

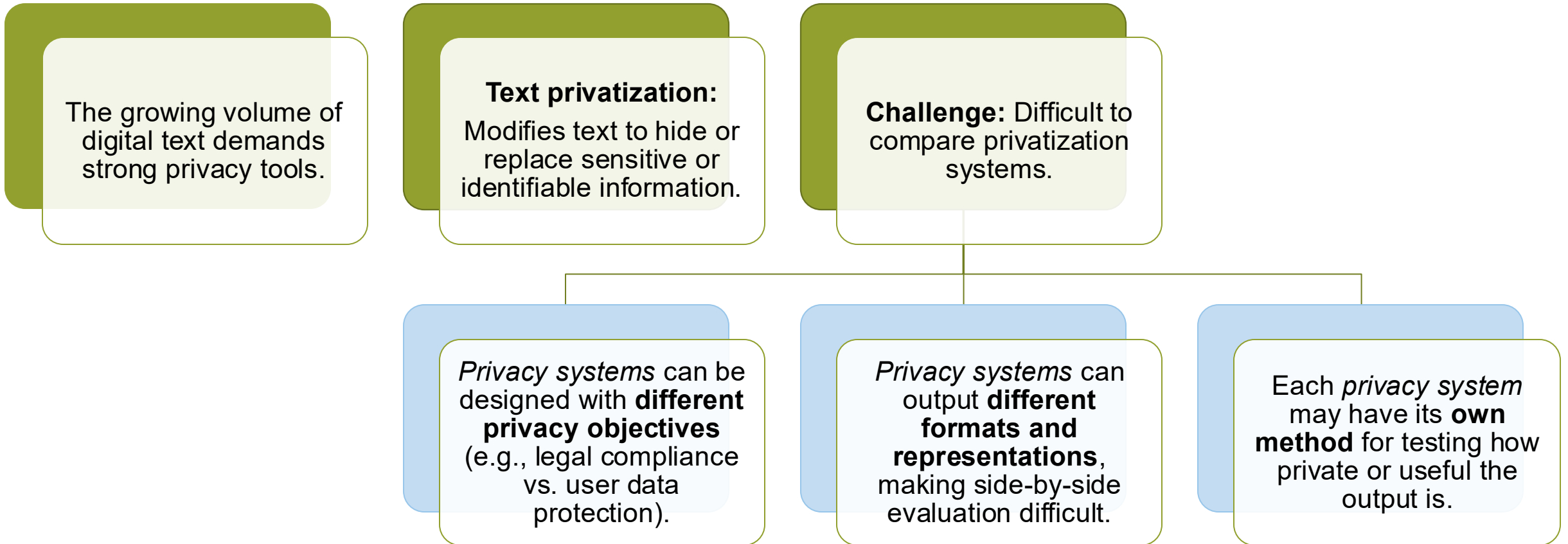
Improving the Functionality and Performance of a Text Privatization Benchmarking Platform

Ahmet Bilal Akın

23.06.2025, Master Thesis Kickoff

Chair of Software Engineering for Business Information Systems (sebis)
Department of Computer Science
School of Computation, Information and Technology (CIT)
Technical University of Munich (TUM)
www.matthes.in.tum.de

1. Introduction
2. Research Questions
3. Methodology
4. Preliminary Findings
5. Next Steps: Prioritized Enhancements & Focus Areas
6. Expected Outcomes & Challenges
7. Timeline



Task:

- An evaluation that checks how effectively a privacy model protects and maintains the quality of privatized data on a specific criterion (e.g., coherence, similarity).
- Tasks are executed in isolated environments and contribute to the overall benchmark score.

Module:

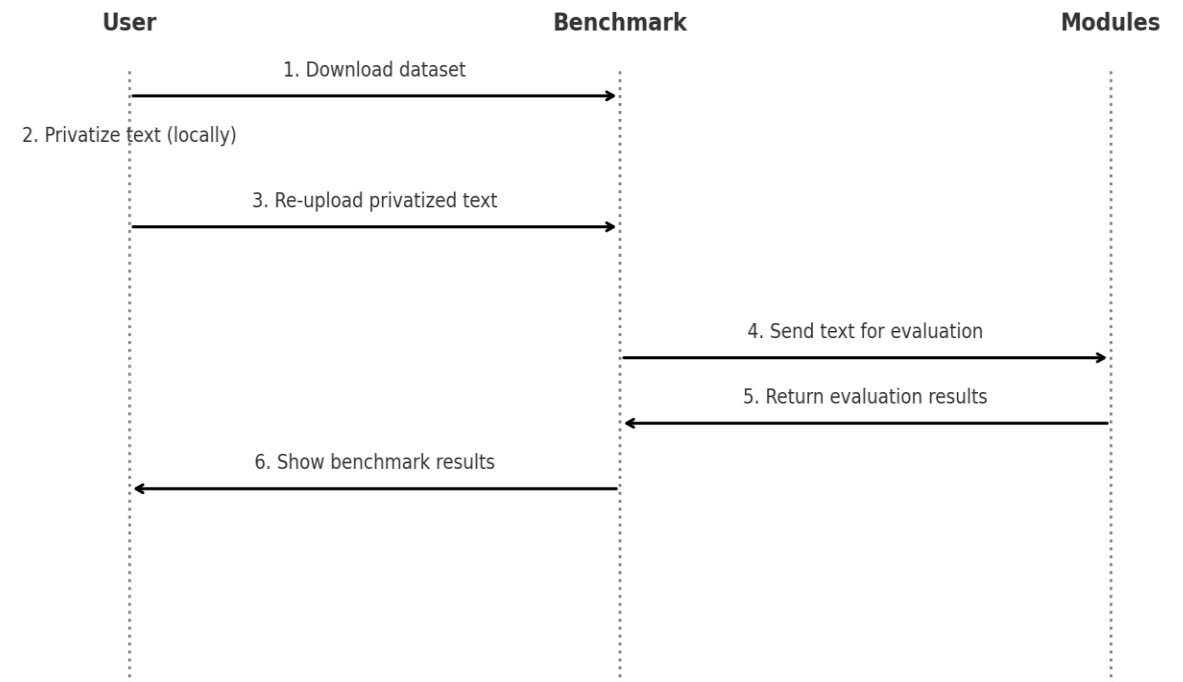
- A functional unit that defines a specific type of task, including input data, evaluation logic, and scoring method.
- Each module targets a distinct aspect of privatized text and can be executed independently during benchmarking.

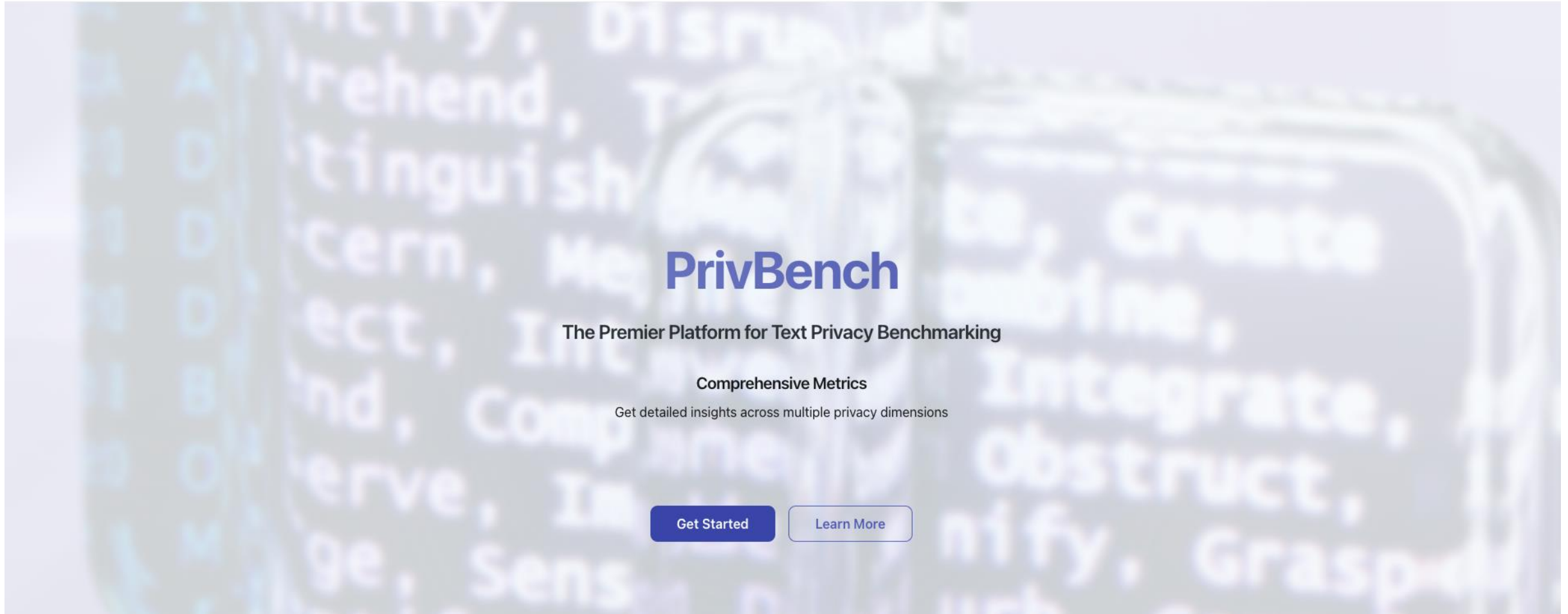
Introduction: Current State of the Benchmark

The current benchmark system provides:

- A **ranking page** to compare submissions
- A **submission interface** for users to upload privatized text
- A set of **implemented evaluation modules**, including:
 - NER Evaluation
 - Coherence
 - NearestNeighbor
 - Similarity
 - MaskedTokenInference

Sequence Diagram: Text Privatization Benchmark Workflow





RQ1: How can an existing modular evaluation framework for text privatization be extended to enable the standardized, reproducible, and interoperable measurement of privacy, utility, and performance?

- How can modules be refined to improve overall evaluation?
- What should each module measure (e.g., privacy, utility)?
- How to standardize these measurements?

RQ2: How can the platform be designed and engineered to fulfill key software quality attributes such as performance, maintainability, extensibility, and fairness in the benchmarking process?

- How to enhance performance, maintainability, and extensibility?
- How to manage versioning for evolving systems?
- What technical/conceptual needs ensure robustness & fairness?

1. Existing System Analysis:

- Evaluate the performance & limitations of modules
- Analyze versioning and its future compatibility
- Measure efficiency: time, scalability, resource use

2. Proposed Enhancements:

- **Module optimization:** Improve accuracy and utility retention.
- **System performance:** Refactor code, reduce runtime.
- **Versioning system:** Add extensibility & result consistency.

3. Testing and Validation:

- Use a shared testing dataset.
- **User workflow:** download → privatize → upload
- Run refined evaluation modules.
- Perform stress testing to evaluate system stability and performance under different load conditions.

Preliminary Findings: Core Challenges & User Experience

Insufficient user control over task execution:

- The current system lacks mechanisms for users to cancel pending or running evaluation tasks.
- The users cannot re-run individual failed tasks.
 - Impacts system robustness and user experience.

Efficiency improvements needed:

- The current approach is Docker-based, and there is a need to explore other approaches to make the application more efficient.
 - Reduce task duration and optimize resource use.

User-friendliness improvements:

- Minor UI bugs are present on certain pages. This can affect user interaction and experience.

Next Steps: Prioritized Enhancements & Focus Areas

High Priority: Enhanced Task Management

- Implement “Cancel” option for pending or running tasks.
→ Improves user control and responsiveness.
- Enable re-running of individual failed tasks.
- **Supports RQ2:** performance, maintainability, and robustness.

High Priority: Module Execution & Platform Optimization

- Evaluate alternative module execution strategies -> reduce task duration.
- Implement a versioning system.
→ Consistent benchmarking results over time.
- Integrate new evaluation modules, i.e., refine and extend the modular framework.

Lower Priority: Maintenance & Minor Fixes

Next Steps: Backlog



is:issue state:open assignee:abilalakin ✕ 🔍 🏷 Labels 📅 Milestones New issue

Open 7 **Closed** 0 Author ▾ Labels ▾ Projects ▾ Milestones ▾ Assignees ▾ ↕ Newest ▾

- Create Account Page - Display error if passwords are not matching** frontend priority: low
#74 · abilalakin opened 2 days ago
- Evaluate Alternative Module Execution Strategies for Improved Efficiency** backend priority: high
#73 · abilalakin opened 3 weeks ago · 📅 Backlog
- Tasks Stay Pending for Too Long** backend priority: high
#72 · abilalakin opened 3 weeks ago · 📅 Backlog
- Add Cancel Option to Stop Running Tasks** backend frontend priority: medium
#71 · abilalakin opened 3 weeks ago · 📅 Backlog
- Rename Frontend Files to .jsx** frontend priority: low
#70 · abilalakin opened 3 weeks ago · 📅 Backlog
- Dataset Download Status Not Persisted** backend frontend priority: medium
#69 · abilalakin opened 3 weeks ago · 📅 Backlog
- Improve UX on Privatization Method Page** frontend priority: low
#68 · abilalakin opened 3 weeks ago · 📅 Backlog

Expected Outcomes and Challenges

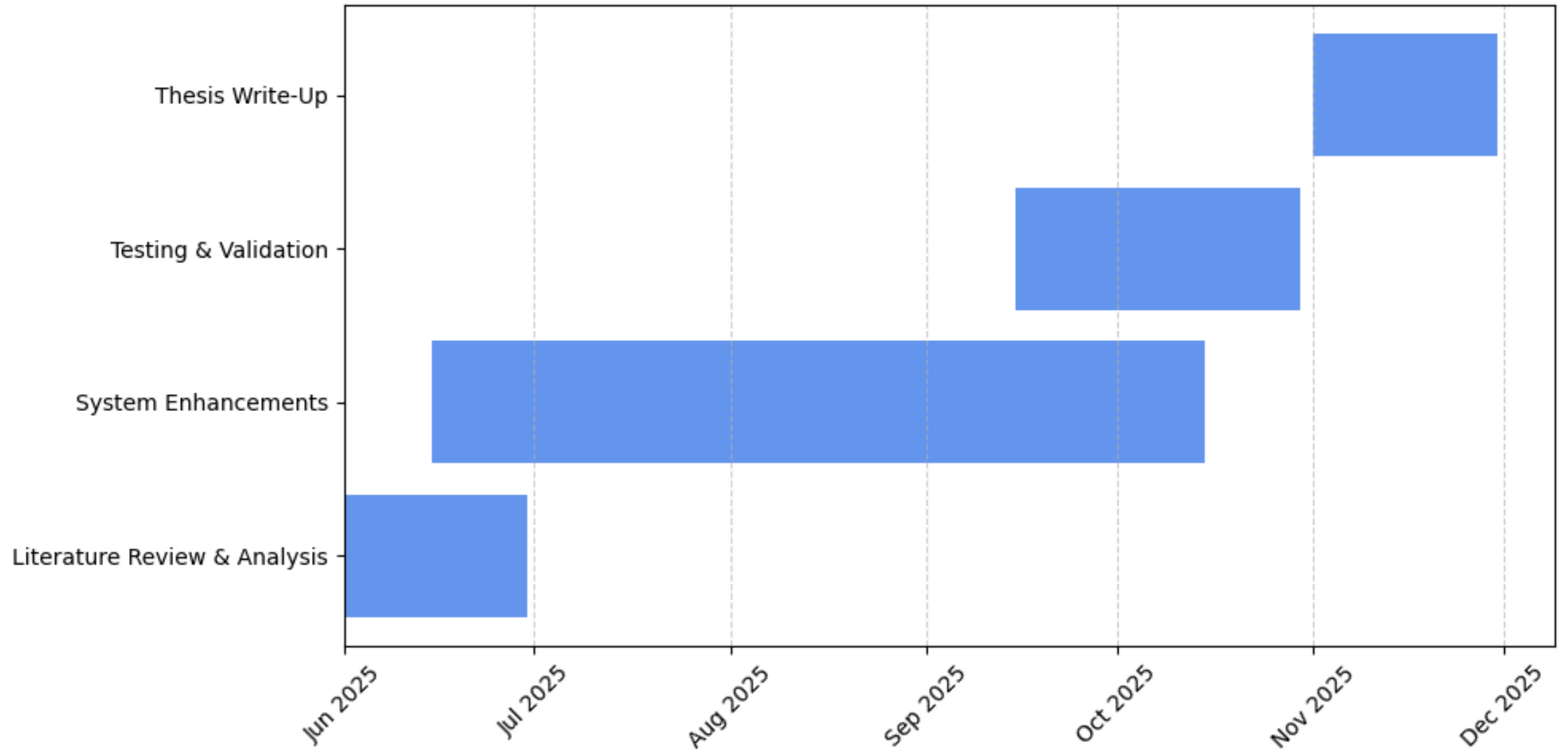
Outcomes:

- Enhanced benchmark: accurate, modular, efficient.
- Practical use for developers and researchers.
- Fair comparison across privacy systems.
- A deployable benchmark system that is ready for real-world use.

Challenges:

- Balancing privacy and utility within modules.
- Ensuring reproducibility.
- Designing modular, extensible tools.

Timeline





Ahmet Bilal Akın

bilal.akn@tum.de

Technical University of Munich (TUM)
TUM School of CIT
Department of Computer Science (CS)
Chair of Software Engineering for Business
Information Systems (sebis)

Boltzmannstraße 3
85748 Garching bei München

+49.89.289.

www.matthes.in.tum.de

