

FEMP-X: A Witness-Augmented Dataset for Candidate Functionally Equivalent Java Method Pairs

Yoshiki Higo
The University of Osaka

Abstract—Datasets of functionally equivalent code are useful for clone detection, semantic code search, test generation, and behavioral program analysis. This paper presents FEMP-X, a witness-augmented dataset built from 13,710 candidate functionally equivalent Java method pairs derived from a prior study. The main novelty of FEMP-X is that it releases reusable execution-validated behavioral evidence for falsified candidate pairs, rather than merely increasing the number of pair-level labels. Instead of attempting to prove equivalence, FEMP-X adds execution-validated counterexamples for pairs whose candidate equivalence can be falsified. Its witness-backed layer was constructed with GPT-5.4 and GPT-5.5 through Codex in an agentic workflow that generates candidate tests and immediately executes them; only reproduced behavioral differences are retained. Accordingly, each stored witness exposes a concrete difference in return values, side effects, exception behavior, or termination behavior. FEMP-X contains execution-validated witnesses for 848 of 852 gold inequivalent pairs, adds 153 manually confirmed witnesses within the original human-equivalent subset, and contributes 4,626 witness-backed pairs from the previously uninspected portion of the candidate pool, where a random audit of 100 pairs found all sampled witnesses to be valid. The dataset, scripts, and documentation are released to support research on semantic clone detection, behavioral similarity analysis, counterexample generation, and LLM-assisted dataset curation.

Zenodo: <https://zenodo.org/records/21149373>;

GitHub: <https://github.com/YoshikiHigo/FEMP-X>

Index Terms—dataset, functionally equivalent methods, counterexample generation, large language models, code clone

I. INTRODUCTION

Modern programming languages allow developers to implement the same functionality in many different ways. As a result, open-source software contains many code pairs that are behaviorally similar despite having different syntactic structures. Such pairs are useful in software engineering because they can support the evaluation of clone detection and semantic similarity techniques, as well as the analysis of alternative implementations.

A prior study introduced FEMPD, a dataset of Java method pairs collected as candidates for functional equivalence through automated test generation and mutual cross-execution [1]. Although FEMPD showed that large-scale behavioral screening can identify many promising pairs, its curation stage relied on manual inspection and therefore scaled only to a subset of the automatically extracted pool.

In this paper, we present FEMP-X, a witness-augmented dataset artifact built from that candidate pool through LLM-based counterexample generation. Rather than asking an LLM

to prove equivalence, our workflow is designed for falsification: it asks the model to find a concrete input on which two methods behave differently. When such an input is obtained and the difference is confirmed by execution, the pair can be recorded as inequivalent together with an executable witness.

This design is important for interpreting the dataset correctly. A pair for which no witness is found should not be treated as proven equivalent. It only means that the present workflow did not obtain an execution-validated witness of inequivalence for that pair. Nevertheless, if the workflow reliably rediscovers inequivalent pairs already known from prior human inspection, the remaining no-witness portion can still serve as a practically useful screened candidate pool for equivalence-oriented tasks. Accordingly, FEMP-X is not merely a larger version of FEMPD. Its main contribution is to attach reusable behavioral evidence of inequivalence to candidate pairs at scale.

The resulting artifact is useful both as a curated dataset and as a basis for downstream empirical studies. Witness-backed inequivalent pairs provide concrete distinguishing tests and reproduced outcomes, while no-witness pairs provide a practically filtered subset of the original candidate pool for tasks that require hard-to-falsify candidates.

This paper makes the following contributions:

- We present FEMP-X, a witness-augmented dataset artifact constructed over 13,710 candidate Java method pairs.
- We describe a scalable curation workflow based on LLM-generated counterexamples and execution-based validation.
- We provide an initial quality assessment showing that the workflow reliably rediscovers gold inequivalent pairs from the human-validated subset and yields useful curation outcomes for previously uninspected pairs.
- We discuss research uses of FEMP-X, including clone and semantic similarity evaluation, manual review prioritization, and exploratory behavioral analysis.

II. BACKGROUND AND RELATED WORK

A. Original Dataset and Extension Goal

The original dataset was built from Java methods extracted from IJADataset [1]. Methods were grouped by identical return types and parameter types, test cases were generated for each method, and the generated tests were executed mutually between method pairs in the same group. Pairs that succeeded in

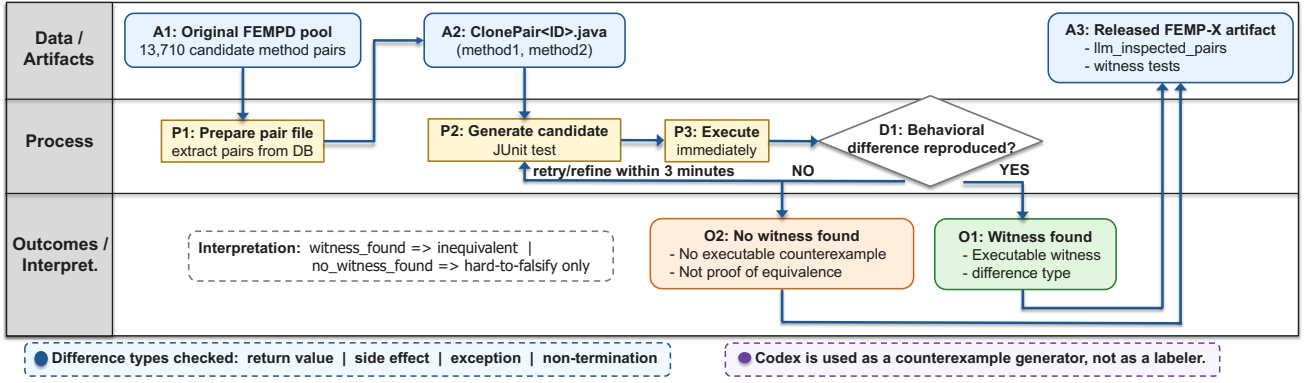


Fig. 1. Overview of the FEMP-X construction workflow. The workflow distinguishes data artifacts, processing steps, and curation outcomes. Codex is used as a counterexample generator rather than as a labeler: candidate distinguishing tests are generated, executed immediately, and retained only when execution reproduces a behavioral difference.

both directions were retained as candidate pairs for functional equivalence, and the final step consisted of manual inspection of candidate pairs.

While this pipeline was effective in producing a behavior-oriented candidate set, the last step was costly and limited scalability. In the original study, 13,710 candidate pairs were obtained automatically, but only 2,194 pairs could be inspected manually [1]. That human-validated subset contained 1,342 pairs judged equivalent and 852 pairs judged inequivalent [1]. In the original FEMPD curation, human inspectors also judged behavioral equivalence using the same four criteria adopted in the present work: return values, side effects, exception behavior, and cases in which only one method fails to terminate. All 13,710 candidate pairs in the original candidate pool are compilable and executable. This property is inherited by the present study and makes the full candidate pool suitable for direct reuse in witness generation and execution-based validation. The present work addresses this limitation by performing large-scale curation on top of the original candidate pool. Our goal is not to redefine functional equivalence, but to add reusable falsification evidence to that pool through execution-validated counterexample generation.

B. Existing Benchmarks and Positioning of FEMP-X

Several datasets have been used to evaluate clone detection and semantic code similarity. BigCloneBench is the most widely used benchmark for recall-oriented evaluation of clone detectors [2], [3]. However, recent work has cautioned that its WT3/T4 portion should be used carefully for semantic clone detection and for training machine learning models [4].

Beyond BigCloneBench, OCD provides an artificial Java benchmark generated through semantics-preserving transformations [5], Clone Oracle offers a small manually curated set of clone pairs [6], and SeSaMe provides manually assessed semantically similar Java method pairs [7]. For semantic clone detection in particular, SemanticCloneBench provides manually validated clone pairs in four programming languages [8], and GPTCloneBench extends this direction with GPT-based generation followed by manual validation [9].

Our work is most closely related to FEMPD, which provides functionally equivalent Java method pairs together with manually validated inequivalent pairs [1]. Unlike FEMPD, FEMP-X adds a reusable data layer over the original candidate pool: execution-validated witnesses of inequivalence, together with generated inputs, executable tests, and validation outcomes. For each falsified pair, FEMP-X stores an execution-validated distinguishing test, the reproduced behavioral difference, and associated outcome metadata. Thus, FEMP-X is not simply a larger collection of equivalent pairs, but a witness-augmented artifact for downstream analysis of why candidate equivalence fails and how such failures can be reproduced. Unlike coverage-oriented tools such as EvoSuite [10], our goal is to obtain one compact, execution-validated distinguishing witness per pair, rather than broad regression test suites.

III. CONSTRUCTION OF FEMP-X

This section details the construction workflow summarized in Fig. 1. The workflow starts from the original FEMPD candidate pool, materializes each target pair as a `ClonePair<ID>.java` file, and then applies a Codex-based generation–execution loop to search for an executable counterexample witness.

A. Input Candidate Set

Our starting point is the original set of 13,710 candidate pairs. All 13,710 pairs are compilable and executable in our environment. This executability was already ensured in the original FEMPD construction and is essential for the execution-based validation used in FEMP-X. Among them, 2,194 pairs were manually inspected in the prior dataset. This human-validated subset is particularly useful because it contains both pairs judged equivalent and pairs judged inequivalent. Importantly, the human-inequivalent pairs are accompanied by concrete distinguishing tests and reasons for inequivalence, which makes them suitable as a gold reference subset for assessing whether our workflow can rediscover already known counterexamples. The remaining 11,516 pairs were not manually inspected previously.

B. Witness Generation Objective

For each candidate pair materialized as a Java file, as shown in Fig. 1, the construction procedure attempts to generate a counterexample witness: an executable JUnit test that makes a behavioral difference between the two methods observable. A witness is considered valid only when execution reproduces one of the four behavioral difference types used in the original manual curation: different return values, different side effects, different exception behavior, or a case in which only one method fails to terminate.

In this work, GPT-5.4 and GPT-5.5 are used through Codex not as labelers but as counterexample generators. The models are therefore asked to search for inputs that falsify candidate equivalence, rather than to assign an equivalence label directly. During this process, the models had access to the target clone-pair file and the surrounding project/test context, but not to the original human judgment, distinguishing tests, or textual reasons for inequivalence. Accordingly, the models analyzed each pair without access to the original curation outcomes from FEMPD.

C. Operational Details

Each candidate pair was given a maximum analysis budget of three minutes, chosen as a practical trade-off between large-scale throughput and allowing iterative test refinement. As shown in Fig. 1, the Codex-based agent repeatedly generates a concrete JUnit witness test rather than a separate input generator, executes it immediately, and checks whether the execution exposes a behavioral difference between `method1` and `method2`. If the executed test does not reproduce any behavioral difference, the test is rejected and the agent retries or refines the test within the remaining budget.

The procedure terminates as soon as one executable test successfully reproduces a behavioral difference. Therefore, at most one newly generated witness test is retained per pair in the released artifact. If no such witness is found within the three-minute budget, the pair is recorded as `no_witness_found`. This outcome means only that the present workflow did not obtain an executable counterexample; it should not be interpreted as a proof of equivalence.

D. Execution-Based Validation

Execution-based validation is integrated into the generation loop rather than performed as a separate post-processing step. A generated test is stored only if it is executable and reproduces a behavioral difference using the released Gradle/JUnit test harness. Thus, the released witnesses are not natural-language judgments produced by the LLM, but executable artifacts whose outcomes were confirmed by running the two methods.

This validation step is central to the interpretation of FEMP-X. It supports treating witness-backed pairs as falsified candidate-equivalent pairs, while leaving `no_witness_found` pairs as hard-to-falsify candidates rather than as proven equivalent pairs.

E. Labeling Policy

The extended dataset distinguishes the following statuses:

- **Human-equivalent:** pairs judged equivalent in the original manual inspection.
- **Human-inequivalent:** pairs judged inequivalent in the original manual inspection and accompanied by concrete distinguishing tests.
- **witness_found/ witness-backed inequivalent:** pairs for which the construction procedure generated a witness and the witness reproduced a behavioral difference by execution.
- **no_witness_found/ no counterexample found:** pairs for which the current construction procedure did not obtain an executable witness.

Importantly, the `no_witness_found` category is not proof of equivalence; it only means that the current construction did not obtain an execution-validated witness of inequivalence. Given the workflow’s falsification goal and high rediscovery rate on the gold inequivalent subset, such pairs can still serve as screened candidates for manual review prioritization and equivalence-oriented benchmarking. They should nevertheless not be treated as ground-truth equivalent pairs without additional validation.

IV. WITNESS-AUGMENTED DATASET AND ARTIFACT DESCRIPTION

FEMP-X consists of the original candidate pool and an additional witness-backed data layer. The artifact separates the original human-validated subset, which provides manual

TABLE I
MAIN FIELDS IN THE CENTRAL ARTIFACT TABLE.

Field	Description
<code>pair_id</code>	Identifier shared with FEMPDataset. This ID can be used to trace the source code and signature information of the two methods in the original dataset.
<code>human_status</code>	Original human curation status in FEMPDataset: <code>human_equivalent</code> , <code>human_inequivalent</code> , or <code>not_manually_inspected</code> .
<code>llm_status</code>	Outcome of the present witness-generation procedure: <code>witness_found</code> if an executable witness was obtained, or <code>no_witness_found</code> otherwise.
<code>difference_type</code>	Type of behavioral difference reproduced by execution. Typical values include <code>return_value</code> , <code>side_effect</code> , <code>exception</code> , and <code>no_termination</code> . This field is null when no witness is found.
<code>pair_src</code>	Source code of the Java file containing the two methods in the pair.
<code>test_src</code>	Full source text of the corresponding generated JUnit test class.

reference labels, from the larger witness-backed layer, which contributes reusable falsification evidence at scale.

Table I summarizes the main fields of `llm_inspected_pairs`, the central artifact table. It records the original human curation status inherited from FEMPDataset, the witness-generation outcome produced in the present work, and, when available, the reproduced difference type and executable witness test. Every stored `test_src` in `llm_inspected_pairs` was generated and executed by the workflow; for witness-backed pairs, it reproduces the recorded behavioral difference.

Each record in FEMP-X contains the pair identifier, the two method bodies, the original human status when available, the witness-generation outcome, and witness information when present. For pairs confirmed as inequivalent by execution, the artifact additionally stores the distinguishing input as an executable test case that reproduces the difference. Because the construction procedure terminates when the first behavioral difference is found, the released artifact stores one witness test per witness-backed inequivalent pair. Among the witness-backed pairs, the reproduced difference types are distributed as follows: `return_value` for 3,386 pairs, `exception` for 1,327 pairs, `no_termination` for 555 pairs, and `side_effect` for 359 pairs.

Table II summarizes how the original human judgments relate to the witness-generation outcomes in FEMP-X. The table highlights three main points. First, FEMP-X rediscovers execution-validated witnesses for 848 of the 852 human-inequivalent pairs, supporting the effectiveness of the falsification workflow. Second, it finds witnesses for 153 pairs originally judged human-equivalent, showing that the prior human-equivalent subset is better interpreted as pairs for which no distinguishing input had been found, rather than as pairs proven equivalent. Third, from the previously uninspected portion of the candidate pool, FEMP-X adds 4,626 witness-backed inequivalent pairs.

Overall, FEMP-X contains 5,627 pairs with execution-validated witnesses and 8,083 pairs for which no executable counterexample was obtained. These outcomes should be interpreted asymmetrically: a witness establishes inequivalence for the exercised behavior, whereas a `no_witness_found` outcome does not prove equivalence. Nevertheless, the `no_witness_found` pairs can serve as a practically screened candidate pool for equivalence-oriented tasks.

The artifact is publicly available at:

Zenodo: <https://zenodo.org/records/21149373>
 GitHub: <https://github.com/YoshikiHigo/FEMP-X>

The Zenodo record provides the archived dataset release, while the GitHub repository provides the dataset files, generated witness tests, schema documentation, witness-generation instructions, execution scripts, and reproduction commands. In brief, Codex is instructed to create a concrete JUnit witness test that calls both methods with the same inputs and exposes a reproducible behavioral difference. This release format is intended to maximize reusability for future studies on semantic clone detection, behavioral similarity analysis, automated test generation, and LLM-assisted dataset curation. More specifically, FEMP-X makes it possible to analyze not only whether a candidate pair was falsified, but also how that falsification can be reproduced and reused in downstream experiments.

V. INITIAL QUALITY ASSESSMENT AND USE CASES

We assess FEMP-X by examining how its witness-backed inequivalence layer aligns with the human-validated subset in Table II.

A. Alignment with the Human-Validated Subset

The most reliable reference portion of the human-validated subset is the set of 852 human-inequivalent pairs. These pairs are particularly suitable for assessment because they are not merely labeled as inequivalent by human judgment; the original dataset also provides concrete reasons for inequivalence and distinguishing test cases. Therefore, they can be treated as a gold inequivalent subset for evaluating whether the present construction procedure can rediscover executable witnesses.

As shown in Table II, FEMP-X contains execution-validated distinguishing witnesses for 848 of the 852 human-inequivalent pairs, while no witness was obtained for only 4 pairs. This corresponds to a recall of 99.53% on the gold inequivalent subset. These rediscovery results are not attributable to access to the original human curation outcomes, because the model was not provided with the original labels, distinguishing tests, or textual explanations. This result indicates that the witness-generation procedure used to construct FEMP-X is highly effective as a curation aid for candidate functionally equivalent pairs, especially for identifying inequivalent pairs. As a consequence, the remaining `no_witness_found` pairs become more meaningful as a practical candidate pool for equivalence-oriented uses: although they are not proven equivalent, they have survived a falsification workflow that successfully rediscovered almost all known inequivalent pairs.

Table II also shows that FEMP-X contains executable witnesses for 153 of the 1,342 pairs originally judged equivalent by human inspection. These cases should not immediately be interpreted as simple errors in the original dataset, but as newly exposed behavioral divergences that were not triggered during the original curation process. Moreover, we manually inspected all 153 witness tests and confirmed them valid under the same four criteria used in the original curation, showing that FEMP-X can uncover overlooked inequivalent pairs and complement manual curation in practice.

TABLE II
 RELATIONSHIP BETWEEN THE ORIGINAL HUMAN JUDGMENTS AND THE CURATION OUTCOMES RECORDED IN FEMP-X.

Original judgment	Witness	No witness	Total
Human-equivalent	153	1,189	1,342
Human-inequivalent	848	4	852
Not manually inspected	4,626	6,890	11,516
Total	5,627	8,083	13,710

B. Manual Audit of Newly Curated Witnesses

The 4,626 witness-backed pairs found in the previously uninspected portion of the candidate pool are the main source of new data contributed by FEMP-X. To assess the reliability of this newly curated portion, we randomly sampled 100 of these 4,626 pairs and manually inspected the generated witness tests. All 100 sampled witnesses were valid, suggesting high label quality while remaining a sample-based assessment rather than full manual verification.

The `no_witness_found` outcome is not formal evidence of equivalence; however, given the strong rediscovery result on the gold inequivalent subset, it can still serve as a screened candidate pool for manual review prioritization and equivalence-oriented benchmarking.

C. Potential Research Uses

The released artifact supports several research directions. It can be used to benchmark semantic clone and functional-equivalence tools; for example, executing the released witnesses helps inspect false positives and distinguish label disagreement from concrete behavioral divergence. It can also be used to evaluate LLMs on program reasoning tasks that require counterexample generation rather than natural-language judgment alone. In addition, the `no_witness_found` subset can support equivalence-oriented studies as a practically screened candidate pool for manual review prioritization, exploratory positive-pool construction, and benchmarking settings where a hard-to-falsify set is preferable to the raw candidate pool. Finally, the artifact can support future research on hybrid human-LLM dataset curation workflows.

VI. LIMITATIONS AND AI USAGE DISCLOSURE

This artifact inherits several limitations from the original candidate construction process. The candidate pool is restricted to Java methods and reflects the original filtering conditions, including signature-based grouping and constraints imposed by compilation and test generation.

A second limitation is methodological asymmetry. The construction procedure is strong at falsifying equivalence when it finds an executable witness, but it does not prove equivalence when no such witness is found. The released witnesses capture execution-validated behavioral differences, including return values, side effects, exception behavior, and non-termination asymmetry, but they should still be understood as exercised behaviors rather than as an exhaustive characterization of program semantics.

A third limitation is that the construction procedure is not perfect: no new witness was obtained for four gold inequivalent pairs. Moreover, although the random audit supports the reliability of the newly added labels, the 4,626 witness-backed pairs were not exhaustively re-checked by humans and should be interpreted as sample-based evidence.

This work uses GPT-5.4 and GPT-5.5 through Codex to generate and refine candidate distinguishing tests, but only execution-validated witness tests are retained in the released

artifact; the number and kinds of witnesses may differ with other models or repeated runs.

VII. CONCLUSION

This paper presented FEMP-X, a witness-augmented dataset built from 13,710 candidate functionally equivalent Java method pairs. Its main contribution is a reusable layer of execution-validated witnesses that falsify candidate equivalence and record the corresponding behavioral differences as executable tests. The results show that FEMP-X can uncover overlooked inequivalent pairs, including 153 pairs originally judged human-equivalent, and can provide high-quality witness-backed labels at scale, as supported by the random audit of newly curated pairs. The remaining `no_witness_found` pairs should not be treated as ground-truth equivalent pairs, but they may serve as a practically screened candidate pool for equivalence-oriented analyses. We expect FEMP-X to support future work on semantic clone detection, behavioral code analysis, test generation, counterexample generation, and LLM-assisted dataset curation.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Numbers JP24H00692, JP25K03102, JP26K02889, and JP26H02500.

REFERENCES

- [1] Y. Higo, “Dataset of Functionally Equivalent Java Methods and Its Application to Evaluating Clone Detection Tools,” *IEICE Transactions on Information and Systems*, vol. E107-D, no. 6, pp. 751–760, 2024.
- [2] J. Svajlenko, J. F. Islam, I. Keivanloo, C. K. Roy, and M. M. Mia, “Towards a Big Data Curated Benchmark of Inter-Project Code Clones,” in *Proceedings of the IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2014, pp. 476–480.
- [3] J. Svajlenko and C. K. Roy, “BigCloneEval: A Clone Detection Tool Evaluation Framework with BigCloneBench,” in *Proceedings of the IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2016, pp. 596–600.
- [4] J. Krinke and C. Ragkhitwetsagul, “How the Misuse of a Dataset Harmed Semantic Clone Detection,” 2025, arXiv:2505.04311.
- [5] C. Ragkhitwetsagul, J. Krinke, and D. Clark, “A Comparison of Code Similarity Analysers,” *Empirical Software Engineering*, vol. 23, no. 4, pp. 2464–2519, 2018.
- [6] D. E. Krutz and W. Le, “A Code Clone Oracle,” in *Proceedings of the 11th Working Conference on Mining Software Repositories (MSR)*. ACM, 2014, pp. 388–391.
- [7] M. Kamp, P. Kreutzer, and M. Philippsen, “SeSaMe: A Data Set of Semantically Similar Java Methods,” in *Proceedings of the IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. IEEE, 2019, pp. 529–533.
- [8] F. Al-Omari, C. K. Roy, and T. Chen, “SemanticCloneBench: A Semantic Code Clone Benchmark Using Crowd-Source Knowledge,” in *Proceedings of the IEEE 14th International Workshop on Software Clones (IWSC)*. IEEE, 2020, pp. 57–63.
- [9] A. I. Alam, P. R. Roy, F. Al-Omari, C. K. Roy, B. Roy, and K. A. Schneider, “GPTCloneBench: A Comprehensive Benchmark of Semantic Clones and Cross-Language Clones Using GPT-3 Model and SemanticCloneBench,” in *Proceedings of the IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2023, pp. 1–13.
- [10] G. Fraser and A. Arcuri, “EvoSuite: Automatic Test Suite Generation for Object-Oriented Software,” in *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering (ESEC/FSE)*. ACM, 2011, pp. 416–419.