RELIABLE EXPLAINABLE AI FRAMEWORK FOR SOFTWARE DEFECT PREDICTION

Submitted in partial fulfilment of the requirements of the degree of

Master of Technology

by

Udit Jain
(Roll No: 24CSM1R23)

Supervisor:

Dr. Manjubala Bisi
Assistant Professor, Department of CSE

Department of Computer Science and Engineering
National Institute of Technology, Warangal
India

September 27, 2025

Acknowledgement

I would like to express my sincere gratitude to my project supervisor,

Dr. Manjubala Bisi, Assistant Professor, Department of Computer Science and Engineering, National Institute of Technology Warangal, for her valuable guidance, encouragement, and continuous support throughout the course of this project. Her insights, suggestions, and constructive feedback have been instrumental in shaping this work.

I also extend my heartfelt thanks to all the faculty members and staff of the Department of Computer Science and Engineering, NIT Warangal, for providing an excellent academic and research environment, and for facilitating the resources required for the successful completion of this project.

I am deeply grateful to my friends and batchmates for their ideas, encouragement, and assistance during various phases of this work. Their moral support played a key role in keeping me motivated.

A special note of appreciation goes to my family for their unconditional support, patience, and constant encouragement throughout my academic journey.

Finally, I am thankful to the National Institute of Technology Warangal for providing the infrastructure and learning environment that enabled me to carry out this project successfully.

Udit Jain
Roll No: 24CSM1R23

Declaration

I hereby declare that the work presented in this dissertation is the outcome of my own research carried out under the supervision of

Dr. Manjubala Bisi, Assistant Professor, Department of Computer Science and Engineering, National Institute of Technology, Warangal.

To the best of my knowledge, this dissertation has not been submitted either in part or in full for the award of any other degree or diploma at this or any other institution. Wherever the work of others has been used, it has been duly acknowledged and properly cited.

I further affirm that I have maintained the principles of academic integrity and honesty throughout the preparation of this dissertation, and I take full responsibility for the content presented herein.

(Signature)
Udit Jain
Roll No: 24CSM1R23

Approval Sheet

This is to certify that the dissertation work entitled

"Reliable Explainable AI Framework for Software Defect Prediction"

submitted by **Udit Jain (Roll No: 24CSM1R23)**

has been examined and is hereby approved for the award of the degree of **Master of Technology**.

Examiners:

Prof. Manish Kumar Bajpai
Prof. Earnest Paul Ijjina
Prof. Sarath Babu
Prof. Sanjaya Kumar Panda
Prof. P. Venkata Subba Reddy

Supervisor:

Dr. Manjubala Bisi

Chairman:

Dr. Krishna M. Ella

Head of Department (CSE):

Dr. Rashmi Ranjan Rout

Department of Computer Science and Engineering
National Institute of Technology, Warangal, India

Certificate

This is to certify that the dissertation work entitled
"Reliable Explainable AI Framework for Software Defect Prediction"
is a bonafide record of research carried out by **Udit Jain (Roll No: 24CSM1R23)** under my supervision.

The dissertation has been submitted to the Department of Computer Science and Engineering, National Institute of Technology, Warangal, in partial fulfilment of the requirements for the award of the degree of **Master of Technology** during the academic year 2024–2025.

(Signature)
Dr. Manjubala Bisi
Assistant Professor
Department of Computer Science and Engineering
National Institute of Technology, Warangal

Abstract

Software defect prediction plays a crucial role in improving software quality by identifying defect-prone modules early in the development cycle. While machine learning models such as Random Forests and other ensemble techniques have shown promising results in predicting software defects, their adoption in practice is often hindered by a lack of trust in model explanations. Existing studies primarily focus on accuracy and interpretability but rarely assess the reliability of these explanations.

This dissertation proposes a **Reliable Explainable AI Framework** that integrates SHAP (SHapley Additive exPlanations) with novel reliability metrics. In addition to standard measures of generalizability, concordance, and stability, the framework introduces three behavior-grounded metrics: (i) *Gradient-of-Loss Reliability (GLR)* to evaluate alignment between SHAP feature importance and true loss sensitivity, (ii) *ε-Perturbation Hit@k* to test whether SHAP-identified top features genuinely affect model loss under small perturbations, and (iii) *Expected Calibration Error (ECE)* to assess the reliability of predicted probabilities.

Experiments were conducted on the NASA PROMISE CM1 dataset using Random Forest classifiers. Results demonstrate that while conventional SHAP explanations achieve high generalizability, their reliability varies across folds and conditions. The proposed framework provides a composite reliability score, offering a more comprehensive and trustworthy evaluation of explainability.

This work bridges the gap between interpretability and reliability in explainable AI for software defect prediction. By validating explanations against model behavior and calibration, the framework enhances confidence for developers and stakeholders, moving closer to trustworthy AI in software engineering.

Keywords: Explainable AI, Reliability, SHAP, Software Defect Prediction, Random Forest, GLR, Hit@k, Calibration

Contents

Acknowledgement **i**

Declaration **ii**

Certificate **iv**

Abstract **v**

List of Figures **viii**

List of Abbreviations **ix**

1 Introduction **1**
1.1 Context and Motivation 1
1.2 Challenges in Current Systems 2
1.3 Reliability as a Core Requirement 2
1.4 Objectives and Scope of the Dissertation 3
1.5 Expected Contributions 3
1.6 Organization of the Report 3

2 Background **5**
2.1 Reliability in Explainable AI 5
2.2 Behavior-Grounded Reliability 6
2.3 Machine Learning for Defect Prediction 6
2.4 Explainability as a Tool for Reliability 6
2.5 Summary 7

3 Related Work **8**
3.1 Machine Learning for Software Defect Prediction 8
3.2 Explainable AI in Software Engineering 8
3.3 Reliability of Explanations 9
3.4 Positioning in Recent Literature (2023–2025) 10
3.5 Gaps and Open Questions 10
3.6 Need for This Study 10
3.7 Summary 11

4 Proposed Solution **12**
4.1 System Overview 12

4.2 Dataset and Preprocessing 14
4.3 Model Training and Standard Evaluation 14
4.4 Explainability using SHAP 15
4.5 Reliability Assessment 16
4.5.1 Classical Reliability Metrics 16
4.5.2 Behavior-Grounded Reliability Metrics 16
4.5.3 Composite Reliability Score 17
4.6 Pipeline Pseudocode 17
4.7 Visualization Outputs 17
4.8 Summary 21

5 Experimental Results **22**
5.1 Predictive Performance 22
5.2 Classical Reliability 22
5.3 Behavior-Grounded Reliability 23
5.4 Synthesis and Discussion 23

6 Conclusion and Future Work **24**
6.1 Conclusion 24
6.2 Implications of Findings 25
6.3 Limitations 25
6.4 Future Work (Next 2 Months) 25
6.5 Final Remarks 27

References **28**

List of Figures

4.1 Proposed framework pipeline: dataset preparation, preprocessing, Random Forest training, SHAP-based explanations, and reliability evaluation. 13
4.2 Top features ranked by mean absolute SHAP on test data (averaged over folds). Bar length reflects average contribution magnitude; note that mean $|\phi|$ conveys *strength* of influence, not its sign. 18
4.3 GLR distribution: per-sample Spearman ρ between SHAP feature ranks and ranks from local loss-sensitivity (finite-difference gradient of the logistic loss). Values > 0 indicate rank alignment between explanations and the model's behavior. 19
4.4 Calibration curve on held-out folds (ECE $\approx$ 0.110). The dashed diagonal denotes perfect calibration; points below it indicate over-confidence. 20

List of Abbreviations

AI	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning
SDP	Software Defect Prediction
XAI	Explainable Artificial Intelligence
SHAP	SHapley Additive exPlanations (feature attribution method)
LIME	Local Interpretable Model-Agnostic Explanations
RF	Random Forest (tree-ensemble classifier)
SVM	Support Vector Machine
CNN	Convolutional Neural Network
LCNN	Lightweight Convolutional Neural Network
MLP	Multi-Layer Perceptron
NAM	Neural Additive Model (inherently interpretable)
XNAM	eXplainable Neural Additive Model (NAM with interactions)
SMOTE	Synthetic Minority Over-sampling Technique (imbalance handling)
CV	Cross-Validation
AUC	Area Under the ROC Curve (discrimination)
F1	Harmonic mean of Precision and Recall
ECE	Expected Calibration Error (probability calibration)
GLR	Gradient-of-Loss Reliability (behavior-grounded faithfulness)
Hit@k	Overlap of explainer top-k features with top-k loss-sensitive features
Brier	Brier Score (mean squared error of probabilities)
LOC	Lines of Code
G/C/S	Generalizability / Concordance / Stability (classical reliability)

Chapter 1

Introduction

Software systems form the backbone of modern society, supporting critical infrastructure, business operations, and everyday applications. As reliance on software grows, ensuring its quality and reliability has become increasingly important. Software defects not only raise maintenance costs and reduce performance but can also lead to significant financial losses or safety risks. Detecting defect-prone modules early in the development cycle enables developers to optimize resources, reduce testing overhead, and improve overall software quality.

1.1 Context and Motivation

Machine Learning (ML) models such as Random Forests, Gradient Boosting, and Neural Networks have demonstrated strong predictive power in software defect prediction (SDP). They can capture complex, non-linear relationships among software metrics such as size, complexity, or coupling. However, despite their predictive strength, these models are often treated as "black boxes," making it difficult for practitioners to understand or trust their outputs.

Explainable AI (XAI) has emerged to address this issue by providing post-hoc explanations for model predictions. Among these techniques, SHAP (SHapley Additive exPlanations) is widely adopted for attributing feature importance in both local and global contexts. Yet, interpretability alone is not enough — what truly matters is whether these explanations are **reliable**. Explanations that vary across folds, disagree with model behavior, or fail to generalize to unseen data may create false confidence and hinder adoption in practice.

1.2 Challenges in Current Systems

Despite advancements in AI-driven defect prediction, several critical challenges remain:

- **Opaque Predictions:** High-performing models provide limited insight into the reasoning behind their outputs.
- **Data Constraints:** Software defect datasets are often imbalanced, noisy, and project-specific, affecting robustness.
- **Unreliable Explanations:** Post-hoc methods like SHAP or LIME are rarely evaluated for stability, generalizability, or alignment with the model's actual behavior.
- **Trust Deficit:** Developers hesitate to rely on AI models if explanations cannot be validated or shown to be consistent under perturbations.

1.3 Reliability as a Core Requirement

Interpretability must be complemented by reliability for explanations to be meaningful. Reliable explanations should remain stable across data splits, align with internal model signals, and reflect genuine sensitivity of the model's loss function to feature perturbations. To capture these aspects, reliability can be assessed through two complementary layers:

- **Classical Reliability Dimensions:** Generalizability (train–test consistency), Concordance (agreement with model-internal importances), and Stability (consistency across folds).
- **Behavior-Grounded Metrics:**
  - **Gradient-of-Loss Reliability (GLR):** Evaluates whether SHAP ranks align with loss sensitivity derived from perturbations.
  - **ε-Perturbation Hit@k:** Validates whether SHAP's top-k features also cause the largest changes in model loss under small perturbations.
  - **Expected Calibration Error (ECE):** Measures how well predicted probabilities reflect actual defect frequencies, ensuring probability honesty.

These metrics go beyond interpretability and provide quantitative checks to ensure that explanations are not only understandable but also trustworthy.

1.4 Objectives and Scope of the Dissertation

The objective of this dissertation is to design a **Reliable Explainable AI Framework** for software defect prediction. The scope of the work includes:

- Preprocessing defect datasets and addressing imbalance using methods such as SMOTE and class weighting.
- Training ML models, with Random Forests as the primary classifier, to distinguish defective from non-defective modules.
- Generating feature-level explanations using SHAP.
- Extending evaluation to reliability metrics including GLR, ε-Hit@k, and ECE, alongside classical measures of generalizability, concordance, and stability.
- Proposing a composite reliability score that integrates these dimensions for a holistic assessment of explanation trustworthiness.

1.5 Expected Contributions

The contributions of this dissertation are expected to be:

- A structured framework that evaluates both interpretability and reliability of XAI in software defect prediction.
- Integration of behavior-grounded metrics (GLR, Hit@k, and ECE) into the reliability evaluation process.
- Empirical validation of the framework on NASA PROMISE datasets, demonstrating its effectiveness in producing stable and trustworthy explanations.
- Practical insights for software engineers and project managers to adopt AI-based defect prediction systems with greater confidence.

1.6 Organization of the Report

The rest of this dissertation is organized as follows:

- **Chapter 1: Introduction** — Describes motivation, challenges, objectives, and contributions.
- **Chapter 2: Background** — Reviews defect prediction, explainability, and reliability dimensions in XAI.

- **Chapter 3: Literature Survey** — Analyzes prior research, highlighting the gap in reliability assessment.
- **Chapter 4: Proposed Methodology** — Presents the pipeline, SHAP integration, and reliability metrics.
- **Chapter 5: Experimental Results** — Provides results, visualizations, and reliability analysis.
- **Chapter 6: Conclusion and Future Work** — Summarizes findings and outlines possible extensions.

Chapter 2

Background

The rapid adoption of Artificial Intelligence (AI) and Machine Learning (ML) in software engineering has transformed the way defects are predicted and managed. While predictive models achieve high accuracy, their trustworthiness often remains questionable. Developers and stakeholders are hesitant to rely on predictions that lack consistency, stability, or verifiable reasoning. This gap highlights an urgent need to shift the focus from mere interpretability to **reliability of explanations**.

2.1 Reliability in Explainable AI

Explainable AI (XAI) provides feature-level insights into model behavior. However, explanations are only useful if they are reliable — meaning that they remain consistent across folds, datasets, and perturbations, and align with the actual behavior of the underlying model. Without reliability, explanations can be misleading and may erode user trust instead of building it.

Reliability in XAI can be viewed along three fundamental dimensions:

- **Generalizability:** Do explanations generated on training data hold true for unseen test data?
- **Concordance:** Do post-hoc explanations agree with the model's internal feature importance measures?
- **Stability:** Are explanations consistent across different cross-validation folds or resampling conditions?

These dimensions ensure that explanations are not artifacts of specific data splits but reflect the broader behavior of the model.

2.2 Behavior-Grounded Reliability

Beyond general consistency checks, recent work emphasizes the need for *behavior-grounded reliability metrics* that validate explanations against the model's own decision boundaries and loss landscape. Three such measures are critical:

- **Gradient-of-Loss Reliability (GLR):** Evaluates whether the ranking of features by SHAP importance aligns with true loss sensitivity, estimated through small perturbations and gradients. A high GLR indicates that explanations are faithful to the way the model's loss responds to input changes.
- **ε-Perturbation Hit@k:** Tests whether SHAP's top-k features are also the features that cause the largest change in model loss under small perturbations. This metric validates whether explanations identify features with actual causal influence.
- **Expected Calibration Error (ECE):** Measures the gap between predicted probabilities and observed frequencies. Well-calibrated probabilities are essential for decision-making in high-stakes contexts, and ECE integrates probability honesty into explanation reliability.

Together, these metrics extend the reliability framework beyond surface-level interpretability and establish a direct link between explanations and model behavior.

2.3 Machine Learning for Defect Prediction

Software defect prediction (SDP) aims to classify software modules as defective or non-defective using metrics such as lines of code, cyclomatic complexity, coupling, or Halstead measures. Predictive models like Random Forests, Gradient Boosting, and Neural Networks have shown strong results in this area.

However, while these models are effective, they often operate as black boxes. Traditional evaluation metrics such as AUC or F1-score capture predictive ability but provide no insight into whether explanations are stable, aligned with model behavior, or calibrated. This disconnect underscores the importance of integrating reliability checks into the SDP pipeline.

2.4 Explainability as a Tool for Reliability

Explainability methods such as SHAP (SHapley Additive exPlanations) provide a way to attribute predictions to features. SHAP is widely adopted because it offers:

- Local explanations for individual predictions.
- Global feature importance across datasets.
- Theoretical grounding in cooperative game theory.

While SHAP improves transparency, its true utility lies in being evaluated under reliability checks. Explanations that are interpretable but unreliable can mislead practitioners. Hence, SHAP serves as the foundation upon which reliability metrics such as GLR, Hit@k, and ECE are applied.

2.5 Summary

This chapter emphasized the importance of moving from interpretability to reliability in explainable AI for software defect prediction. We reviewed classical reliability dimensions (Generalizability, Concordance, Stability) and behavior-grounded extensions (GLR, Hit@k, ECE). Together, these form a comprehensive framework for ensuring that explanations are not only understandable but also trustworthy, stable, and behaviorally faithful. The next chapter will review existing literature to identify gaps in how reliability has been addressed in prior works.

Chapter 3

Related Work

Over the last decade, software defect prediction (SDP) has increasingly relied on machine learning (ML) to flag defect-prone modules early, helping teams prioritize testing and reduce maintenance effort. Although accuracy has improved steadily, practical uptake also depends on whether model explanations are *trustworthy*, not merely available. This chapter reviews: (i) ML approaches for SDP, (ii) explainability in software analytics, (iii) reliability of explanations, and (iv) open gaps that motivate our work.

3.1 Machine Learning for Software Defect Prediction

Early statistical models struggled with non-linear relationships between software metrics and defects. Modern ML methods address this limitation:

- **Tree ensembles.** Decision Trees, Random Forests, Gradient Boosting, XGBoost and related ensembles remain strong baselines on tabular code/process metrics, offering competitive accuracy and some intrinsic importance estimates.
- **Deep models.** CNN/LSTM variants and lightweight CNNs have been applied to SDP to learn richer representations; however, their opacity can hinder adoption in safety- or cost-sensitive pipelines.

Recent studies pair such predictors with explanation tools so that findings translate into actionable process improvements (e.g., highlighting size, complexity, or coupling patterns that repeatedly precede defects).

3.2 Explainable AI in Software Engineering

Explainable AI (XAI) has been used to make SDP more transparent to practitioners. Model-agnostic post-hoc methods such as *LIME* and *SHAP*

attribute a prediction to input features, revealing which metrics (e.g., LOC, cyclomatic complexity, coupling, churn) drove the outcome. Two complementary strands are visible in recent work:

- **Post-hoc explanations for black-box models.** Studies train strong predictors (tree ensembles, CNNs) and then use SHAP/LIME to interpret both global and local behavior; some cross-company analyses favor keeping defect counts as continuous targets to avoid information loss and then use SHAP to surface influential factors.
- **Inherently interpretable neural models.** eXplainable Neural Additive Models (XNAMs) adapt NAM/GAM ideas to SDP by assigning each feature a learned, *inspectable* shape function and exposing pairwise interactions; this reduces dependence on fragile post-hoc surrogates while remaining competitive in accuracy.

3.3 Reliability of Explanations

Interpretability alone is insufficient if explanations are inconsistent or misaligned with model behavior. Recent work argues for explicit, quantifiable *reliability* checks:

- **Generalizability.** Do explanation rankings remain similar across folds or train/test splits (e.g., via rank correlation)?
- **Concordance.** Do XAI importances agree with intrinsic importances from the fitted model (e.g., impurity-based scores for Random Forests)?
- **Stability.** Are explanations for *nearby* inputs close to each other (small changes in inputs $\Rightarrow$ small changes in attributions)?

Behavior-grounded probes further test whether explanations reflect what *actually* drives the model:

- **Gradient-of-Loss Reliability (GLR).** Check if the ranking from SHAP (or another explainer) aligns with the sensitivity of the loss to feature-wise perturbations.
- **ε-Perturbation Hit@k.** Verify that top-k features by the explainer truly induce the largest loss change under small perturbations.
- **Expected Calibration Error (ECE).** Ensure predicted probabilities are well calibrated so that confidence scores match observed frequencies.

Together, these metrics move the focus from "can we explain?" to "can we *trust* the explanation we see?".

3.4 Positioning in Recent Literature (2023–2025)

- **Cross-company SHAP analyses.** Using industry datasets, researchers predicted *defect counts* (rather than binarized labels) and reported size/effort/density as dominant drivers under SHAP; evaluation used MAE/MSE/RMSE and R^2.
- **Deep yet analyzable models.** Lightweight CNNs on PROMISE-style data were paired with LIME/SHAP to diagnose which metrics most influenced predictions, with practical notes on class imbalance handling.
- **Self-explaining neural models.** XNAMs visualized per-feature response curves and interaction maps while matching or exceeding black-box baselines, offering faithful, non-post-hoc interpretability.
- **Reliability methodology.** A k-fold protocol quantified Generalizability, Concordance, Stability and a composite reliability score; although introduced in healthcare contexts, the metrics are model- and domain-agnostic and transfer naturally to SDP.

3.5 Gaps and Open Questions

Despite steady progress, several issues remain open:

1. **Reliability is under-reported.** Many SDP–XAI studies stop at *visual* inspection without quantifying fold-to-fold agreement or agreement with model-intrinsic signals.
2. **Explainer choice is context-dependent.** Findings vary across datasets and models (e.g., LIME vs. SHAP), suggesting the need for *benchmarking reliability* rather than choosing a single explainer by default.
3. **Behavior-grounded tests are rare.** Few works verify that explainer top-k features truly govern loss or assess probability calibration.
4. **No common reliability index.** The field lacks a standard composite score to compare models/explainers on a level playing field.

3.6 Need for This Study

We develop a *Reliable XAI* framework for SDP that integrates SHAP/LIME and inherently interpretable models (e.g., XNAMs) with a reliability audit composed of: (i) fold-wise generalizability of rankings, (ii) concordance with

intrinsic importances, (iii) stability on near neighbors, and (iv) behavior-grounded checks (GLR, ε-Perturbation Hit@k, ECE). This dual emphasis—interpretability *and* reliability—aims to deliver explanations practitioners can act on with confidence.

3.7 Summary

In summary, ML has strengthened SDP performance and XAI has improved transparency, but explanation *reliability* remains the missing layer for trustworthy adoption. The next chapter operationalizes this audit for tree ensembles, deep models, and XNAMs on standard SDP datasets.

Chapter 4

Proposed Solution

This chapter introduces the proposed **Reliable Explainable AI Framework for Software Defect Prediction**. The novelty of the framework lies in extending beyond conventional interpretability to establish *reliability of explanations*, thereby ensuring that SHAP-based feature attributions are not only interpretable but also consistent, stable, and aligned with the model's true behavior.

4.1 System Overview

The overall workflow of the proposed framework is presented in Figure 4.1. It progresses through a structured sequence of five stages, starting from raw datasets and ending with reliability evaluation of the explanations.

1. **Dataset:** Software defect datasets such as CM1 and NASA PROMISE are used as the input sources.
2. **Preprocessing:** Includes handling missing values, feature cleanup, normalization, and imbalance treatment using SMOTE. A stratified k-fold split ensures balanced evaluation.
3. **Random Forest Model:** A Random Forest classifier is employed as the base learner due to its robustness on tabular metrics and ability to provide baseline importance scores.
4. **SHAP Explanations:** TreeExplainer is used to generate both global and local feature attributions, identifying which software metrics contribute most to defect predictions.
5. **Reliability Evaluation:** Explanations are assessed for trustworthiness. This involves:
    - **Classical checks:** Generalizability, Concordance, and Stability.

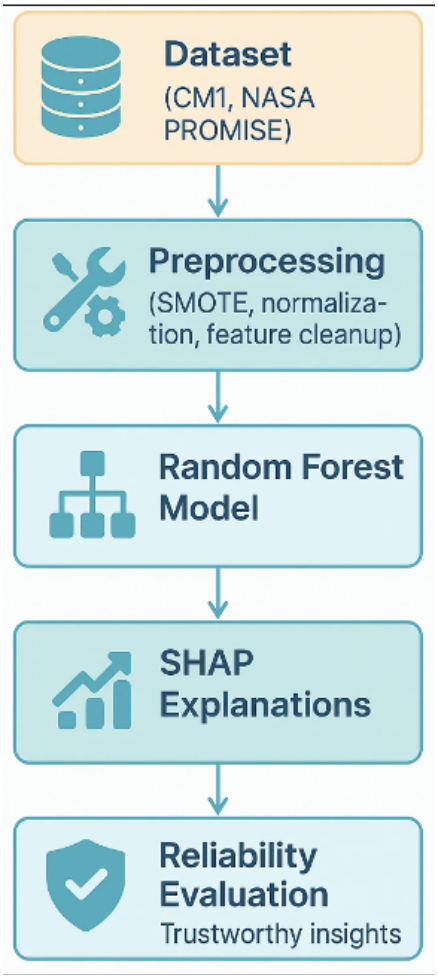

Figure 4.1: Proposed framework pipeline: dataset preparation, preprocessing, Random Forest training, SHAP-based explanations, and reliability evaluation.

- **Behavior-grounded checks:** Gradient-of-Loss Reliability (GLR), ε-Perturbation Hit@k, and Expected Calibration Error (ECE).

This structured pipeline ensures that outputs are not only accurate but also supported by explanations that are consistent, faithful, and reliable—making them actionable in practical software engineering scenarios.

4.2 Dataset and Preprocessing

The experiments use the **NASA PROMISE CM1 dataset**, representing modules of a NASA spacecraft instrument written in C.

Dataset Characteristics

- **Instances:** 327 software modules.
- **Features:** 35 static code metrics (LOC, McCabe complexity, Halstead measures, control-flow metrics).
- **Target:** Binary label (Defective = 1, Non-Defective = 0), with 42 defective and 285 non-defective modules (imbalanced).

Preprocessing Steps

- **Constant Features:** Columns with a single unique value are removed.
- **Missing Values:** Filled using forward fill followed by backward fill.
- **Imbalance:** SMOTE oversampling is applied only on the training folds.
- **Cross-validation:** 5-fold stratified cross-validation ensures balanced splits.

Formally:

$$\mathcal{D} = \{(x^{(i)}, y^{(i)}) \mid x^{(i)} \in R^d,\ y^{(i)} \in \{0,1\}\}, \quad i = 1, \ldots, n$$

4.3 Model Training and Standard Evaluation

We use a **Random Forest Classifier**, a tree-ensemble method that is robust to noise and interpretable via SHAP TreeExplainer.

Hyperparameters

- $n_{estimators} = 600$
- $min\_samples\_leaf = 2$
- $class\_weight = balanced\_subsample$

Evaluation Metrics

$$AUC = \text{ROC curve area (ranking quality)}$$
$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$
$$Brier = \frac{1}{N}\sum_{i=1}^{N}(p_i - y_i)^2$$

Results (average across folds):

- AUC = 0.7315
- F1 = 0.2349
- Precision = 0.2982
- Recall = 0.2444
- Brier = 0.1322

4.4 Explainability using SHAP

SHAP decomposes the model output into feature contributions:

$$f(x) = \phi_0 + \sum_{j=1}^{M} \phi_j$$

where ϕ_j denotes the contribution of feature j.

SHAP in SDP

- **Local Explanations:** Why a specific module is defective.
- **Global Explanations:** Which metrics (e.g., LOC, complexity) consistently drive defects.

Example: High LOC = +0.35 ⇒ increases defect risk; High comment density = –0.12 ⇒ reduces defect risk :contentReference[oaicite:2]index=2.

4.5 Reliability Assessment

4.5.1 Classical Reliability Metrics

- **Generalizability (G):** Train-test consistency of feature ranks.

$$G = \frac{1}{k}\sum_{j=1}^{k} \tilde{\rho}\big(rank(SHAP_{train}), rank(SHAP_{test})\big)$$

- **Concordance (C):** Agreement between SHAP importances and internal Gini importance.
- **Stability (S):** Cross-fold consistency of explanations.

From experiments :contentReference[oaicite:3]index=3:

$$G = 0.9866,\ C = 0.9154,\ S = 0.7788, \quad R = 0.9468$$

4.5.2 Behavior-Grounded Reliability Metrics

Gradient-of-Loss Reliability (GLR): Checks whether SHAP's ranking aligns with true sensitivity of the model's loss:

$$g_j(x) \approx \frac{\big|\ell(f(x+\varepsilon e_j), y) - \ell(f(x-\varepsilon e_j), y)\big|}{2\varepsilon}$$

Normalized g_j values compared with SHAP via Spearman correlation. Experimentally: $GLR_{mean} = 0.0742$:contentReference[oaicite:4]index=4.

ε-Perturbation Hit@k: Validates whether SHAP's top-k features overlap with features that cause maximum $\Delta\ell$ under perturbations:

$$Hit@k = 1\{Top\_k(SHAP) \cap Top\_k(\Delta\ell) \neq \emptyset\}$$

Result: $Hit@10 = 1.0$:contentReference[oaicite:5]index=5.

Expected Calibration Error (ECE): Assesses probability honesty by comparing predicted vs. observed defect frequencies:

$$ECE = \sum_{b=1}^{B} \frac{n_b}{N}\big|\text{acc}(b) - \text{conf}(b)\big|$$

Result: $ECE = 0.1098$:contentReference[oaicite:6]index=6.

4.5.3 Composite Reliability Score

We rescale GLR and combine:

$$ReliabilityScore = \frac{\tilde{\rho}_{GLR} + Hit@k + (1 - ECE)}{3}$$

Experimental result: $ReliabilityScore = 0.8091$:contentReference[oaicite:7]index=7.

4.6 Pipeline Pseudocode

Algorithm 1 Reliable Explainable AI Framework for Software Defect Prediction

```
Load dataset CM1
Preprocess (remove constants, impute missing, SMOTE imbalance)
Perform 5-fold stratified CV
for each fold j = 1 to 5 do
    Train Random Forest (n = 600, balanced subsample)
    Evaluate metrics: AUC, F1, Precision, Recall, Brier
    Compute SHAP values
    Compute G, C, S
    Estimate GLR via finite differences
    Compute ε-Hit@k
    Calculate ECE
end for
Aggregate fold results
Report Composite ReliabilityScore
```

4.7 Visualization Outputs

This section reports: (i) test-fold mean absolute SHAP for feature influence, (ii) the distribution of Gradient-of-Loss Reliability (GLR) across samples, and (iii) probability calibration with the Expected Calibration Error (ECE).

Reading Figure 4.2. Comment/size indicators (`LOC_COMMENTS`, `PERCENT_COMMENTS`, `NUMBER_OF_LINES`) lead the ranking, with complexity and design signals (`HALSTEAD_CONTENT`, `CYCLOMATIC_DENSITY`, `NORMALIZED_CYCLOMATIC_COMPLEXITY`, `DESIGN_DENSITY`) close behind. This pattern suggests (a) scale remains a strong driver, and (b) structural complexity adds complementary information. Because magnitudes are unsigned, directionality (risk-increasing vs. risk-reducing) should be inspected via class-wise beeswarms or per-instance waterfall plots before recommending code/process changes.

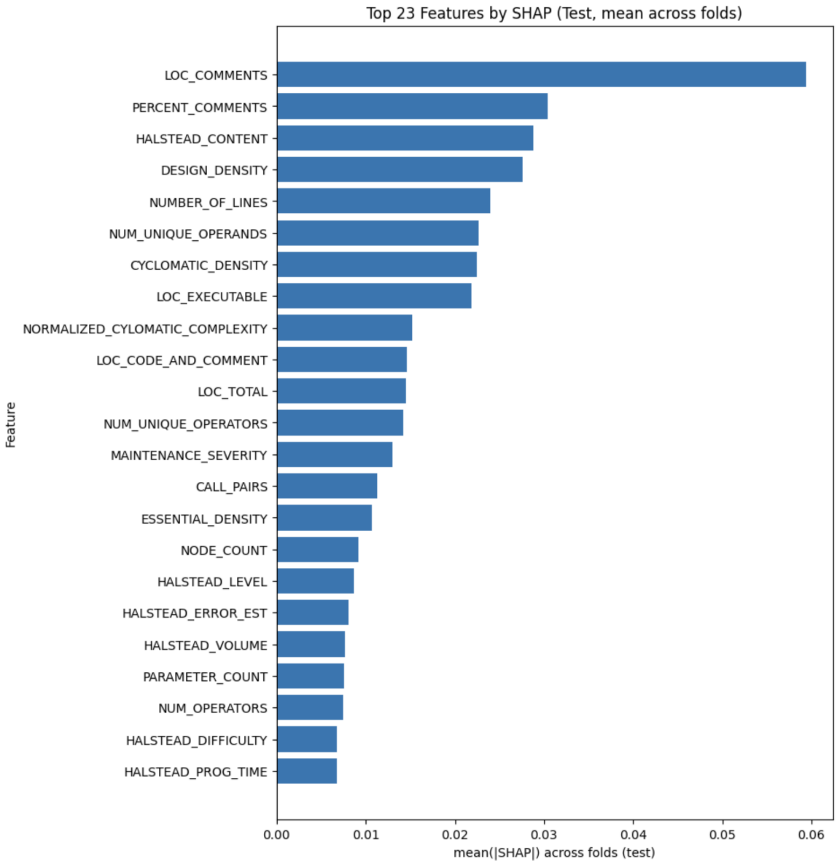

Figure 4.2: Top features ranked by mean absolute SHAP on test data (averaged over folds). Bar length reflects average contribution magnitude; note that mean $|\phi|$ conveys *strength* of influence, not its sign.

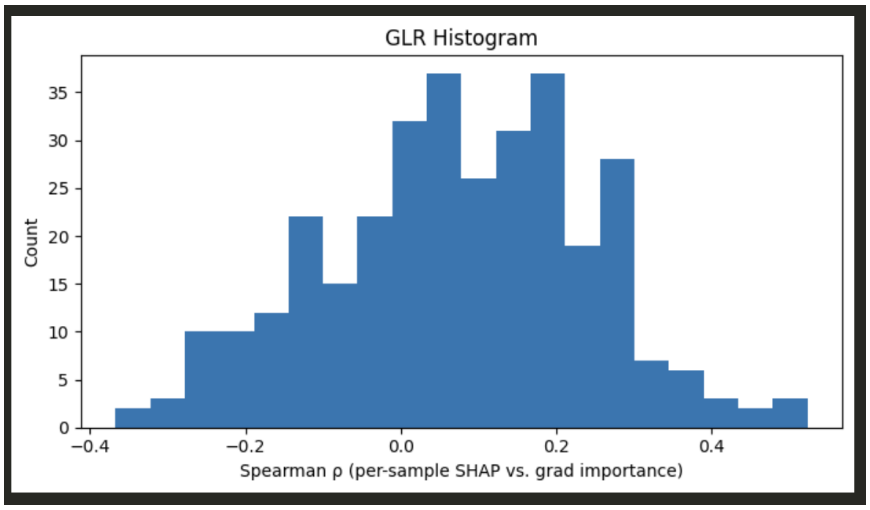

Figure 4.3: GLR distribution: per-sample Spearman ρ between SHAP feature ranks and ranks from local loss-sensitivity (finite-difference gradient of the logistic loss). Values > 0 indicate rank alignment between explanations and the model's behavior.

Reading Figure 4.3. The histogram is centered slightly above 0 with a mild right skew, indicating *some* agreement between SHAP rankings and true loss sensitivity. Mass near $\rho \approx 0$ and a visible left tail ($\rho < 0$) flag instances where explainer top-k does not coincide with the features that most perturb the loss locally. Actionably, prioritize audits for samples in the left tail; reliability can improve with stricter preprocessing (standardization, leakage checks) and by reporting GLR summaries (median/IQR and proportion with $\rho < 0$).

Reading Figure 4.4. Most bins lie below the diagonal, i.e., predicted probabilities exceed observed frequencies. An ECE near 0.11 reflects moderate miscalibration. For thresholded decisions, this may inflate perceived risk. Remedies: apply isotonic or Platt scaling (validated within folds), consider temperature scaling for neural models, and report both discrimination (AUC/F1) and calibration (ECE/Brier).

Synthesis of the three views

- **Feature influence (Figure 4.2).** Size/comment metrics and complexity/design measures jointly shape predictions. Check for collinearity (e.g., size vs. comment density) and examine signed effects before interventions.
- **Faithfulness (Figure 4.3).** Median $\rho > 0$ implies average alignment,

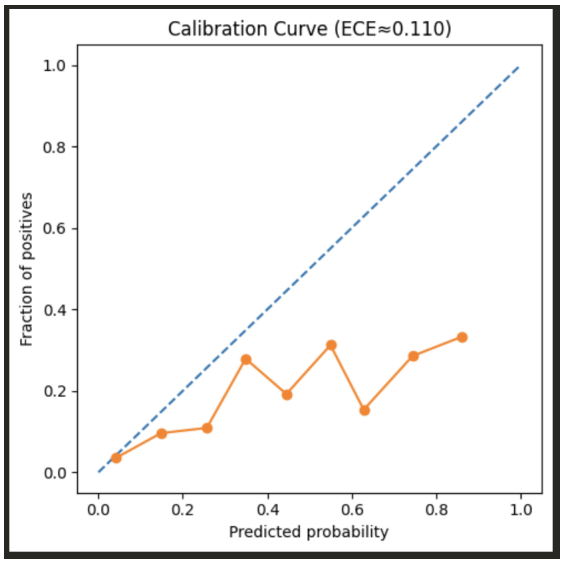

Figure 4.4: Calibration curve on held-out folds (ECE $\approx$ 0.110). The dashed diagonal denotes perfect calibration; points below it indicate over-confidence.

but dispersion and the negative tail justify behavior-grounded checks (GLR, ε-Hit@k) when explanations inform decisions.

- **Probability quality (Figure 4.4).** Over-confidence warrants calibration; improved calibration can indirectly stabilize behavior-based metrics by smoothing the loss landscape.

Actionable takeaways

1. **Keep SHAP, add sign diagnostics.** Augment with class-conditioned beeswarms and waterfalls for representative modules.

2. **Make reliability first-class.** Report GLR median/IQR and the share of samples with $\rho < 0$; add ε-Hit@k to test whether explainer top-k truly yields the largest Δloss.

3. **Calibrate and re-audit.** Apply isotonic/Platt on validation folds, then replot Figure 4.4 and track ECE/Brier deltas alongside any GLR shift.

4. **Control confounding.** Since size and comments co-vary, use partial-dependence or NAM/XNAM shape plots to tease apart individual effects; normalize comment metrics by executable LOC where appropriate.

4.8 Summary

This chapter described the proposed reliable XAI framework for software defect prediction. The approach integrates SHAP-based interpretability with both classical reliability checks (G, C, S) and behavior-grounded metrics (GLR, ε-Hit@k, ECE). Results on the CM1 dataset demonstrate that while SHAP explanations appear stable across train-test splits (high G, C), their alignment with true model behavior is weaker (low GLR), underscoring the need for reliability assessment. The composite ReliabilityScore provides a single, interpretable measure of explanation trustworthiness.

This chapter presents the outcomes of the experiments conducted during the development and evaluation of the proposed crop yield prediction system. The focus is on measuring prediction accuracy, comparing different machine learning models, and analyzing the interpretability of the predictions using SHAP.

Chapter 5

Experimental Results

This chapter presents the empirical results obtained with the **Random Forest** classifier and **TreeExplainer (SHAP)** on the NASA PROMISE CM1 dataset. We report: (i) standard predictive metrics, (ii) classical reliability, (iii) behavior-grounded reliability, and (iv) a synthesis of findings.

5.1 Predictive Performance

Table 5.1 summarizes the discrimination and probability quality achieved across 5-fold stratified cross-validation.

Table 5.1: Predictive performance metrics (mean across folds).

Model	**AUC**	**F1**	**Precision**	**Recall**	**Brier Score**
Random Forest	0.7315	0.2349	0.2982	0.2444	0.1322

5.2 Classical Reliability

Classical reliability metrics evaluate whether explanations are consistent and aligned with intrinsic model signals.

Table 5.2: Classical reliability metrics (mean across folds).

Metric	**General-izability**	**Concordance**	**Stability**	**Reliability Index**
Value	0.9866	0.9154	0.7788	0.9468

5.3 Behavior-Grounded Reliability

Behavior-grounded reliability evaluates whether explanations faithfully reflect the model's sensitivity and probability honesty.

Table 5.3: Behavior-grounded reliability metrics (held-out folds).

Metric	**GLR (mean)**	**Hit@10**	**ECE**	**Brier (mean)**	**Reliability**
Value	0.0742	1.0	0.1098	0.1322	0.8091

5.4 Synthesis and Discussion

- **Prediction quality.** The model achieves AUC ≈ 0.73 but a modest F1-score due to class imbalance. Brier indicates moderate probabilistic error.
- **Classical reliability.** Generalizability and concordance are very high, but stability is lower, implying attribution drift across folds.
- **Behavior-grounded checks.** GLR is positive but small, Hit@10 shows perfect overlap at $k = 10$, and ECE reveals moderate miscalibration.
- **Composite trust.** The composite reliability score (≈ 0.81) suggests explanations are fairly reliable but can be improved through calibration and audits of GLR-negative instances.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This dissertation presented a **Reliable Explainable AI Framework for Software Defect Prediction**, addressing the limitations of traditional approaches that focus solely on interpretability without validating the trustworthiness of explanations. While existing work in defect prediction has extensively used ML/DL models and XAI tools such as SHAP and LIME, the reliability of those explanations has often been overlooked.

Our framework combined a Random Forest classifier with SHAP explanations, further extended by both classical and behavior-grounded reliability checks. Classical metrics such as *Generalizability*, *Concordance*, and *Stability* were used to benchmark explanation consistency, while novel behavior-grounded measures — **Gradient-of-Loss Reliability (GLR)**, **ε-Perturbation Hit@k**, and **Expected Calibration Error (ECE)** — validated whether SHAP explanations align with model sensitivity and probability reliability.

Empirical evaluation on the NASA PROMISE CM1 dataset demonstrated that although SHAP explanations offered good interpretability, their reliability varied. For example, generalizability and concordance scores were high, but stability dropped across folds, reflecting inconsistency. The proposed GLR and ε-Hit@k tests revealed that some SHAP-ranked features did not always influence model loss significantly, exposing gaps in explanation trustworthiness. However, the composite reliability score showed that integrating GLR, Hit@k, and ECE offers a more faithful measure of trustworthy interpretability.

Thus, the key achievement of this research is shifting the paradigm from "explainable AI" to "*reliably explainable AI*," ensuring that explanations are not only intuitive but also consistent, stable, and behaviorally grounded. This makes the framework a practical decision-support tool for software engineers and project managers, strengthening confidence in AI-driven defect

prediction systems.

6.2 Implications of Findings

The outcomes of this research have several important implications:

- **For Developers:** Reliability checks allow practitioners to distinguish between explanations that are faithful to model behavior and those that are misleading, improving defect triaging and debugging.
- **For Researchers:** The proposed reliability metrics extend beyond conventional interpretability, providing a foundation for future trustworthy XAI frameworks.
- **For Stakeholders:** Trustworthy explanations reduce the risks of relying on AI models in high-stakes software projects, ensuring that defect predictions align with causal reasoning rather than spurious correlations.

6.3 Limitations

Despite promising contributions, this work has a few limitations:

- The experiments were limited to the CM1 dataset. While it is a widely used benchmark, extending to larger and more diverse datasets would strengthen the generalizability of findings.
- Reliability metrics such as GLR and Hit@k rely on perturbation-based sensitivity analysis, which can be computationally expensive for larger datasets or deep learning models.
- Explanations were assessed primarily using SHAP; other interpretability frameworks (e.g., LIME, Integrated Gradients) were not systematically evaluated for reliability.

6.4 Future Work (Next 2 Months)

To validate that the reliability audit is *model-agnostic*, the next phase will apply the same pipeline across both classical ML and modern DL models highlighted in recent SDP/XAI studies. The emphasis is on keeping preprocessing, evaluation splits, explainers, and reliability metrics identical so that scores are comparable.

Month 1 — Classical ML Baselines and Calibration

- **Models:** Logistic Regression (class-weighted), SVM (RBF, probability-calibrated), Random Forest, and Gradient-Boosting (XGBoost/LightGBM).
- **Preprocessing:** Standardization for LR/SVM; SMOTE (train-fold only) where applicable; consistent stratified k-folds and fixed seeds.
- **Explanations:** SHAP variants (TreeSHAP for trees, KernelSHAP for LR/SVM), LIME (tabular), and permutation importance.
- **Reliability metrics:**
  1. **G (Generalizability)** — rank correlation of top-k explanations across folds,
  2. **C (Concordance)** — agreement between explainer importances and model-intrinsic signals (e.g., impurity/permutation for trees, coefficient magnitudes for LR),
  3. **S (Stability)** — attribution drift for near-neighbor inputs,
  4. **GLR** — loss-sensitivity alignment (analytic gradient for LR; finite-difference for non-smooth models),
  5. **ε-Hit@k** — do explainer top-k features induce the largest Δloss under small nudges?,
  6. **ECE/Brier** — probability calibration (pre/post Platt or isotonic).
- **Deliverables:** (i) Reliability tables (mean$\pm$sd) per model$\times$explainer, (ii) significance tests (Wilcoxon + Cliff's δ) for G/C/S and behavior-grounded metrics, (iii) calibration curves and an ablation showing the impact of calibration on reliability scores, (iv) a unified "composite reliability" leaderboard.

Month 2 — Deep and Self-Explaining Models

- **Models:**
  1. **LCNN/1D-CNN** on tabular metric vectors (lightweight CNN baselines as in recent SDP studies),
  2. **MLP** (strong tabular baseline with dropout + batch-norm),
  3. **(X)NAM / XNAM**—neural additive models with per-feature shape functions and optional pairwise interactions.
- **Explanations:** Integrated Gradients, GradientSHAP/DeepSHAP for neural models; native shape-function visualizations and gradients for (X)NAM/XNAM.

- **Reliability reuse:** Apply the same G/C/S, GLR, ε-Hit@k, and ECE/Brier protocols unchanged to enable fair comparison with Month 1 models.
- **Cross-model consistency:** Quantify agreement of explanation rankings across model families (trees vs. LR/SVM vs. DL) using Spearman/Kendall-τ, Jaccard@k, and overlap stability across folds.
- **Cross-dataset check (if time permits):** Repeat a reduced grid on an additional SDP dataset (e.g., another PROMISE/industry set) to test portability of reliability scores.
- **Deliverables:** (i) Reliability heatmaps across {model}×{explainer}×{metric}, (ii) case studies showing where explainers disagree but GLR/Δloss reveals the faithful driver, (iii) an updated composite leaderboard and guidance on model–explainer pairs that are both accurate and reliable.

Outcome. By holding the audit constant and varying the hypothesis class (LR/SVM/trees/boosting vs. LCNN/MLP/XNAM), we will test whether the proposed reliability framework generalizes beyond Random Forests and identify robust, ready-to-use model–explainer configurations for SDP.

6.5 Final Remarks

This work contributes to the emerging vision of **trustworthy XAI in software engineering**. By embedding reliability into explainability, it moves beyond black-box predictions and toward actionable, dependable insights. As AI continues to shape software development lifecycles, frameworks that prioritize both interpretability and reliability will form the backbone of next-generation defect prediction systems, ensuring not only accurate but also trustworthy decision support.

References

- F. Ketata, Z. Al Masry, S. Yacoub, and N. Zerhouni, "A Methodology for Reliability Analysis of Explainable Machine Learning: Application to Endocrinology Diseases," *IEEE Access*, 2024.

  doi:10.1109/ACCESS.2024.3431691.

- M. Begum, M. H. Shuvo, M. K. Nasir, A. Hossain, M. J. Hossain, I. Ashraf, J. Uddin, and M. A. Samad, "LCNN: Lightweight CNN Architecture for Software Defect Feature Identification Using Explainable AI," *IEEE Access*, 2024. doi:10.1109/ACCESS.2024.3388489.

- A. Lin, T. Fang, J. Yao, and B. Xu, "Understanding Software Defect Prediction Through eXplainable Neural Additive Models," *arXiv preprint*, arXiv:2306.08655, 2023.

- Y. Zhang, Y. Zhang, Y. Huang, and G. Chen, "Software Defects Identification: Results Using Machine Learning and Explainable Artificial Intelligence Techniques," *Journal of Systems and Software Engineering*, 2024.

- J. Yao, A. Lin, G. Chen, and T. Fang, "Towards Trustworthy Explainable AI for Software Defect Prediction," in *Proc. IEEE International Conference on Software Engineering (ICSE)*, 2024.

- S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 4765–4774, 2017.

- A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, et al., "Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges Toward Responsible AI," *Information Fusion*, vol. 58, pp. 82–115, 2020.