TESIS

DETEKSI CACAT KODE PADA BAHASA PEMROGRAMAN PYTHON MENGGUNAKAN MODEL BAHASA CODET5-SMALL BERBASIS FOCAL LOSS

Oleh:

Hanny Handayani Sucinta

24206051005

PROGRAM STUDI INFORMATIKA

PROGRAM MAGISTER FAKULTAS SAINS DAN TEKNOLOGI

UIN SUNAN KALIJAGA

YOGYAKARTA

2026

PENGESAHAN TUGAS AKHIR

KEMENTERIAN AGAMA
UNIVERSITAS ISLAM NEGERI SUNAN KALIJAGA
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Marsda Adisucipto Telp. (0274) 540971 Fax. (0274) 519739 Yogyakarta 55281

PENGESAHAN TUGAS AKHIR

Nomor : B-1099/Un.02/DST/PP.00.9/06/2026

Tugas Akhir dengan judul : DETEKSI CACAT KODE PADA BAHASA PEMROGRAMAN PYTHON MENGGUNAKAN MODEL BAHASA CODET5-SMALL BERBASIS FOCAL LOSS

yang dipersiapkan dan disusun oleh:

Nama : HANNY HANDAYANI SUCINTA, S.Kom.
Nomor Induk Mahasiswa : 24206051005
Telah diujikan pada : Senin, 18 Mei 2026
Nilai ujian Tugas Akhir : A

dinyatakan telah diterima oleh Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta

TIM UJIAN TUGAS AKHIR

Ketua Sidang

Dr. Agung Fatwanto, S.Si., M.Kom.
SIGNED

Valid ID: 6a1d3e1f28a151

Penguji I

Muhammad Mustakim, S.T. M.T.
SIGNED

Valid ID: 6a1a6e0bc0b56

Penguji II

Dr. Ir. Bambang Sugiantoro, M.T., IPU., ASEAN Eng.
SIGNED

Valid ID: 6a1ce9561d0b4

Yogyakarta, 18 Mei 2026
UIN Sunan Kalijaga
Dekan Fakultas Sains dan Teknologi

Prof. Dr. Dra. Hj. Khurul Wardati, M.Si.
SIGNED

Valid ID: 6a1e5dfdbcf30

1/1 02/06/2026

SURAT PERNYATAAN KEASLIAN

SURAT PERNYATAAN KEASLIAN TESIS

Yang bertanda tangan di bawah ini:

Nama	: Hanny Handayani Sucinta
NIM	: 24206051005
Program Studi	: Informatika
Fakultas	: Sains dan Teknologi

Menyatakan dengan sesungguhnya, bahwa tesis saya yang berjudul: **Deteksi Cacat Kode Pada Bahasa Pemrograman Python Menggunakan Model Bahasa CodeT5-Small Berbasis Focal Loss** adalah hasil karya pribadi dan sepanjang pengetahuan penyusun tidak berisi materi yang dipublikasikan atau ditulis orang lain, kecuali bagian-bagian tertentu yang penyusun ambil sebagai acuan.

Apabila terbukti pernyataan ini tidak benar, maka sepenuhnya menjadi tanggungjawab penyusun.

Yogyakarta, 08 Mei 2026

Yang menyatakan,

Hanny Handayani Sucinta

24206051005

SURAT PERNYATAAN BEBAS PLAGIASI

SURAT PERNYATAAN BEBAS PLAGIASI

Yang bertanda tangan di bawah ini:

Nama	: Hanny Handayani Sucinta
NIM	: 24206051005
Program Studi	: Informatika
Fakultas	: Sains dan Teknologi

menyatakan bahwa naskah tesis ini secara keseluruhan benar-benar bebas dari plagiasi. Jika di kemudian hari terbukti melakukan plagiasi, maka saya siap ditindak sesuai ketentuan hukum yang berlaku

Yogyakarta, 08 Mei 2026

Yang menyatakan,

Hanny Handayani Sucinta

24206051005

NOTA DINAS PEMBIMBING

NOTA DINAS PEMBIMBING

Kepada Yth.,

Dekan Fakultas Sains dan teknologi
UIN Sunan Kalijaga
Yogyakarta

Assalamu 'alaikum wr. wb.

Disampaikan dengan hormat, setelah melakukan bimbingan, arahan, dan koreksi terhadap penulisan tesis yang berjudul:

DETEKSI CACAT KODE PADA BAHASA PEMROGRAMAN PYTHON MENGGUNAKAN MODEL BAHASA CODET5-SMALL BERBASIS FOCAL LOSS

Yang ditulis oleh :

Nama	: Hanny Handayani Sucinta
NIM	: 24206051005
Fakultas	: Sains dan Teknologi
Jenjang	: Program Studi Magister (S2)
Program Studi	: Informatika

Saya berpendapat bahwa tesis tersebut sudah dapat diajukan kepada Program Studi Magister (S2) Informatika UIN Sunan Kalijaga untuk diujikan dalam rangka memperoleh gelar Magister Komputer.

Wassalamu'alaikum wr. wb.

Yogyakarta, 07 Mei 2026

Pembimbing

Dr. Agung Fatwanto, S.Si., M.Kom.
19770103 200501 003

ABSTRAK

Deteksi cacat kode merupakan bagian penting dalam proses quality assurance perangkat lunak karena membantu mengidentifikasi potongan kode yang berpotensi mengandung kesalahan sebelum berdampak pada tahap pengujian atau produksi. Pada bahasa Python, proses ini menjadi lebih menantang karena karakteristiknya yang dinamis dan adanya ketidakseimbangan kelas pada dataset defect detection. Penelitian ini bertujuan mengembangkan model klasifikasi biner untuk mendeteksi cacat kode Python menggunakan CodeT5-small berbasis arsitektur Sequence-to-Sequence. Dataset yang digunakan berasal dari PyTraceBugs dan melalui tahapan pra-pemrosesan, deduplikasi, tokenisasi, serta pembagian data secara stratified berdasarkan label dan length-bin untuk menjaga distribusi kelas dan panjang kode. Untuk menangani class imbalance, penelitian ini mengombinasikan WeightedRandomSampler pada level distribusi data dan Focal Loss pada level fungsi kerugian. Evaluasi dilakukan menggunakan accuracy, precision, recall, F1-score, MCC, AUC, confusion matrix, baseline length-only, dan length-matched test. Hasil evaluasi pada test set menunjukkan bahwa model memperoleh accuracy sebesar 80,14%, precision kelas defective sebesar 67,89%, recall sebesar 86,51%, F1-score sebesar 76,08%, MCC sebesar 60,79%, dan AUC sebesar 83,73%. Dibandingkan baseline length-only, model utama menunjukkan peningkatan yang jelas pada seluruh metrik utama. Pada length-matched test, model tetap mempertahankan recall sebesar 83,39% dan F1-score sebesar 77,30%, sehingga menunjukkan bahwa performa model tidak semata-mata dipengaruhi oleh panjang kode. Hasil penelitian ini menunjukkan bahwa kombinasi CodeT5-small, WeightedRandomSampler, dan Focal Loss mampu meningkatkan

sensitivitas terhadap kelas defective serta memberikan evaluasi yang lebih kuat terhadap potensi bias panjang kode.

Kata kunci: deteksi cacat kode, Python, CodeT5-small, Focal Loss, WeightedRandomSampler, PyTraceBugs.

ABSTRACT

Code defect detection is an important component of software quality assurance because it helps identify potentially faulty code before such defects affect testing or production environments. In Python, this task is particularly challenging due to the dynamic nature of the language and the presence of class imbalance in defect detection datasets. This study aims to develop a binary classification model for detecting defects in Python code using CodeT5-small with a Sequence-to-Sequence architecture. The dataset was obtained from PyTraceBugs and processed through preprocessing, deduplication, tokenization, and stratified data splitting based on label and length-bin to preserve both class distribution and code-length distribution. To address class imbalance, WeightedRandomSampler was applied at the data distribution level, while Focal Loss was used at the loss function level. The model was evaluated using accuracy, precision, recall, F1-score, MCC, AUC, confusion matrix, a length-only baseline, and a length-matched test. The evaluation on the test set achieved an accuracy of 80.14%, defective-class precision of 67.89%, recall of 86.51%, F1-score of 76.08%, MCC of 60.79%, and AUC of 83.73%. Compared with the length-only baseline, the proposed model showed clear improvements across the main evaluation metrics. In the length-matched test, the model maintained a recall of 83.39% and an F1-score of 77.30%, indicating that its performance was not solely driven by code length. The findings suggest that the combination of CodeT5-small, WeightedRandomSampler, and Focal Loss improves sensitivity toward the defective class while providing a more robust evaluation against potential code-length bias.

Keywords: code defect detection, Python, CodeT5-small, Focal Loss, WeightedRandomSampler, PyTraceBugs.

MOTTO

"The goal is… being OVERLY, DISGUSTINGLY and ANNOYINGLY educated. Where I'll read every book, so I can be a human Encyclopedia."

"Qur'an [8:30][3:54]"

"…and I won't settle for the live I don't want, even though I should restart thousand times."

PERSEMBAHAN

Dengan penuh rasa syukur kepada Allah SWT atas segala rahmat, nikmat, dan kekuatan yang telah diberikan, karya tesis ini saya persembahkan kepada:

1. Kedua orang tua saya tercinta, Ayah Arhendri dan Ibu Hidayatul Fitri, yang senantiasa memberikan doa, kasih sayang, dukungan, serta pengorbanan yang tidak pernah ternilai sepanjang perjalanan hidup dan pendidikan saya. Terima kasih atas setiap doa yang menjadi kekuatan, setiap nasihat yang menjadi pegangan, dan setiap pengorbanan yang menjadi jalan bagi saya untuk sampai pada tahap ini.

2. Kepada Nenek dan Atuk tercinta, yang selalu memberikan doa, kasih sayang, nasihat, serta dukungan yang tulus. Terima kasih atas perhatian, kehangatan, dan doa-doa baik yang senantiasa mengiringi setiap langkah saya hingga sampai pada tahap ini.

3. Kepada keluarga besar saya yang selalu memberikan semangat, doa, dan dukungan dalam setiap proses yang saya lalui.

4. Kepada Muhammad Syafiq Akmal, yang telah memberikan dukungan, semangat, perhatian, dan motivasi selama proses penyusunan tesis ini. Terima kasih telah menjadi bagian dari perjalanan ini, baik dalam keadaan sulit maupun menyenangkan, serta selalu memberikan dorongan agar saya tetap kuat dan terus berusaha hingga tesis ini dapat diselesaikan.

5. Kepada dosen pembimbing, dosen penguji, serta seluruh pihak akademik yang telah memberikan ilmu, arahan, dan bimbingan selama proses penyusunan tesis ini.

6. Kepada rekan-rekan dan sahabat yang selalu memberikan bantuan, motivasi, serta kebersamaan dalam menyelesaikan studi ini.

7. Terakhir, karya ini saya persembahkan untuk diri saya sendiri, yang telah berusaha bertahan, belajar, dan menyelesaikan setiap proses dengan penuh kesabaran dan tanggung jawab.

Semoga karya ini dapat memberikan manfaat bagi pembaca dan menjadi bagian kecil dari kontribusi dalam pengembangan ilmu pengetahuan.

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah SWT atas limpahan rahmat, taufik, dan hidayah-Nya, sehingga penulis dapat menyelesaikan tesis yang berjudul "Deteksi Cacat Kode Python Menggunakan Model CodeT5-Small Berbasis Focal Loss" dengan baik.

Tesis ini disusun sebagai salah satu syarat untuk memperoleh gelar Magister pada Program Studi Magister Informatika, UIN Sunan Kalijaga Yogyakarta. Penelitian ini dilatarbelakangi oleh pentingnya proses deteksi cacat kode dalam mendukung kualitas perangkat lunak, khususnya pada bahasa pemrograman Python. Melalui penelitian ini, penulis berupaya menerapkan model CodeT5-small berbasis Focal Loss untuk mendeteksi kode Python yang bersifat clean maupun defective, serta mengkaji strategi penanganan ketidakseimbangan kelas dan potensi bias panjang kode.

Penulis menyadari bahwa penyusunan tesis ini tidak terlepas dari bantuan, bimbingan, dukungan, serta doa dari berbagai pihak. Oleh karena itu, dengan segala kerendahan hati, penulis menyampaikan terima kasih kepada:

1. Bapak Dr. Agung Fatwanto, S.Si., M.Kom., selaku dosen pembimbing yang telah memberikan arahan, bimbingan, masukan, dan motivasi selama proses penyusunan tesis ini.
2. Bapak Muhammad Mustakim, S.T., M.T., selaku Penguji 1 yang telah memberikan masukan, kritik, dan saran yang membangun demi penyempurnaan penelitian ini.

3. Bapak Dr. Ir. Bambang Sugiantoro, M.T., IPU., ASEAN Eng., selaku Penguji 2 yang telah memberikan arahan, evaluasi, serta saran yang sangat berharga bagi pengembangan tesis ini.
4. Seluruh dosen Program Studi Magister Informatika UIN Sunan Kalijaga yang telah memberikan ilmu, wawasan, dan pengalaman akademik selama masa perkuliahan.
5. Rekan-rekan mahasiswa Magister Informatika serta semua pihak yang telah membantu, baik secara langsung maupun tidak langsung, dalam proses penyelesaian tesis ini.

Penulis menyadari bahwa tesis ini masih memiliki keterbatasan dan belum sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun demi perbaikan dan pengembangan penelitian di masa mendatang.

Akhir kata, semoga tesis ini dapat memberikan manfaat bagi pengembangan ilmu pengetahuan, khususnya dalam bidang machine learning, software quality assurance, dan deteksi cacat kode berbasis deep learning.

Yogyakarta, 19 Mei 2026

Penulis,

Hanny Handayani Sucinta

NIM 24206051005

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

A. Latar Belakang

Dalam ekosistem digital saat ini, menjaga kualitas perangkat lunak merupakan hal yang sangat krusial (Laracy *et al.*, 2025). Perangkat lunak bukan hanya berfungsi sebagai alat operasional, tetapi juga menjadi penentu keandalan layanan dan keamanan data yang berpengaruh langsung terhadap daya saing bisnis(Nabot and Al-Qerem, 2025). Kepercayaan pengguna dibangun melalui pengalaman penggunaan yang stabil dan minim gangguan, sehingga deteksi cacat kode atau code defect detection perlu diposisikan sebagai bagian inti dari pengendalian kualitas untuk menjamin reliabilitas aplikasi sebelum dirilis maupun selama siklus pengembangan berlangsung(van Dinter *et al.*, 2023). Kompleksitas basis kode modern terus meningkat seiring bertambahnya fitur, dependensi, dan frekuensi perubahan, sehingga laju pertumbuhan kompleksitas tersebut semakin sulit diimbangi oleh kemampuan manusia untuk melakukan code review secara manual dan konsisten(Akimova *et al.*, 2021). Cacat yang tidak terdeteksi sejak dini dapat memicu pembengkakan biaya pemeliharaan, karena perbaikan pada tahap akhir umumnya membutuhkan analisis akar masalah, regression testing, dan penyesuaian dependensi yang jauh lebih kompleks (Albattah and Alzahrani, 2024). Oleh sebab itu, kondisi ini menuntut adanya mekanisme deteksi cacat yang lebih cerdas dan otomatis agar potensi defect dapat diidentifikasi lebih awal tanpa bergantung sepenuhnya pada peninjauan manual(Xiong *et al.*, 2025).

Python telah menjadi salah satu bahasa pemrograman utama dalam pengembangan sistem cerdas karena ekosistem pustaka machine learning

dan deep learning yang sangat matang. Namun, di sisi lain, Python memiliki karakteristik struktur yang dinamis, terutama melalui dynamic typing dan dynamic binding, sehingga perilaku program dapat berubah pada saat runtime(Tang, 2026). Karakteristik ini menyebabkan sejumlah kesalahan logika maupun tipe data menjadi tersembunyi dan sulit dideteksi oleh alat analisis statis konvensional(Mukherjee *et al.*, 2022).

Dalam praktiknya, pengembang sering terjebak dalam siklus debugging yang berulang, yaitu mereproduksi bug, menelusuri akar masalah, memperbaiki kode, lalu mengujinya kembali. Proses ini membutuhkan waktu yang tidak sedikit hanya untuk mengembalikan program ke kondisi stabil(Fan *et al.*, 2025). Secara ideal, potensi cacat pada kode seharusnya dapat dideteksi lebih awal melalui mekanisme analisis statis yang mampu memberikan early warning sebelum program dijalankan, sehingga risiko lolosnya cacat ke lingkungan produksi dapat ditekan(Bryan and Moriano, 2023). Tren riset terkini juga menunjukkan peningkatan pemanfaatan model machine learning dan deep learning untuk memprediksi komponen atau perubahan kode yang rentan cacat (defect-prone), sehingga proses quality assurance tidak lagi sepenuhnya bergantung pada inspeksi manual(Destefanis *et al.*, 2026).

Berbagai studi telah mengeksplorasi pendekatan machine learning untuk mengklasifikasikan kecacatan pada kode guna membantu proses pendeteksian secara lebih dini dan sistematis, salah satunya dengan memanfaatkan dataset PyTraceBugs. Penelitian oleh (Akimova *et al.*, 2021) memperkenalkan dataset tersebut dan mendemonstrasikan prediksi cacat menggunakan embedding CodeBERT dengan batas input 512 token yang dilatih menggunakan LightGBM, serta dapat dipadukan dengan metrik kompleksitas seperti Radon untuk meningkatkan performa.

Sementara itu, studi komparatif lain oleh (Harsh, Santoshi and Singh, 2025) memanfaatkan metrik statis dari Understand tool, menerapkan preprocessing seperti min-max scaling dan oversampling untuk mengatasi class imbalance, lalu membandingkan beberapa model klasik seperti Random Forest, KNN, Decision Tree, dan Logistic Regression. Namun demikian, penelitian terbaru oleh (Harsh, Santoshi and Singh, 2025) menunjukkan bahwa dataset tersebut memiliki tantangan besar berupa ketidakseimbangan kelas yang sangat ekstrem, di mana instansi kode cacat (buggy) hanya sekitar 0,038% dari total data sehingga model berisiko tinggi bias terhadap kelas mayoritas non-cacat. Sejalan dengan itu, meskipun pendekatan model bahasa seperti BERT telah dilakukan sebelumnya oleh (Akimova *et al.*, 2021), sensitivitas (recall) terhadap kelas kode cacat masih tergolong rendah, yaitu sekitar 34%, yang menunjukkan bahwa sistem masih melewatkan banyak potensi kesalahan pada kode.

Berdasarkan kondisi tersebut, terdapat dua persoalan penting yang menjadi celah penelitian. Pertama, ketidakseimbangan kelas yang ekstrem menyebabkan model cenderung lebih mudah memprediksi kelas mayoritas dibanding benar-benar mengenali karakteristik kode cacat. Kedua, pada tugas klasifikasi kode, model berpotensi memanfaatkan ciri-ciri sederhana seperti panjang potongan kode sebagai shortcut, bukan benar-benar memahami pola defect. Dengan demikian, penelitian ini tidak hanya perlu meningkatkan kemampuan deteksi terhadap kelas buggy, tetapi juga perlu memastikan bahwa performa model tidak semata-mata dipengaruhi oleh bias distribusi data, termasuk bias panjang kode.

Penelitian ini mengusulkan penggunaan arsitektur Sequence-to-Sequence (Seq2Seq) berbasis CodeT5-small untuk mengotomatisasi

proses klasifikasi biner pada deteksi cacat kode Python. CodeT5 dipilih karena memiliki kemampuan merepresentasikan semantik kode melalui arsitektur encoder-decoder dan proses pretraining yang peka terhadap identifier, sehingga relevan untuk tugas analisis kode sumber (Wang *et al.*, 2021).

Untuk menghadapi masalah ketidakseimbangan kelas, penelitian ini mengombinasikan CodeT5-small dengan Focal Loss dan WeightedRandomSampler agar proses pelatihan lebih menekankan sampel yang sulit serta mengurangi dominasi kelas mayoritas. Selain itu, untuk mengurangi risiko evaluasi yang bias, penelitian ini menerapkan stratified split berdasarkan label dan length-bin, menggunakan baseline berbasis panjang kode (length-only baseline), serta melakukan evaluasi pada length-matched test agar performa yang dilaporkan lebih mencerminkan kemampuan model dalam mendeteksi cacat, bukan sekadar memanfaatkan panjang kode sebagai indikator tidak langsung(Shao *et al.*, 2020).

Melalui pendekatan tersebut, penelitian ini diharapkan dapat memberikan kontribusi pada pengembangan metode deteksi cacat kode yang lebih akurat, lebih andal, dan lebih kuat terhadap bias data pada basis kode modern. Berdasarkan uraian tersebut, penelitian tesis ini berjudul "Deteksi Cacat Kode pada Bahasa Pemrograman Python Menggunakan Pendekatan CodeT5-small Berbasis Focal Loss".

B. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam penelitian ini adalah sebagai berikut:

1) Bagaimana menerapkan CodeT5-small berbasis arsitektur Seq2Seq untuk melakukan klasifikasi biner deteksi cacat kode pada dataset PyTraceBugs secara otomatis?

2) Sejauh mana Focal Loss dan WeightedRandomSampler efektif dalam mengatasi class imbalance serta meningkatkan kemampuan model dalam mendeteksi kelas buggy?
3) Bagaimana memastikan bahwa performa model benar-benar mencerminkan kemampuan deteksi cacat, bukan sekadar memanfaatkan shortcut seperti panjang kode, melalui split stratifikasi label + length-bin, baseline length-only, dan length-matched test?

C. Batasan Masalah

Agar penelitian lebih terarah, maka batasan masalah dalam penelitian ini adalah sebagai berikut:

1) Penelitian ini hanya berfokus pada klasifikasi biner deteksi cacat kode (buggy dan non-buggy) pada potongan kode Python dari dataset PyTraceBugs.
2) Model utama yang digunakan dalam penelitian ini adalah CodeT5-small dengan arsitektur Seq2Seq.
3) Penanganan ketidakseimbangan kelas dibatasi pada penggunaan Focal Loss dan WeightedRandomSampler.
4) Evaluasi bias panjang kode dibatasi pada penggunaan stratified split berbasis label + length-bin, baseline length-only, dan length-matched test.
5) Penelitian ini menggunakan batas panjang input maksimum 512 token sesuai konfigurasi model yang digunakan.

D. Tujuan Penelitian

Tujuan dari penelitian ini adalah sebagai berikut:

1) Mengembangkan dan mengevaluasi model CodeT5-small berbasis Seq2Seq untuk mengotomatisasi klasifikasi biner deteksi cacat kode pada dataset PyTraceBugs.
2) Menganalisis efektivitas Focal Loss dan WeightedRandomSampler dalam mengatasi class imbalance serta meningkatkan kemampuan model dalam mengenali kelas buggy.
3) Mengevaluasi apakah performa model benar-benar merefleksikan kemampuan deteksi cacat, bukan sekadar memanfaatkan bias panjang kode.
4) Menyusun skema evaluasi yang lebih kuat melalui split label + length-bin, baseline length-only, dan length-matched test.

E. Manfaat penelitian

Manfaat yang diharapkan dari penelitian ini adalah sebagai berikut:

1) Secara praktis, penelitian ini dapat membantu proses quality assurance perangkat lunak dengan menyediakan mekanisme deteksi cacat kode lebih dini, sehingga waktu debugging dapat dikurangi, kualitas perangkat lunak meningkat, dan risiko lolosnya defect ke lingkungan produksi dapat ditekan.
2) Secara metodologis, penelitian ini menyediakan pipeline evaluasi yang lebih kuat untuk code defect detection melalui pengendalian bias panjang kode menggunakan length-binned split, baseline length-only, dan length-matched test, sehingga hasil eksperimen menjadi lebih valid dan tidak terjebak pada shortcut.
3) Secara akademik, penelitian ini memperkaya kajian deteksi cacat kode dengan memanfaatkan CodeT5-small serta strategi penanganan class imbalance yang dapat dijadikan rujukan bagi penelitian selanjutnya.

BAB V
PENUTUP

A. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa model CodeT5-small berbasis arsitektur Sequence-to-Sequence dapat diterapkan untuk melakukan klasifikasi biner deteksi cacat kode Python pada dataset PyTraceBugs. Dalam penelitian ini, potongan kode Python digunakan sebagai input, sedangkan label kelas direpresentasikan dalam bentuk teks, yaitu "clean" dan "defective". Pendekatan ini menunjukkan bahwa CodeT5-small dapat digunakan tidak hanya untuk tugas pemahaman dan generasi kode, tetapi juga untuk tugas klasifikasi kode dalam bentuk generatif.

Hasil penelitian menunjukkan bahwa model CodeT5-small dengan kombinasi Focal Loss dan WeightedRandomSampler mampu memberikan performa yang lebih baik dibandingkan baseline length-only. Pada test set, model usulan memperoleh accuracy sebesar 80,14%, precision kelas defective sebesar 67,89%, recall sebesar 86,51%, F1-score sebesar 76,08%, MCC sebesar 60,79%, dan AUC sebesar 83,73%. Dibandingkan baseline length-only yang hanya memperoleh accuracy sebesar 69%, precision sebesar 55%, recall sebesar 76%, dan F1-score sebesar 64%, model usulan menunjukkan peningkatan pada seluruh metrik utama. Hal ini menunjukkan bahwa model tidak hanya mengandalkan panjang kode sebagai dasar prediksi, tetapi mampu mempelajari representasi kode yang lebih informatif.

Penggunaan Focal Loss dan WeightedRandomSampler juga terbukti membantu meningkatkan sensitivitas model terhadap kelas defective.

WeightedRandomSampler digunakan untuk meningkatkan peluang kemunculan sampel dari kelas minoritas selama pelatihan, sedangkan Focal Loss digunakan untuk memberi perhatian lebih besar pada sampel yang sulit diklasifikasikan. Kombinasi kedua strategi tersebut membuat model lebih peka dalam mengenali kode defective. Hal ini terlihat dari nilai recall kelas defective yang mencapai 86,51%, sehingga sebagian besar kode cacat pada test set berhasil dikenali oleh model.

Evaluasi terhadap potensi bias panjang kode menunjukkan bahwa panjang kode memang memiliki pengaruh terhadap proses klasifikasi. Hal ini terlihat dari baseline length-only yang masih mampu memperoleh F1-score sebesar 64,00%. Namun, model utama tetap menunjukkan performa yang baik ketika diuji menggunakan length-matched test. Pada skenario tersebut, model memperoleh accuracy sebesar 75,51%, precision kelas defective sebesar 72,03%, recall sebesar 83,39%, F1-score sebesar 77,30%, dan MCC sebesar 51,66%. Meskipun accuracy dan MCC mengalami penurunan dibandingkan test set normal, recall dan F1-score kelas defective tetap berada pada tingkat yang baik. Hasil ini menunjukkan bahwa model tidak sepenuhnya bergantung pada panjang kode sebagai shortcut, tetapi juga mampu menangkap pola lain dari isi kode sumber.

Secara keseluruhan, penelitian ini menunjukkan bahwa kombinasi CodeT5-small, Focal Loss, dan WeightedRandomSampler dapat digunakan sebagai pendekatan yang cukup efektif untuk deteksi cacat kode Python. Model usulan mampu meningkatkan kemampuan deteksi terhadap kelas defective, sekaligus memberikan evaluasi yang lebih kuat melalui penggunaan baseline length-only dan length-matched test. Dengan demikian, penelitian ini tidak hanya berfokus pada peningkatan performa

model, tetapi juga memperhatikan validitas evaluasi agar hasil yang diperoleh tidak semata-mata dipengaruhi oleh bias panjang kode.

B. Saran

Berdasarkan hasil penelitian dan keterbatasan yang ditemukan, terdapat beberapa saran yang dapat dipertimbangkan untuk penelitian selanjutnya.

1) Pertama, penelitian selanjutnya disarankan untuk melakukan ablation study secara terpisah antara Focal Loss dan WeightedRandomSampler. Pada penelitian ini, kedua strategi tersebut digunakan secara bersamaan, sehingga hasil yang diperoleh merepresentasikan performa dari skema gabungan. Oleh karena itu, penelitian selanjutnya dapat membandingkan beberapa skenario, seperti model tanpa Focal Loss dan tanpa WeightedRandomSampler, model dengan Focal Loss saja, model dengan WeightedRandomSampler saja, serta model dengan kombinasi keduanya. Dengan demikian, kontribusi masing-masing metode terhadap peningkatan performa dapat dianalisis secara lebih jelas.
2) Kedua, penelitian selanjutnya dapat mengeksplorasi penggunaan model dengan kapasitas yang lebih besar atau model bahasa kode lain yang memiliki kemampuan representasi lebih kuat. CodeT5-small dipilih dalam penelitian ini karena lebih sesuai dengan keterbatasan sumber daya komputasi. Namun, model yang lebih besar atau model dengan konteks input lebih panjang berpotensi menangkap struktur kode yang lebih kompleks, terutama pada sampel yang melebihi batas 512 token.

3) Ketiga, penelitian berikutnya dapat mengembangkan strategi untuk menangani sampel kode yang mengalami truncation. Dalam penelitian ini, sebagian data masih melebihi batas panjang input maksimum 512 token, sehingga sebagian konteks kode berpotensi terpotong. Untuk mengatasi hal tersebut, penelitian selanjutnya dapat menggunakan pendekatan chunking, sliding window, atau model dengan kapasitas konteks yang lebih panjang agar informasi penting pada kode panjang tetap dapat diproses oleh model.
4) Keempat, penelitian selanjutnya dapat melakukan pengujian pada dataset defect detection lain atau pada bahasa pemrograman selain Python. Hal ini penting untuk melihat apakah pendekatan CodeT5-small dengan Focal Loss dan WeightedRandomSampler dapat digeneralisasikan pada konteks dataset dan bahasa pemrograman yang berbeda. Dengan pengujian lintas dataset, kekuatan generalisasi model dapat dianalisis secara lebih luas.
5) Kelima, penelitian selanjutnya juga dapat melakukan tuning threshold dan kalibrasi skor prediksi untuk menyesuaikan kebutuhan penggunaan model. Dalam konteks deteksi cacat kode, recall yang tinggi memang penting untuk mengurangi risiko bug yang terlewat, tetapi precision juga tetap perlu diperhatikan agar jumlah false positive tidak terlalu besar. Oleh karena itu, pengaturan threshold dapat menjadi langkah lanjutan untuk menyeimbangkan kebutuhan antara sensitivitas dan ketepatan prediksi.
6) Keenam, penelitian ini masih berfokus pada evaluasi model secara eksperimental. Untuk pengembangan lebih lanjut, model

dapat diintegrasikan ke dalam sistem bantu code review atau pipeline quality assurance perangkat lunak. Dengan demikian, model tidak hanya dievaluasi dari sisi metrik, tetapi juga dapat diuji manfaatnya dalam membantu pengembang memprioritaskan potongan kode yang berpotensi mengandung cacat.

DAFTAR PUSTAKA

Akimova, E.N. *et al.* (2021) 'A survey on software defect prediction using deep learning', *Mathematics*, 9(11), pp. 1–14. Available at: https://doi.org/10.3390/math9111180.

Albattah, W. and Alzahrani, M. (2024) 'Software Defect Prediction Based on Machine Learning and Deep Learning Techniques: An Empirical Approach', *AI (Switzerland)*, 5(4), pp. 1743–1758. Available at: https://doi.org/10.3390/ai5040086.

Barr, C.J.S. *et al.* (2026) 'A Review of Fairness and a Practical Guide to Selecting Context-Appropriate Fairness Metrics in Machine Learning', pp. 1–31.

Boabang, F. and Gyamerah, S.A. (2025) 'An Enhanced Focal Loss Function to Mitigate Class Imbalance in Auto Insurance Fraud Detection with Explainable AI'. Available at: http://arxiv.org/abs/2508.02283.

Bryan, J. and Moriano, P. (2023) 'Graph-based machine learning improves justin- time defect prediction', *PLoS ONE*, 18(4 April), pp. 1–19. Available at: https://doi.org/10.1371/journal.pone.0284077.

Chicco, D., Starovoitov, V. and Jurman, G. (2021) 'The Benefits of the Matthews Correlation Coefficient (MCC) over the Diagnostic Odds Ratio (DOR) in Binary Classification Assessment', *IEEE Access*, 9(Mcc), pp. 47112–47124. Available at: https://doi.org/10.1109/ACCESS.2021.3068614.

Destefanis, G. *et al.* (2026) 'An audit of machine learning experiments on software defect prediction'.

van Dinter, R. *et al.* (2023) 'Just-in-time defect prediction for mobile applications: using shallow or deep learning?', *Software Quality Journal*,

31(4), pp. 1281–1302. Available at: https://doi.org/10.1007/s11219-023-09629-1.

Fan, X. *et al.* (2025) 'Software Fault Localization Based on SALSA Algorithm', *Applied Sciences (Switzerland)*, 15(4). Available at: https://doi.org/10.3390/app15042079.

Harsh, Santoshi, H. and Singh, P. (2025) 'Comparative Study of Machine Learning Based Defect Prediction Models for Python Software', *Proceedings of the 6th International Conference on Inventive Research in Computing Applications, ICIRCA 2025*, pp. 1867–1872. Available at: https://doi.org/10.1109/ICIRCA65293.2025.11089647.

Laracy, J.R. *et al.* (2025) 'Software Quality Assurance and AI: A Systems-Theoretic Approach to Reliability, Safety, and Security', *Software*, 4(4), p. 30. Available at: https://doi.org/10.3390/software4040030.

Li, J. (2024) 'Area under the ROC Curve has the most consistent evaluation for binary classification', *PLoS ONE*, 19(12 December). Available at: https://doi.org/10.1371/journal.pone.0316019.

Malhotra, R. *et al.* (2021) 'Support vector based oversampling technique for handling class imbalance in software defect prediction', *Proceedings of the Confluence 2021: 11th International Conference on Cloud Computing, Data Science and Engineering*, pp. 1078–1083. Available at: https://doi.org/10.1109/Confluence51648.2021.9377068.

Mukherjee, R. *et al.* (2022) 'Static Analysis for AWS Best Practices in Python Code', *Leibniz International Proceedings in Informatics, LIPIcs*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany, pp. 14:1-14:0. Available at: https://doi.org/10.4230/LIPIcs.ECOOP.2022.14.

Nabot, A. and Al-Qerem, A. (2025) 'Impact of software quality on organizational performance', *Array*, 27(February), p. 100476. Available at: https://doi.org/10.1016/j.array.2025.100476.

Rashed, A., Kallich, A. and Eltayeb, M. (2025) 'Analyzing Fairness of Classification Machine Learning Model with Structured Dataset', *International Journal of Advanced Multidisciplinary Research and Studies*, 5(6), pp. 445–453. Available at: https://doi.org/10.62225/2583049x.2025.5.6.5222.

Shao, Y. *et al.* (2020) 'A novel test case prioritization method based on problems of numerical software code statement defect prediction', *Eksploatacja i Niezawodnosc*, 22(3), pp. 419–431. Available at: https://doi.org/10.17531/ein.2020.3.4.

Sujon, K.M. *et al.* (2025) 'Accuracy, precision, recall, f1-score, or MCC? empirical evidence from advanced statistics, ML, and XAI for evaluating business predictive models', *Journal of Big Data*, 12(1). Available at: https://doi.org/10.1186/s40537-025-01313-4.

Tang, S. (2026) 'Creating a Python platform for students to learn full-stack development of financial applications', *Financial Innovation*, 12(1). Available at: https://doi.org/10.1186/s40854-025-00813-9.

Ustyannie, W., Setyaningsih, E. and Iswahyudi, C. (2024) 'Optimization of software defects prediction in imbalanced class using a combination of resampling methods with support vector machine and logistic regression', *Jurnal Infotel*, 13(4), pp. 176–184. Available at: https://doi.org/10.20895/infotel.v13i4.726.

Wang, Y. *et al.* (2021) 'CodeT5: Identifier-aware Unified Pre-trained Encoder-Decoder Models for Code Understanding and Generation', *EMNLP 2021 - 2021 Conference on Empirical Methods in Natural*

Language Processing, Proceedings, pp. 8696–8708. Available at: https://doi.org/10.18653/v1/2021.emnlp-main.685.

Xiong, S. *et al.* (2025) 'Defect Detection in Source Code via Multimodal Feature Fusion', *Applied Sciences (Switzerland)*, 15(17). Available at: https://doi.org/10.3390/app15179358.

Zhu, H.-N. *et al.* (2025) 'From Bugs to Benchmarks: A Comprehensive Survey of Software Defect Datasets', *ACM Computing Surveys*, 1(1). Available at: http://arxiv.org/abs/2504.17977.