Predicting Bug-inducing Tickets

The Impact of Temporal Proximity

Daniele La Prova (Candidato)
Chiarissimo Prof. Davide Falessi (Relatore)
Ing. Emanuele Gentili (Correlatore)

21/02/2025

TOR VERGATA
UNIVERSITY OF ROME

Roadmap

Why Bother?

- Modern society heavily relies on **software**, permeating all aspects of our lives.
- When software fails, the **costs** can be immense.
- The later a **bug** is found, the more expensive its aftermaths are.

Figure 1: Some critical services that rely on software.

Why Bother?

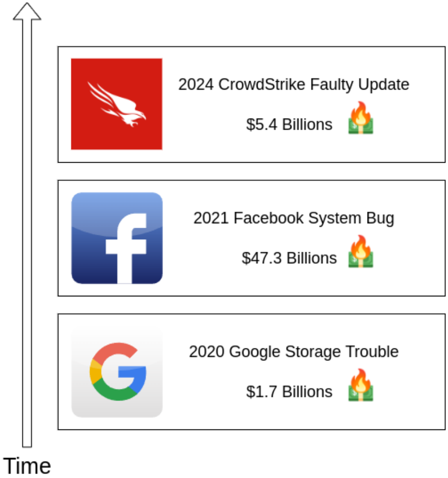


Figure 2: Most recent software outages caused by bugs, resulting in costly losses.

What is Bug Prediction?

- Given a set of entities composing the project, **bug prediction** aims to identify those that are more likely to contain bugs.
- Testing efforts focus on **predicted buggy entities**
- State-of-the-art bug prediction techniques are focused on **classes, methods, LOCs, files or commits**.
- However, these predicted entities **already contain bugs**.

What is our Aim?

- For the first time to the best of our knowledge, we predict bugs **before they have been injected**.
- With the idea that prevention is better than cure, our aim is to propose and evaluate a first approach for **ticket-level prediction (TLP)**;
- Our contribution is to **define, measure and evaluate 62 features** for a new task named **TLP**.

What is a Ticket?

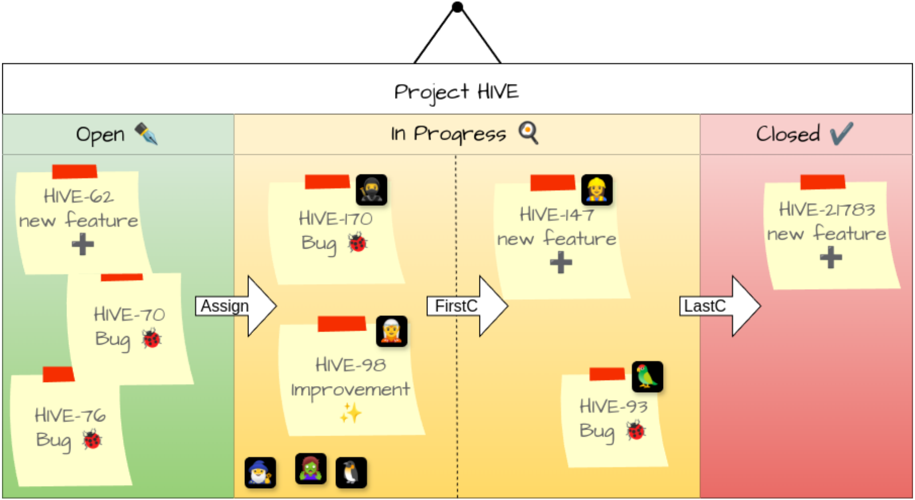


Figure 3: Ticket lifecycle. Developers use tickets to track project work.

What is a Ticket?

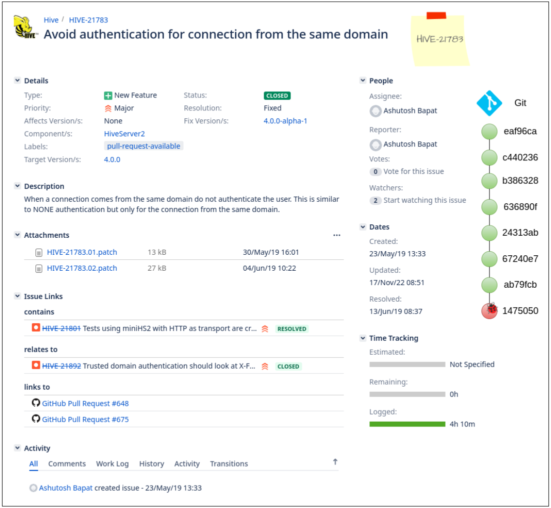


Figure 4: Ticket Example

What is a Ticket?

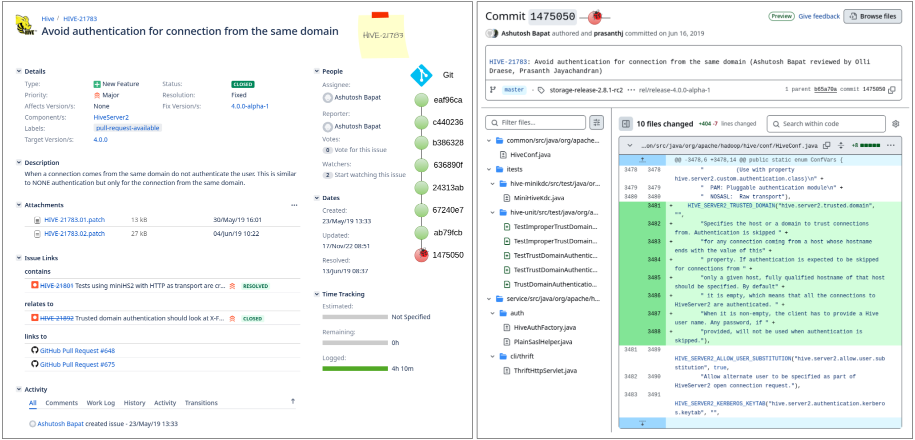


Figure 5: Ticket Example vs Commit Example. We focus on the Tickets.

When do we make prediction?

- The earlier, the better.

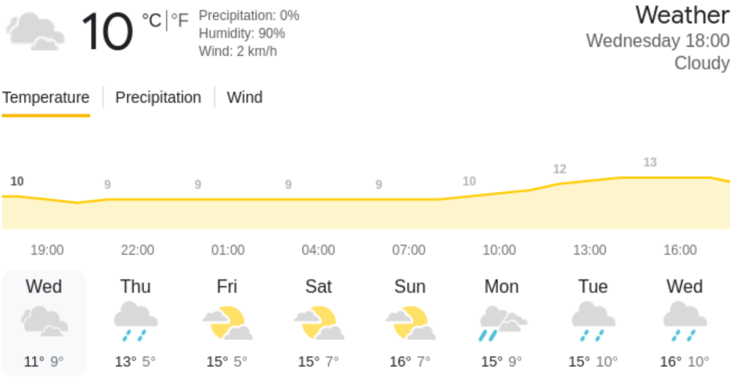
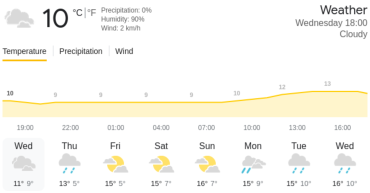


Figure 6: Weather forecast

When do we make prediction?



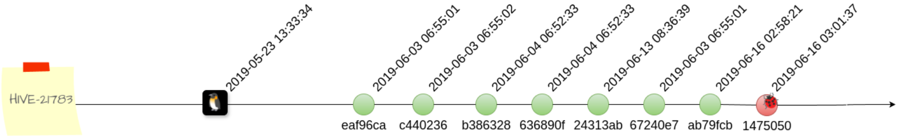
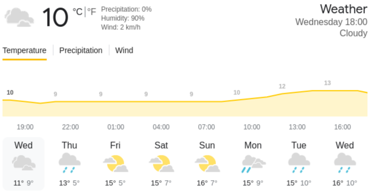


Figure 7: Ticket timeline

When do we make prediction?



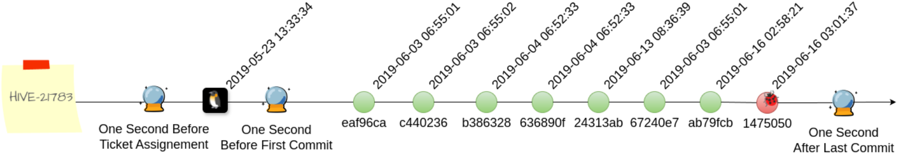


Figure 7: Ticket timeline with Measurement Dates.

Roadmap

Research Questions

RQ1

Does temporal proximity impact the **accuracy** of TLP?

RQ2

Does temporal proximity impact the predictive **power** of TLP features?

Measurement Procedure

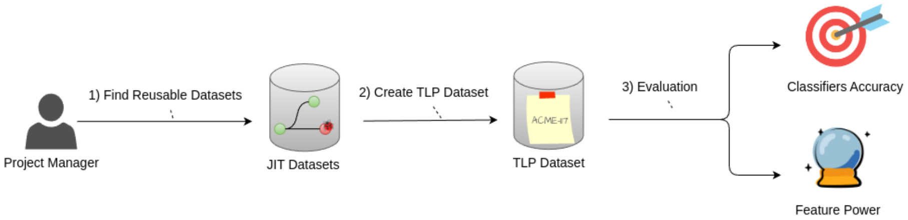


Figure 8: Phases overview

Features

- Since TLP is an innovative approach implemented by no one before, we had to **define, measure and evaluate the features** to feed the models.
- Leveraging **SE principles**, we propose and measure **62 features** belonging to **7 families**:
  - **Code**: 4 features
  - **Developer**: 2 features
  - **External Temperature**: 6 features
  - **Internal Temperature**: 10 features
  - **Intrinsic**: 22 features
  - **Requirement to Requirements**: 3 features
  - **JIT**: 15 features

Code

- The **code base** on which a ticket is implemented impacts the bugginess of the ticket
- The same ticket could lead to a bug according to how easy the code base is to accept its implementation.
- 4 features:
  - **Quality**: Number of code smells in the code base.

Developer

- We take into account the **assigned developer** to the ticket.
- SE gives to the human factor a crucial role in the ticket implementation process.
- 2 features:
  - **Familiarity**: How many tickets have been historically assigned to the developer divided by the total number of project tickets.

External Temperature

- The family takes into account **how often the project is subject to changes.**
- Implementing a ticket in an ever-changing environment can be hard.
- 6 features:
  - **Temporal Locality**: The proportion of bug-inducing tickets among all tickets prior to the measured ticket in a limited time horizon.

Internal Temperature

- We measure when, how and **how often the ticket was changed**.
- "Hot" tickets can be suspected to be problematic at least.
- 10 features:
  - **Comments count**: Ticket participants use comments to express their opinions, ask for clarifications, provide additional information, etc.

Intrinsic

- We measure the **ticket intrinsic characteristics**.
- Intuitively, SE practitioners consider some tickets **inherently more difficult** to implement than others.
- 22 features:
    - **Priority**: A level of importance telling what ticket should be implemented first.
    - **Type**: i.e: bug, improvement, new feature, subtask, etc.

R2R

- We measure the similarity between the ticket and the previous tickets that induced a bug.
- It is intuitive that tickets **semantically similar** to tickets that induced a bug are more prone to induce a bug.
- 3 features:
  - **Levenshtein Max Title**: Max Levenshtein distance calculated on Title.

JIT

- We consider the **aggregated features of the commits** linked to the ticket when they are available according to the measurement date.
- Since commits can contain the bug, they have been consistently studied in the SE domain.
- 15 features:
  - **Sum LOCs Added**
  - **Number of Commits**

Features Measurement

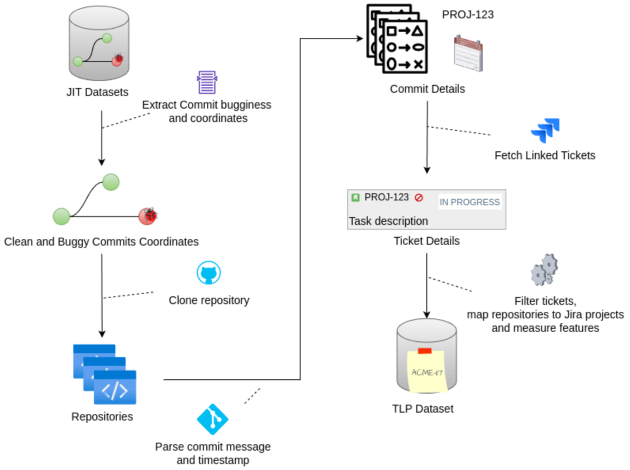


Figure 9: TLP dataset creation overview.

Features Measurement

- We selected the projects **HIVE** and **HBASE** since they have the highest buggy linkage while having lots of usable tickets.
  - Other project had too much noise in data, which would have made the evaluation less reliable.
  - A priori, there is no evidence suggesting that HIVE and HBASE make TLP more effective than other projects.
- We analyzed $\sim \mathbf{11\,000}$ **tickets** in total

RQ1: Does temporal proximity impact the accuracy of TLP?

- Independent variables:
  - Temporal Proximity (Open, InProgress, Closed).
- Dependent variables:
  - Accuracy metrics (**AUC**, Precision, Recall, Kappa, Specificity, GMean)
- H10: Temporal proximity does not impact TLP accuracy.
- Models: Random Forest (RF), Logistic Regression (LR), Neural Network (NN)

RQ2: How do TLP features perform across different temporal points?

- Independent variables:
  - Temporal Proximity, Features.
- Dependent variables:
  - **IGR**, Backward FS result (selected, not selected).
- H20: The power of TLP features does not vary across feature family, temporal points, and their combination.
- Models: same as RQ1.

Validation Technique

- **Moving Window**:
  - Addresses Concept Drift.
  - Feature Selection: FS and No FS.
  - Balancing: SMOTE and no SMOTE.

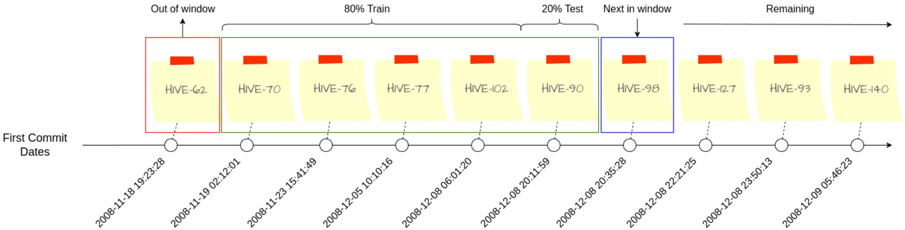


Figure 10: Sliding Window example using the first commit date as measurement date.

Roadmap

RQ1: Temporal Proximity impacts Accuracy

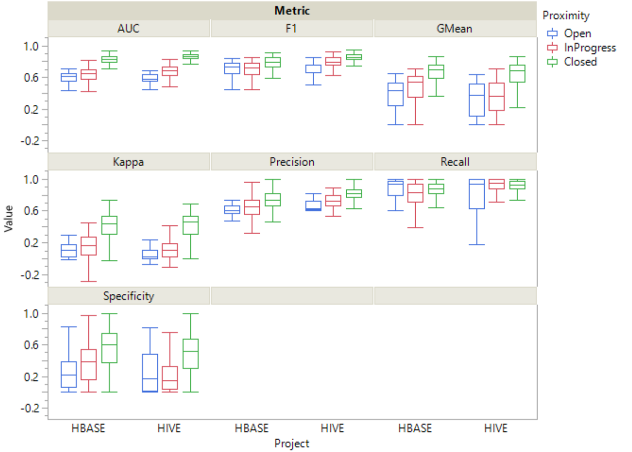


Figure 11: Distributions of TLP accuracy using moving-window in three proximity points.

RQ1: Temporal Proximity impacts Accuracy

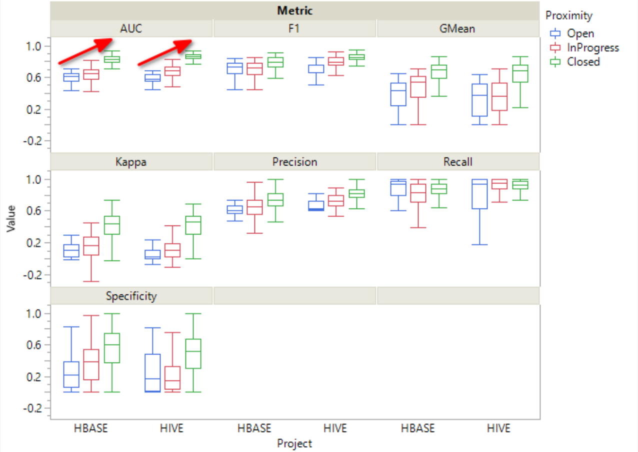


Figure 11: Distributions of TLP accuracy using moving-window in three proximity points.

RQ1: Temporal Proximity impacts Accuracy

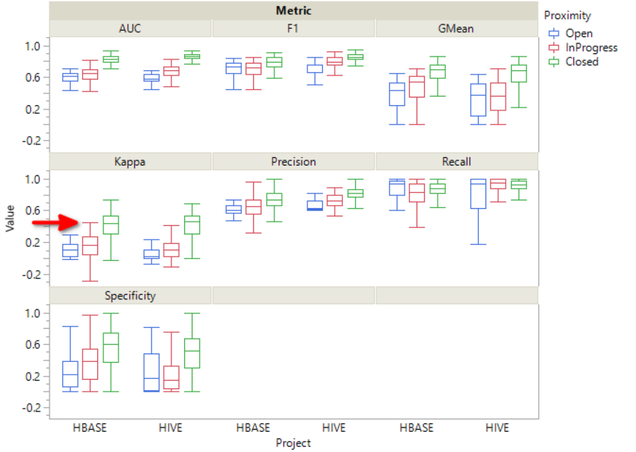


Figure 11: Distributions of TLP accuracy using moving-window in three proximity points.

RQ1: Temporal Proximity impacts Accuracy

Table 1: Average gain across classifiers in TLP accuracy using moving-window in HBASE when increasing the proximity.

	HBASE						
	Precision	**Recall**	**F1**	**AUC**	**GMean**	**Specificity**	**Kappa**
OpenToInProgress	4%	-7%	-2%	6%	44%	45%	22%
InProgressToClosed	15%	8%	13%	29%	154%	49%	42%

Table 2: Average gain across classifiers in TLP accuracy using moving-window in HIVE when increasing the proximity.

	HIVE						
	Precision	**Recall**	**F1**	**AUC**	**GMean**	**Specificity**	**Kappa**
OpenToInProgress	9%	13%	14%	17%	120%	-18%	11%
InProgressToClosed	12%	0%	6%	27%	234%	137%	84%

RQ2: Temporal Proximity impacts Feature Power

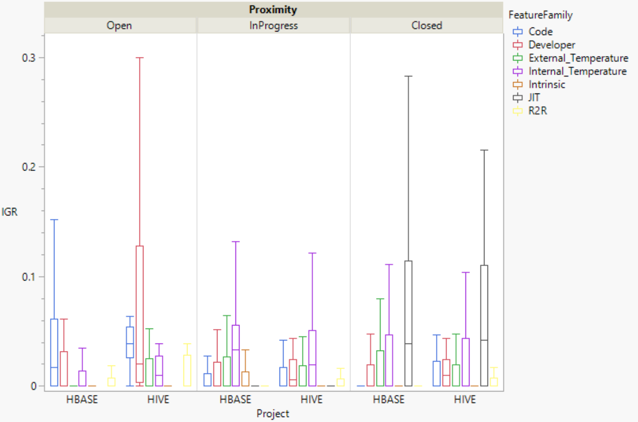


Figure 12: Distributions of feature family power, in terms of IGR, across different proximity points, in specific projects.

RQ2: Temporal Proximity impacts Feature Power

Table 3: Statistical test comparison on the impact on IGR of Feature Family, Proximity and their interaction using moving-window.

Independent Variable	Pvalue HBASE	Pvalue HIVE
FeatureFamily	0.0001	0.0001
Proximity	0.0001	0.0001
Proximity × FeatureFamily	0.0001	0.0001

- Most powerful features include:
  - # **Participants**, # **Parallel Commits** when in **Open**;
  - # **Activities** when **InProgress**;
  - **JIT** and # **languages** when in **Closed**.

Roadmap

Conclusion

- This work aimed to leverage SE principles in order to **define, measure and evaluate 62 features** for a new task named **TLP**.

RQ1: Does temporal proximity impact the accuracy of TLP?

- **TLP accuracy improves** as proximity to the ticket closing event increases.
- Practitioners should favor a **Moving Window strategy** when implementing TLP.

RQ2: How do TLP features perform across different temporal points?

- Predictive power of TLP features changes according to the **Family, the Temporal Proximity, and their combination**.
- Prediction models should **dynamically adapt** feature selection based on the proximity stage.

Future Work

- Expanding the dataset scope (more projects, both open and proprietary);
- Explore additional feature families;
- Investigate how TLP insights can be leveraged to teach better **bug prevention strategies**.

Thank you!

- **More details** can be found in the **Thesis**.
  - Other validation techniques, statistical tests, complete feature set, and more.
- We are going to publish this work in a **journal paper**.
- Any questions?

Made with ♥ and ☕ (lots of).