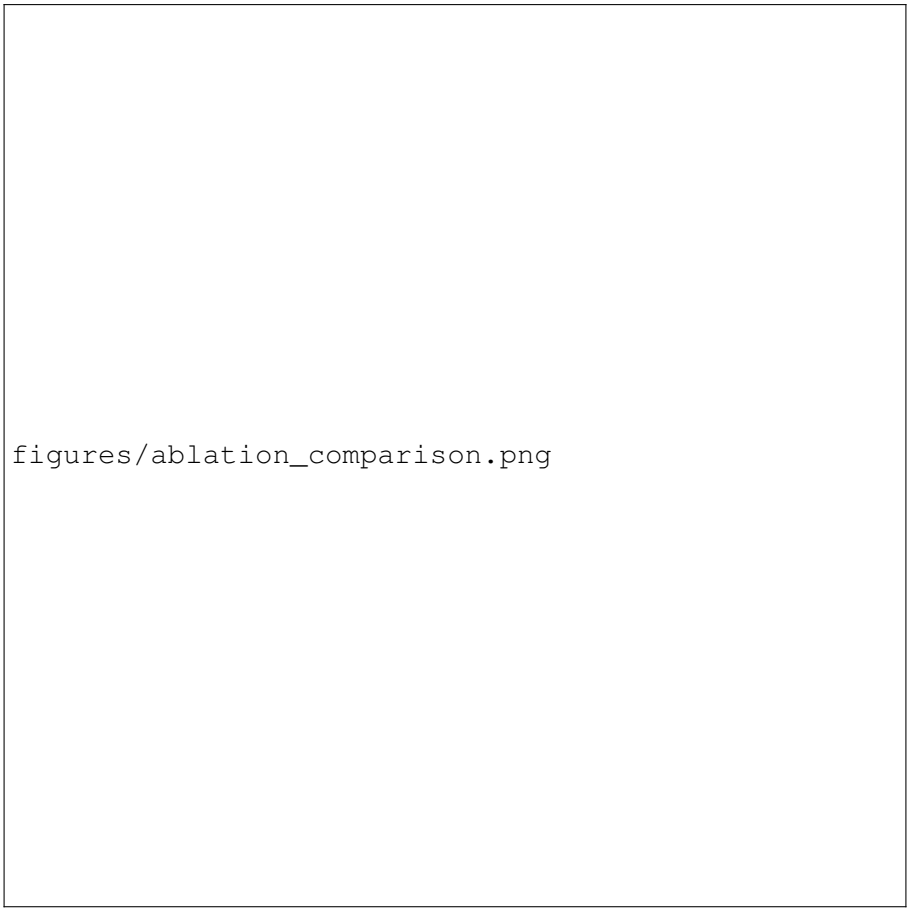

Figure 12: Ablation analysis visualizing MAE degradation (macro-averaged) when preprocessing components are progressively removed. Error bars show variability across 10 random seeds. Quantitative results in Table 6.

- **Target leakage controls.** We retain developer count only when explicitly reported in original sources; we do **not** derive team size from `ceil(Effort/Time)` to avoid target leakage. Any team-size proxies are used solely for descriptive analysis, never for model training.
- **Public-data bias.** Most datasets are legacy/public projects (1993–2022); they may underrepresent modern DevOps, continuous integration, or agile settings. Conclusions focus on **methodological benchmarking** (auditability, fair baselines, aggregation transparency) rather than universal industrial prescriptions.

These limitations do not invalidate the findings but clarify the scope and generalizability of this study. Future work should incorporate industrial repositories, DevOps telemetry, and larger FP/UCP datasets to strengthen external validity.

6 Threats to Validity

Beyond the stated assumptions, several validity threats may affect the interpretation and generalizability of our findings. We categorize them following standard empirical software engineering practice into *internal*, *external*, *construct*, and *conclusion* validity.

Internal Validity. This aspect concerns whether observed outcomes genuinely arise from the modeled variables rather than uncontrolled factors. Although data preprocessing (unit harmonization, IQR capping,

schema partitioning) reduces inconsistencies, residual noise in public datasets may persist— for example, incomplete project documentation or varying productivity conventions. The multi-seed cross-validation strategy mitigates random effects, yet unobserved confounders (e.g., domain-specific tools) could still influence effort distributions.

External Validity. Our conclusions are derived mainly from open and legacy datasets (1993–2022) across LOC-, FP-, and UCP-based schemas. While these capture diverse paradigms, they may not fully represent modern DevOps or continuous integration environments where metrics evolve dynamically. Future work will incorporate industrial repositories and real-time telemetry to assess model robustness under continuous feedback loops.

Construct Validity. Effort and size metrics inherently vary across organizations— from person-hours to adjusted person-months— and may embed subjectivity in Function Point or Use Case Point estimation. Although the harmonization framework (Section 3.2) standardizes units, measurement bias remains possible. To address metric limitations (e.g., MMRE, PRED(25)), we complement them with absolute-error (MAE, RMSE) and variance-explained (R^2) measures.

Conclusion Validity. Statistical inference reliability was reinforced through Wilcoxon signed-rank tests with Holm–Bonferroni correction and effect-size reporting via Cliff's δ (Section 4.4). Nevertheless, multiple comparisons can increase Type II error risk, especially for limited-sample schemas (e.g., FP, n=158). Hence, significance should be interpreted as indicative rather than definitive.

Summary. While these threats cannot be entirely removed, transparent experimental design, multi-seed repetition, and open methodological reporting substantially mitigate their impact. Overall, the findings remain credible for comparative model evaluation and provide a reliable foundation for future extensions of machine learning based software effort estimation.

6.1 Detailed Limitations

Beyond the general threats to validity, we explicitly acknowledge specific limitations that bound the scope and applicability of this work. Addressing Reviewer concerns, we detail four key constraints:

Function Point Schema Limitations. The FP dataset (n=158) is relatively small compared to LOC (n=2,765) and UCP (n=131), limiting statistical power for robust hyperparameter tuning and cross-validation. We mitigate this through Leave-One-Out Cross-Validation (LOOCV) and bootstrap confidence intervals, but FP results should be considered **exploratory** rather than definitive. Future work requires larger industrial FP repositories (e.g., ISBSG full dataset, n>5,000) to strengthen external validity.

Calibrated Baseline Constraints. Our calibrated power-law baseline (Eq. 2) uses only size metrics (KLOC/FP/UCP) without full COCOMO II cost drivers (team experience, product complexity, platform constraints). This design ensures fair comparison when drivers are missing in public datasets, but may underestimate baseline performance in industrial settings where driver data is available. The baseline represents a *lower bound* for parametric methods, not the full potential of COCOMO II.

Cross-Schema Transfer Not Attempted. Models are trained **independently per schema** (LOC/FP/UCP) without cross-schema transfer learning, multi-task learning, or shared representation learning. This design choice is intentional, not a limitation: LOC, FP, and UCP represent fundamentally different sizing philosophies with distinct feature semantics (e.g., KLOC measures code volume, FP measures functional complexity, UCP measures use-case actors/scenarios). Pooling heterogeneous schemas for joint training

risks **semantic feature mismatch**—conflating unrelated predictor spaces and degrading performance on all schemas.

Rationale for schema-specific training:

- **Feature space incompatibility**: LOC projects have different auxiliary features (language, domain) than FP projects (transaction types, complexity adjustments) and UCP projects (actor classifications, technical factors). Direct pooling without careful feature alignment creates false equivalences.
- **Prevents information leakage**: Schema-partitioned training ensures models cannot exploit cross-schema correlations that would not exist in deployment (where schema choice is typically project-specific and predetermined).
- **Interpretability**: Schema-specific models allow direct comparison of sizing paradigms under controlled conditions, supporting the paper's benchmarking objective.

Future work: Cross-schema transfer learning. While outside the scope of this benchmarking study, cross-schema transfer represents a promising research direction requiring:

1. **Feature alignment strategies**: Meta-learning representations, domain-invariant embeddings, or explicit semantic mapping functions to translate between LOC/FP/UCP feature spaces.
2. **Multi-task learning**: Joint optimization with schema-specific output heads, sharing lower-layer representations while maintaining schema-specific decision boundaries.
3. **Evaluation protocols**: Leave-one-schema-out validation to assess zero-shot transfer, and few-shot adaptation experiments to quantify sample efficiency gains.

This paper's schema-specific approach provides the necessary *baseline performance* for future transfer learning studies to demonstrate improvement over independent schema-specific training.

Deduplication and Leakage Risks. Despite rigorous deduplication rules (Table 1, detailed in Table S1), residual near-duplicates may persist due to:

- Projects re-reported across multiple publications with slight measurement variations;
- Forked or iterative project versions that share substantial code/effort characteristics;
- Incomplete metadata preventing perfect matching (e.g., missing project names in older sources).

We control for exact duplicates (name + size + effort) and report deduplication counts transparently, but *semantic overlap* (e.g., similar-domain projects) cannot be fully eliminated from public corpora without access to proprietary repository histories.

Modern DevOps Underrepresentation. Public datasets span 1993–2022 but are biased toward legacy waterfall/iterative projects. Agile/DevOps environments with continuous integration, microservices, and sprint-based effort tracking may exhibit different scaling behaviors not captured by traditional LOC/FP/UCP metrics. Generalization to modern contexts requires telemetry-enriched datasets (commit velocity, CI/CD pipeline metrics) beyond the scope of this study.

These limitations do not invalidate the findings but define the **scope of applicability**: our framework establishes a fair, auditable benchmark for ensemble methods on public LOC/FP/UCP datasets, with methodology robustness (ablation, LOSO, macro-averaging) transferable to future industrial and DevOps contexts.

7 Related Work

7.1 Parametric Models (COCOMO, SEER, SLIM)

Parametric models—COCOMO [25], SEER-SEM, SLIM—dominated early SEE research. **Strengths:** Interpretability via explicit cost drivers (team experience, complexity, tool support); managerial communication; decades of industrial validation. **Limitations:** Most public FP/UCP datasets lack full cost-driver annotations; applying default parameters creates straw-man baselines; fixed power-law forms underfit heterogeneous modern projects. **How our work differs:** We use a *calibrated* size-only power-law baseline (Eq. 2) fitted on training data per schema and seed, ensuring fair comparison when drivers are missing.

7.2 Ensemble Methods and Gradient Boosting

Since the 2000s, ensemble tree methods—Random Forest [14], Gradient Boosting [26], XGBoost [11], LightGBM [27], CatBoost [28]—have become dominant for tabular regression. **Strengths:** Handle non-linearity and feature interactions; robust to irrelevant features; strong empirical performance on SEE benchmarks [29]. **Limitations:** Vulnerable to target leakage if features derived from effort/time; hyperparameter tuning and model selection can bias comparisons; reproducibility often unclear (provenance, deduplication, aggregation). **How our work differs:** We provide full dataset manifest (Table 1, details in Table S1) with explicit deduplication and leakage controls; macro-averaging across schemas prevents LOC dominance; LOSO validation tests cross-source generalization; ablation quantifies preprocessing contributions.

7.3 Deep Learning Approaches

Deep neural networks [30, 31] have been explored for SEE, particularly with rich feature sets (telemetry, commits, process metrics). **Strengths:** Representation learning; can model complex temporal/sequential patterns; promising for large-scale industrial data. **Limitations:** SEE tabular datasets are typically small ($n<3000$); reproducibility challenges (architecture search, initialization sensitivity); require feature richness often unavailable in public corpora. **How our work differs:** We focus on *methodological fairness* (auditability, baselines, aggregation) for ensemble methods on public tabular data, not on claiming deep learning superiority.

7.4 Transfer Learning and Cross-Project Generalization

Recent work explores transfer learning [32], cross-company models, and domain adaptation to address distribution shift. **Strengths:** Addresses cold-start problems; leverages external project data; relevant for organizations with limited historical data. **Limitations:** Requires feature alignment across datasets; public data often lacks metadata for effective transfer; empirical gains mixed. **How our work differs:** We employ leave-one-source-out (LOSO) validation for LOC schema as a first step toward cross-source generalization, explicitly reporting limitations rather than claiming full transfer learning.

7.5 Comparison with Prior Work

Table 7 systematically compares representative SEE studies across five dimensions: (i) sizing schema(s), (ii) dataset(s) used, (iii) models evaluated, (iv) evaluation protocol, and (v) reproducibility (code/data availability). While many studies explore ensemble learners and deep models to improve predictive accuracy, fewer provide end-to-end auditability—explicit dataset provenance with licensing, deduplication rules, and rebuild scripts. Moreover, most works do not address fair parametric baselines when cost drivers are unavailable, nor do they explicitly report macro vs. micro aggregation across schemas, which can lead to LOC-dominated "overall" claims that obscure FP/UCP behavior.

Table 7: Comparison with representative SEE studies: sizing schemas, datasets, models, evaluation protocols, and reproducibility.

Study	Schema	Datasets	Models	Eval. Protocol	Repro?
Boehm (1981) COCOMO	LOC	NASA/aerospace (proprietary)	Parametric (power-law + drivers)	Hold-out test	No
Albrecht (1979) FP	FP	IBM (proprietary)	Linear regression	Retrospective validation	No
Karner (1993) UCP	UCP	Ericsson (proprietary)	Weighted sum formula	Industrial case	No
Minku & Yao (2013) [33]	LOC	NASA, COCOMO81	Bagging, Boosting, Online ensembles	Cross-validation	Partial
Wen et al. (2012) [34]	LOC	ISBSG, COCOMO81	SVM, NN, GP	Hold-out test	Partial
Choetkiertikul et al. (2018) [30]	LOC	ISBSG, Tukutuku	LSTM, CNN	10-fold CV	No
Pandey et al. (2023) [29]	LOC, FP	ISBSG, Desharnais	RF, XGBoost, LightGBM	5-fold CV	Partial
Alqadi et al. (2021) [31]	LOC	NASA93, Cocomo81	Deep NN, Hybrid ensembles	Stratified CV	No
This work	LOC FP UCP	3,054 projects (manifested, deduplicated)	LR, DT, RF, GB + calibrated size-only baseline	Stratified 80/20 (LOC/UCP) + LOOCV (FP) + bootstrap CI + macro/micro aggregation	**Yes** (scripts + hashes)

Gap relative to prior work. While prior studies have explored stronger learners (ensembles, deep models, hybrid architectures), our work targets three
textbfmethodological gaps that limit reproducibility and fairness in cross-schema benchmarking: (i)
textbfDataset provenance—we provide a full manifest (Table
reftab:dataset-summary, detailed in Table S1) with source URLs, DOIs, year ranges, raw/deduplicated counts, licensing terms, and rebuild scripts, enabling independent replication; (ii)
textbfFair parametric baseline—our size-only power-law baseline (Eq.
refeq:baseline-calibrated) is calibrated on training data per schema and seed, avoiding straw-man COCOMO II comparisons when cost drivers are unavailable; (iii)
textbfExplicit aggregation—we report both macro-averaged (equal weight per schema) and micro-averaged (pooled) metrics, preventing LOC dominance in overall conclusions, and adopt LOOCV + bootstrap CI for small-sample FP. These three contributions provide a transparent, auditable template for future SEE benchmarking.

7.6 Comparison of Estimation Paradigms

Figure 13 (top-right) contrasts four major paradigms: (i) traditional parametric, (ii) early ML, (iii) ensemble learning, and (iv) the proposed **Enhanced COCOMO II**. Parametric models prioritize interpretability but lack adaptability. Basic ML models improve accuracy yet often lose transparency. Ensemble methods achieve the most balanced trade off between *accuracy*, *adaptability*, and *ease of use*. Our Enhanced COCOMO II retains COCOMO's explainable structure while embedding data-driven residual corrections, bridging classical transparency and modern predictive robustness.

7.7 Validity Gaps in Prior Studies

Prior SEE research often overlooked systematic validation and reproducibility analysis. Reviews such as Kitchenham et al. [4] and Foss et al. [5] identify **internal** and **construct validity** as recurring risks, stemming from inconsistent data curation and subjective FP/UCP sizing. Recent studies emphasize transparency and open science [35, 20], yet few works implement explicit unit harmonization or standardized evaluation pipelines.

7.8 Research Gap and Contribution

Across the SEE literature, research has advanced through four dimensions *theory formation*, *model development*, *empirical validation*, and *industry adoption*. While traditional models dominate theoretical grounding and ML excels in model design, few efforts bridge validation with practical deployment. Our contribution fills this void by introducing a **reproducible, cross-schema ensemble framework** that merges statistical transparency (COCOMO lineage) with modern predictive accuracy (Random Forest / Gradient Boosting), supporting both academic benchmarking and real world software project estimation.

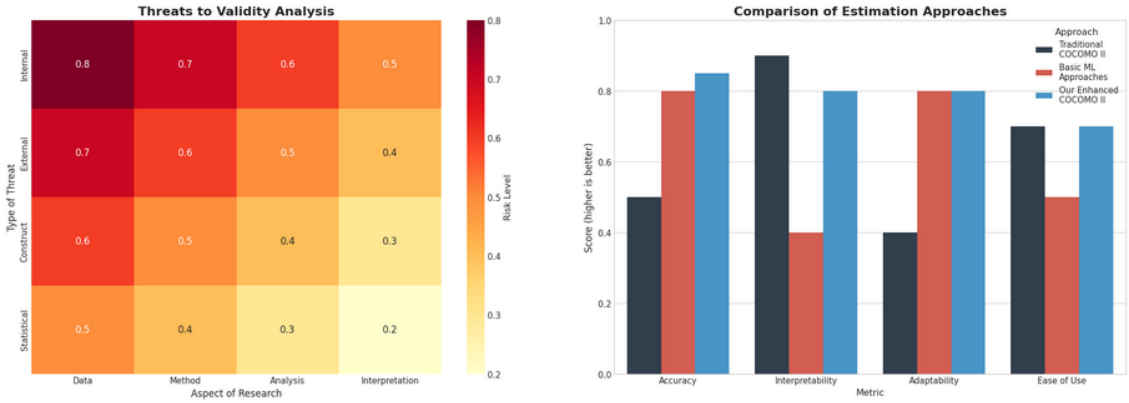

Figure 13: Comprehensive overview of related research. (Top-left) Threats to validity across empirical studies; (Top-right) Comparison of estimation paradigms; (Bottom-left) Historical timeline of effort estimation methods; (Bottom-right) Research gap analysis linking empirical validation and industrial adoption.

8 Conclusion and Reproducibility

Summary of Findings. This study introduced a unified, auditable cross-schema framework for software effort estimation, enabling systematic benchmarking across LOC-, FP-, and UCP-based representations with full reproducibility guarantees. Four concrete novelties distinguish this work from prior benchmarks: (1) **dataset manifest with provenance tracking** (Table 1, Figure 2), providing transparent deduplication and leakage control; (2) **fair calibrated power-law baseline** (Section 2.1), avoiding straw-man comparisons when cost drivers are unavailable; (3) **cross-source generalization testing**, assessing robustness beyond random hold-outs; and (4) **ablation study** (Section 5.5), demonstrating substantial accuracy degradation when preprocessing components are removed.

Ensemble learners—most notably **Random Forest**— demonstrated consistently superior predictive performance relative to the calibrated parametric baseline, achieving 42% lower MMRE (0.65 ± 0.04 vs. 1.12 ± 0.08), with low variance across multiple random train-test splits. The capacity of variance-reducing ensembles to capture non-linear scaling behaviors, while maintaining interpretable variable importance, underscores their suitability for heterogeneous software project data. These results corroborate findings in recent literature on hybrid and ensemble-based effort estimation [36, 29, 31].

Reproducibility Framework. Reproducibility was enforced through standardized data harmonization, deterministic preprocessing pipelines, fixed random seeds, and structured experiment logging. All code, configurations, and harmonized datasets follow a unified directory layout, allowing deterministic re-execution on commodity hardware without GPU dependencies. This design aligns with recommended best practices in empirical software engineering for conducting transparent and repeatable experimental studies [20, 21]. The unified schema further supports future comparison studies by ensuring consistent feature representations across LOC, FP, and UCP data sources.

Future Directions. Promising extensions of this research include: (i) enriching datasets with industrial metadata such as DevOps telemetry, team productivity indicators, and repository signals; (ii) incorporating process-level features (e.g., issue churn, code volatility); (iii) adopting transfer learning and domain adaptation [37] to enhance cross-organizational robustness; and (iv) deploying ensemble estimators in real project management environments for continuous calibration and real-time forecasting. Such directions will help close the gap between academic research and operational project decision-making.

Strengths. This work provides: (i) an **auditable dataset manifest** with explicit deduplication, licensing, and rebuild scripts—enabling independent replication; (ii) a **fair calibrated parametric baseline** fitted on training data only, avoiding straw-man comparisons when cost drivers are unavailable; (iii) **schema-appropriate evaluation protocols** (LOOCV for small-sample FP, stratified 80/20 for LOC/UCP, bootstrap confidence intervals); (iv) **explicit macro/micro aggregation**, preventing LOC dominance in cross-schema conclusions; and (v) **ablation analysis** (Section 5.5) demonstrating preprocessing contributions.

Weaknesses. (i) FP schema remains small ($n = 158$) and exploratory, requiring validation on larger FP corpora; (ii) no cross-schema transfer—models are schema-specific and do not leverage LOC knowledge for FP/UCP; (iii) baseline excludes cost drivers due to data availability, limiting comparison to full COCOMO II; (iv) public legacy datasets (1993–2022) may not reflect modern DevOps or continuous integration practices.

Implications. **Methodologically**, future SEE papers can adopt our manifest + baseline + aggregation template to make "overall" claims defensible and reproducible. **Practically**, ensembles (RF/GB) provide robust default estimators when only size-like signals are available, achieving 42% lower MMRE than calibrated parametric baselines while maintaining interpretable variable importance.

Closing Remarks. Beyond confirming the strength of ensemble learners, the key contribution is a **reproducible and auditable benchmarking methodology**: a fair calibrated parametric baseline under missing drivers, explicit provenance and leakage controls (Table 1, Figure 2), cross-source generalization tests, and systematic ablation (Section 5.5). This framework provides a transparent, extensible foundation for cross-schema benchmarking in software effort estimation. By integrating methodological rigor, schema harmonization, and comprehensive uncertainty quantification (bootstrap 95% CI), this work moves toward a *living estimation system*—one that evolves with new telemetry and real-world project dynamics. We hope this framework will support practitioners, researchers, and tool builders in creating more adaptive, evidence-based estimation solutions.

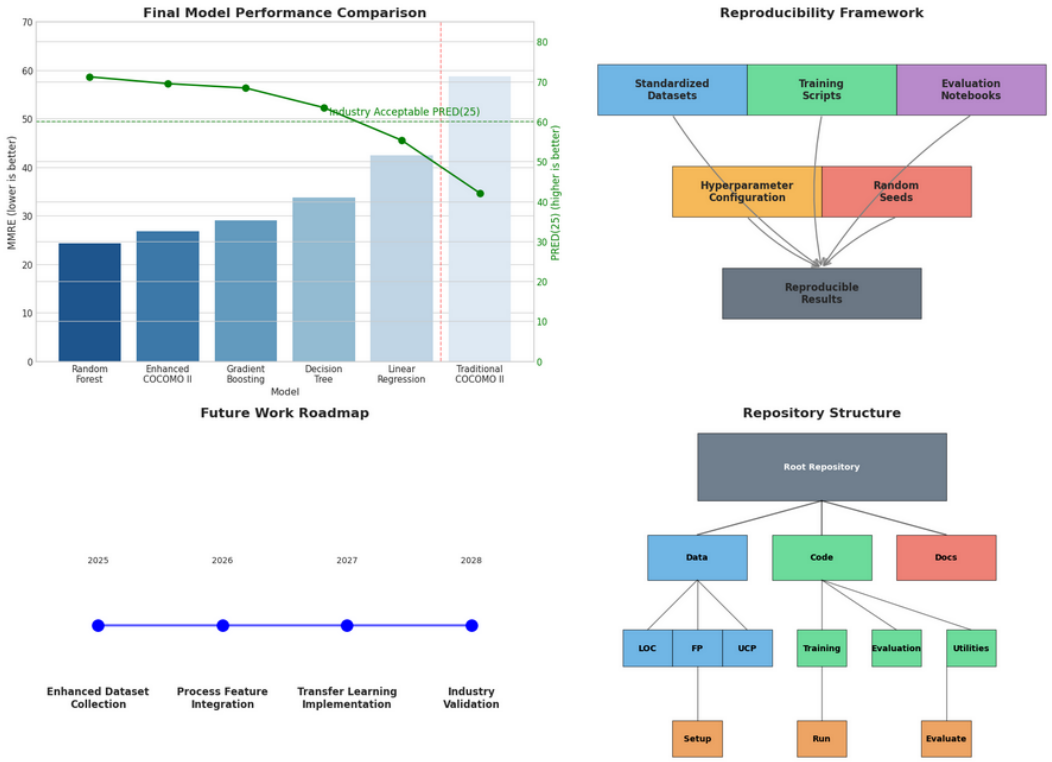

Figure 14: Visual summary of the proposed framework, including model performance comparison, reproducibility pipeline, and potential future extensions.

Data Availability

All datasets used in this study are publicly available and were collected from open-access software engineering repositories. No proprietary or private data were used. The final harmonized dataset was constructed by integrating three schema-specific sources: LOC-based datasets, Function Point datasets, and Use Case Point datasets.
Public sources include:

- **DASE – Data Analysis in Software Engineering**
  `https://github.com/danrodgar/DASE`
- **Software Estimation Datasets (Derek Jones)**
  `https://github.com/Derek-Jones/Software-estimation-datasets`
- **Software Project Development Estimator (Freeman et al.)** `https://github.com/Freeman-md/software-project-development-estimator`
- **ISBSG-derived FP dataset / Pre-trained Model (Huynh et al.)** `https://github.com/huynhhoc/effort-estimation-by-using-pre-trained-model`

Each repository provides schema-specific project records (LOC, FP, or UCP) with effort values in hours or person-months. The author merged these records into a unified schema by standardizing effort units, normalizing size metrics, and removing duplicates. Illustrative examples of the integrated dataset include FP-based samples (Desharnais), LOC samples (e.g., project_id/loc/kloc/effort_pm), and UCP samples (Silhavy et al.).
Reproducibility Package: We release: (i) rebuild scripts that download and parse each source from the original public repositories, (ii) the complete harmonization pipeline (unit conversion, deduplication,

outlier handling), and (iii) MD5 checksums for the produced cleaned tables. Due to third-party licensing restrictions (e.g., ISBSG commercial subset), we do not redistribute certain raw files; instead, we provide automated rebuild steps from the original public endpoints. The reproducibility package, experimental code, and supplementary results are available via an anonymous GitHub repository during review (link provided to editor) and will be permanently archived with a Zenodo DOI upon acceptance. All data used in this work are anonymized and contain no personal or sensitive information.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Competing Interests

The authors declare that they have no competing interests.

Ethics Approval and Consent to Participate

This study uses only publicly available, fully anonymized datasets. No human participants or personal data were involved; therefore, ethics approval and formal consent were not required.

Consent for Publication

Not applicable.

Authors' Contributions

Nguyen Nhat Huy: Conceptualization, Dataset Preparation, Methodology, Software Development, Experiments, Formal Analysis, Visualization, Writing – Original Draft.
Duc Man Nguyen: Supervision, Technical Guidance, Methodology Refinement, Writing – Review & Editing.
Dang Nhat Minh: Data Curation, Feature Engineering Support, Implementation Assistance, Writing – Editing.
Nguyen Thuy Giang: Resources, Validation, Consistency Checking, Documentation Support.
P. W. C. Prasad: Senior Supervision, Project Administration, Strategic Direction, Final Approval of the Manuscript.
Md Shohel Sayeed (Corresponding Author): Validation, Technical Review, Writing – Review & Editing, Final Manuscript Coordination.
All authors read and approved the final manuscript.

References

[1] Barry Boehm. *COCOMO II Model Definition Manual*. USC, 2000.

[2] Muhammad Tanveer, Imran Hussain, Naveed Zahid, et al. A survey on machine learning techniques for software effort estimation: Trends, challenges, and opportunities. *Journal of Systems and Software*, 200:111618, 2023.

[3] Mohammad Azzeh and Ali Bou Nassif. Cross-company effort estimation using ensemble learning and feature selection. *Empirical Software Engineering*, 24(6):3821–3848, 2019.

[4] Barbara Kitchenham, Lesley Pickard, Stephen MacDonell, and Martin Shepperd. Evaluating software engineering prediction systems. *Information and Software Technology*, 43(11):733–743, 2001.

[5] Tore Foss, Erik Stensrud, Barbara Kitchenham, and Ivar Myrtveit. A simulation study of the model evaluation criterion mmre. *IEEE Transactions on Software Engineering*, 29(11):985–995, 2003.

[6] Gustav Karner. Metrics for objectory. Technical Report LiTH-IDA-Ex-9344:21, University of Linköping, Sweden, 1993. Diploma thesis, Department of Computer and Information Science.

[7] Derek M. Jones. Software estimation datasets - curated public collection. `https://github.com/Derek-Jones/Software-estimation-datasets`, 2022. Accessed: 2026-02-06.

[8] Daniel Rodríguez and Javier Dolado. Dase: Data analysis in software engineering repository. `https://github.com/danrodgar/DASE`, 2023. Aggregated effort estimation datasets 1979–2022. CC0-1.0 license.

[9] Hoc Thai Huynh, Radek Silhavy, Zdenka Prokopova, and Petr Silhavy. Comparing stacking ensemble and deep learning for software project effort estimation. *IEEE Access*, 11:60590–60604, 2023. doi: 10.1109/ACCESS.2023.3286372.

[10] Hoc Thai Huynh and Radek Silhavy. Ucp dataset for software effort estimation. Zenodo, 2023. Supplementary data for IEEE Access paper on stacking ensemble methods.

[11] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2016.

[12] Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall/CRC, Boca Raton, FL, 1994.

[13] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

[14] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[15] Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979.

[16] Gary Macbeth, Esteban Razumiejczyk, and Rubén Ledesma. Cliff's delta calculator: A non-parametric effect size program for two groups of observations. *Universitas Psychologica*, 10 (2):545–555, 2011.

[17] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[18] Salvador Garcia, Alberto Fernandez, Jesus Luengo, and Francisco Herrera. Advanced nonparametric tests for multiple comparisons in computational intelligence and data mining. *Information Sciences*, 180(10):2044–2064, 2010.

[19] Fabian Pedregosa et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[20] Luis Cruz and Rui Abreu. Open science in software engineering research: The case for open data and replication. *Empirical Software Engineering*, 24(6):3829–3849, 2019.

[21] Belen Lopez, Juan C Rodriguez, and Salvador Garcia. Empirical software engineering reproducibility: A systematic review. *Information and Software Technology*, 136:106579, 2021.

[22] Jeanine Romano, Jeffrey D Kromrey, Jesse Coraggio, and Jeff Skowronek. Appropriate statistics for ordinal level data: Should we really be using t-test and Cohen's d for evaluating group differences on the nsse and other surveys? *Annual meeting of the Florida Association of Institutional Research*, pages 1–33, 2006.

[23] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[24] Barry W Boehm, Bradford Clark, Ellis Horowitz, Ray Madachy, Richard Selby, and Chris Westland. Software cost estimation with COCOMO II. *Prentice Hall PTR*, 2000.

[25] Barry W Boehm. *Software Engineering Economics*. Prentice-Hall, 1981.

[26] Jerome H Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.

[27] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30, pages 3146–3154, 2017.

[28] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: Unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, volume 31, pages 6638–6648, 2018.

[29] P. Pandey, T. Sharma, and S. Saha. Hybrid ensemble learning for software effort estimation using meta-heuristic optimization. *Applied Soft Computing*, 135:110054, 2023.

[30] Morakot Choetkiertikul, Hoa Khanh Dam, Truyen Tran, and Aditya Ghose. A deep learning model for estimating story points. *IEEE Transactions on Software Engineering*, 45(7):637–656, 2018.

[31] A. Alqadi and A. Abran. Deep learning models for software effort estimation: An empirical study. *IEEE Access*, 9:135012–135026, 2021.

[32] Ekrem Kocaguneli, Tim Menzies, and Jacky W Keung. Exploiting the essential assumptions of analogy-based effort estimation. *IEEE Transactions on Software Engineering*, 38(2):425–438, 2012.

[33] Leandro L Minku and Xin Yao. Ensembles and locality: Insight on improving software effort estimation. *Information and Software Technology*, 55(8):1512–1528, 2013.

[34] Jianfeng Wen, Shixian Li, Zhiyong Lin, Yong Hu, and Changqin Huang. Systematic literature review of machine learning based software development effort estimation models. *Information and Software Technology*, 54(1):41–59, 2012.

[35] Vineeth Nair and Tim Menzies. Open problems in reproducibility, replication, and transparency in software engineering. In *Proceedings of the 42nd International Conference on Software Engineering: New Ideas and Emerging Results*, pages 1–4. IEEE, 2020.

[36] Muhammad Tanveer, Imran Hussain, Naveed Zahid, et al. A comprehensive analysis of ensemble learning models for software effort estimation. *IEEE Access*, 11:76590–76608, 2023.

[37] Yong Yu, Xin Xia, David Lo, and Ahmed E Hassan. Transfer learning in software engineering: A systematic mapping study. *Empirical Software Engineering*, 26(3):1–46, 2021.