THESIS

STUDI PERBANDINGAN CODET5 DENGAN PLBART UNTUK *BUG FIXING* SCRIPT BAHASA PEMROGRAMAN JAVA

Oleh:

Rahmatun Nazila

NIM : 23206051027

PROGRAM STUDI INFORMATIKA

PROGRAM MAGISTER FAKULTAS SAINS DAN TEKNOLOGI

UIN SUNAN KALIJAGA

YOGYAKARTA

2025

HALAMAN PENGESAHAN

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-1939/Un.02/DST/PP.00.9/08/2025

Tugas Akhir dengan judul : Study perbandingan CodeT5 dengan PLBART untuk bug fixing script bahasa pemrograman java

yang dipersiapkan dan disusun oleh:

Nama : RAHMATUN NAZILA, S.Kom
Nomor Induk Mahasiswa : 23206051027
Telah diujikan pada : Sabtu, 16 Agustus 2025
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: 68a7d05107147

Penguji I

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

Valid ID: 68a7463f07223

Penguji II

Dr. Ir. Sumarsono, S.T., M.Kom.
SIGNED

Valid ID: 68a48e8f533c7

Yogyakarta, 16 Agustus 2025
UIN Sunan Kalijaga
Dekan Fakultas Sains dan Teknologi

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

Valid ID: 68a8a3bca8061

PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini:

Nama	: Rahmatun Nazila
NIM	:23026051027
Jenjang	:Magister
Program Studi	:Informatika

Menyatakan bahwa naskah tesis ini secara keseluruhan adalah hasil penelitian/karya saya sendiri, kecuali pada bagian-bagian yang dirujuk sumbernya.

Yogyakarta, 20 Agustus 2025

Saya yang menyatakan

Rahmatun Nazila

NIM:23206051027

PERNYATAAN BEBAS PLAGIASI

Yang bertanda tangan dibawah ini:

Nama	: Rahmatun Nazila
NIM	:23026051027
Jenjang	:Magister
Program Studi	:Informatika

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

Yogyakarta, 20 Agustus 2025

Saya yang menyatakan

Rahmatun Nazila

NIM:23206051027

NOTA DINAS PEMBIMBING

Nota dinas pembimbing

Kepada Yth,

Dekan Fakultas Sains dan Teknologi

UIN Sunan Kalijaga

Yogaykarta

Assalamualaikum wr.wb

Setelah melakukan bimbingan, arahan dan k oreksi terhadap penulisan tesis yang berjudul

"Studi Perbandingan codeT5 dengan PLBART untuk bug fixing script bahasa pemrograman java".

Yang ditulis oleh:

Nama	: Rahmatun Nazila
NIM	:23026051027
Jenjang	:Magister
Program Studi	:Informatika

Saya berpendapat bahwa tesis tersebut sudah dapat diajukan kepada Magister Informatika UIN sunan kalija untuk diujikan dalam rangka memperoleh gelar Magister Informatika

Wassalamualaikum wr.wb.

Yogyakarta, 12 Agustus 2025

Pembimbing

Dr. Agung Fatwanto S.Si, M.Kom

ABSTRAK

Penelitian ini bertujuan untuk mengevaluasi dan membandingkan kinerja dua model bahasa besar (LLM), CodeT5 dan PLBART, dalam tugas perbaikan bug otomatis pada kode pemrograman Java. Proses perbaikan bug otomatis menjadi penting karena dapat meningkatkan efisiensi dan mengurangi ketergantungan pada perbaikan manual. Penelitian ini menggunakan dataset HumanEval dan melibatkan proses fine-tuning untuk meningkatkan performa model. Hasil penelitian menunjukkan bahwa PLBART memiliki kinerja lebih baik daripada CodeT5, khususnya pada metrik fungsional pass@10, dengan peningkatan dari 0 menjadi 3 setelah fine-tuning, sedangkan CodeT5 meningkat dari 0 menjadi 1. Kedua model masih menghadapi kesulitan dalam menghasilkan patch yang valid, dengan mayoritas hasil tergolong non-sensical dan in-plausible. Penelitian ini menyimpulkan bahwa fine-tuning berperan penting dalam meningkatkan kinerja model LLM, namun keterbatasan jumlah data latih memengaruhi capaian hasil. Selain perbandingan, penelitian ini juga memberikan rekomendasi praktis: CodeT5 lebih sesuai digunakan untuk eksperimen awal dengan kode sederhana, sementara PLBART direkomendasikan untuk kasus bug yang lebih kompleks.

MOTTO

“Selalu Siapkan Ruang ikhlas dalam setiap doa dan harapan yang kau langitkan”

“Tidak ada sesuatu yang terjadi secara kebetulan, semua Atas izin Allah”

-syekh Ali Jaber

HALAMAN PERSEMBAHAN

Tesis ini penulis persembahkan untuk:

Almamater Tercinta

Prodi Magister Informatika

Fakultas Sains dan

Teknologi

Universitas Islam Negeri Sunan Kalijaga Yogyakarta

KATA PENGANTAR

Assalamu'alaikum wr.wb

Dengan mengucap puji dan syukur kepada Allah SWT yang telah memberikan rahmat, nikmat, dan hidayah-Nya, serta sholawat beserta salam saya curahkan kepada Nabi Muhammad SAW, saya dapat menyelesaikan tesis ini dengan baik. Tesis ini berjudul "Studi Perbandingan codeT5 dengan PLBART untuk bug fixing script bahasa pemrograman java". Penulisan tesis ini disusun sebagai pengajuan untuk memenuhi salah satu syarat kelulusan guna memperoleh gelar Magister Informatika Fakultas Sains dan Tekonologi di Universitas Islam Negeri Sunan Kalijaga Yogyakarta. Penulis menyadari bahwa dalam menyelesaikan ini tidak terlepas dari bantuan bimbingan dari berbagai pihak. Oleh karena itu, penulis mengucapkan terimakasih kepada:

1. Ibu Nyai Hj Khusnul Khotimah warson, Abah nanang, umi bety, Abah kholid, Umi aina beserta keluarga. Selaku pengasuh komplek Q Yang telah memberikan ridho, semangat dan kepercayaan kepada penulis,semoga Allah SWT selalu memberkahi dan memberikan kesehatan kepada keluarga ndalem.
2. Keluarga penulis Umi dandian (jimatku), adek Aulia, mbak dwi, ponakan tercinta (alif, abiyasa, dan abisma), dan aa ridho. Terima kasih atas segala jasa, iringan doa yang tidak pernah terputus sampai sekarang dan selalu menyemangati dalam

mengiringi setiap perjalanan menyelesaikan pendidikan sampai jenjang ini, semoga Allah SWT selalu memberkahi kehidupan kita.

3. Prof. Noorhaidi Hasan. Selaku Rektor di Universitas Islam Negeri Sunan Kalijaga Yogyakarta yang telah memberikan kesempatan untuk menempuh studi.
4. Prof. Dr. Dra. Hj. Khurul Wardati, M.Si. Selaku dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Sunan Kalijaga Yogyakarta yang telah memberikan kesempatan untuk menempuh Pendidikan di Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta.
5. Dr. Ir. Sumarsono, S.T., M.Kom. selaku Kepala Prodi Magister Informatika Universitas Islam Negeri Sunan Kalijaga Yogyakarta dan Ir. Muhammad Didik Rohmad Wahyudi, S.T., M.T. Selaku Sekretaris Jurusan Prodi Informatika Universitas Islam Negeri Sunan Kalijaga Yogyakarta.
6. Bapak Dr. Agung Fatwanto S.Si, M.Kom. Selaku pembimbing yang telah banyak berkenan meluangkan waktu, tenaga, serta ilmunya untuk memberikan bimbingan kepada penulis dalam menyelesaikan tesis ini. Terima kasih penulis ucapkan dan penghargaan setinggi-tingginya yang dengan penuh kesabaran dan kearifan telah memberikan arahan dan semangat di sela-sela kesibukannya.
7. Bapak dan Ibu Dosen, Staff Tata Usaha, dan tenaga lain Prodi Magister Informatika Universitas Islam Sunan Kalijaga Yogyakarta, terkhusus yang memberikan materi kuliah.

Terima kasih penulis ucapkan karena sudah memberikan banyak ilmu pengetahuan sehinga penulis dapat menyelesaikan perkuliahan dan penelitian yang menjadi tesis ini.

8. Teman-teman Ciblon terutama Annida rizki Lutfi Astuti dan Muh. Nur Aslam yang telah membantu penelitian dan penulisan tesis ini hingga selesai, Zishwa Muhammad Jauhar Nafis, Muhammad Fahson Hakim, dan Supardi. Terima kasih sudah menjadi bagian perjalanan hidup selama masa perkuliahan.
9. Keluarga bani alawi, terutama mamah (Dewi habibatul alawiyah) dan alifia yg selalu support dan exited atas segala pencapaian penulis, farida, mbak qorry, bulek (mbak fifi) dan syifa. Yg selalu support dan baik kepada penulis.
10. Ala Asmaul Husna, Siti Muslichah, dan teman-teman mahasiswa Prodi Magister Informatika dan Program Beasiswa Santri Berprestasi 2023 Universitas Islam Negeri Sunan Kalijaga Yogyakarta.
11. Teman-teman kamar, zeni, frida, sabila, sayyidah, afida, peni, dini, irma, ifa, dan ima. Yang menjadi tempat keluh kesah pulang kuliah dan nugas,selalu bilang "Amin" ketika penulis melangitkan harapan dan doa dikamar.
12. Kru londry MTPR yg Strong dan hebat, Peni, Hesti, Nafila, Zeni, Shofi, dan mbak diah terima kasih sudah ikhlas dan tetap solid dalam pengabdian ini
13. Teman-teman guru MATQ Bu dewi kamad, irdho, husna, mbak elmi, hanin,mbak jeki, usro, yumna, mila, mbak dina,

pak humam, dan pak wasul yang selalu support dan mendoakan penulis.

14. Rahmatun nazila, yang sudah berjuang sejauh ini, mengahadapi semua masalah dan tantangan yang ada, berbekal doa umi, Sholawat dan usaha insya Allah sebuah bisa. Kamu hebat dan luar biasa.
15. Ucapan terima kasih penulis sampaikan kepada semua pihak yang tidak mungkin penulis sebutkan satu demi satu, yang telah banyak memberikan dukungan selama perkuliahan sampai penyelesaian tesis ini.

Semoga Allah SWT melimpahkan Rahmat dan Karunia-Nya kepada kita semua. Penulis berharap semoga tesis ini bermanfaat bagi penulis dan pembaca lainnya.

Wassalamu'alaikum wr. wb.

Yogyakarta, 20 Agustus
2025 Penyusun

Rahmatun Nazila
NIM. 23206051021

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

A. Latar Belakang Masalah

Perkembangan teknologi kecerdasan buatan (*Artificial Intelligence*/AI) telah memberikan dampak besar yang signifikan terhadap berbagai aspek kehidupan manusia, mulai dari bidang kesehatan, keuangan, hingga rekayasa perangkat lunak. AI merupakan cabang di bidang ilmu komputer bertujuan mengembangkan sistem yang mampu meniru kemampuan berpikir dan pengambilan keputusan seperti manusia. Salah satu cabang dari AI yang mengalami kemajuan signifikan dalam beberapa tahun terakhir adalah *Natural Language Processing* (NLP).

Natural Language Processing (NLP) adalah bidang kecerdasan buatan yang berfokus pada bagaimana komputer memahami, memproses dan memproduksi komputer secara alami. Perkembangan NLP telah mendorong kemampuan mesin dalam menjembatani komunikasi antara manusia dan computer, tidak hanya dalam bentuk teks umum, tetapi juga dalam konteks-konteks teknis seperti kode pemrograman. Salah satu tahap awal dalam perkembangan NLP modern adalah diperkenalkannya arsitektur Transformer oleh (Vaswani et al., 2017). Arsitektur ini menggantikan pendekatan berbasis urutan seperti *Recurrent Neural Network* (RNN) dan *Long Short-Term Memory* (LSTM), dengan menerapkan mekanisme self-attention penuh yang memungkinkan pemrosesan data secara parallel dan pemahaman konteks global secara efisisen. Inovasi ini membuka jalan bagi lahirnya berbagai model Bahasa berskala besar atau Large Language Models (LLMs), seperti BERT, GPT, CodeT5 dan varian lainnya. Large Language Models (LLMs) adalah model bahasa berskala besar yang dilatih

dengan miliaran parameter untuk memahami dan menghasilkan teks secara kontekstual. LLM tidak hanya menunjukkan performa luar biasa dalam memahami teks alami, tetapi juga telah berhasil diaplikasikan dalam berbagai tugas yang berkaitan dengan code pemrograman, seperti code generation, code completion, dan code repair (R. Wang et al., 2024).

Salah satu permasalahan atau tantangan dalam pengembangan perangkat lunak adalah keberadaan bug atau kesalahan dalam kode program yang dapat menyebabkan kerusakan sistem, celah keamanan, hingga kerugian ekonomi(Ge et al., 2024). Bug dapat menimbulkan berbagai dampak negative, mulai dari kerusakan system, resiko keamanan, hingga kerugian finansial. Proses perbaikan bug secara manual membutuhkan waktu dan keahlian teknis yang tinggi, sehingga menjadi beban signifikan dalam siklus pengembangan perangkat lunak. Dalam konteks ini, LLM yang dilatih secara khusus untuk memahami kode pemrograman dinilai memiliki potensi besar untuk digunakan sebagai alat bantu dalam proses perbaikan bug secara otomatis. Dengan memahami struktur sintaksis dan semantik dari kode, LLM dapat menghasilkan patch perbaikan yang kontekstual dan relevan. Meskipun demikian, implementasi LLM dalam code repair masih menghadapi berbagai tantangan, seperti keterbatasan generalisasi, risiko data leakage, dan kebutuhan komputasi yang tinggi. Oleh karena itu, pemanfaatan LLM dalam perbaikan kode menjadi topik yang penting untuk dikaji lebih lanjut, baik dari sisi efektivitas, arsitektur model, dataset yang digunakan, hingga evaluasi hasil perbaikannya.

Untuk mengatasi permasalahan bug dalam perangkat lunak secara lebih sistematis dan efisien, dikembangkanlah pendekatan *Automated Program Repair* (APR), yaitu proses otomatis dalam mendeteksi dan memperbaiki kesalahan kode. *Automated Program Repair* (APR) sendiri memiliki

beberapa teknik pendekatan seperti, Search-Based Repair menggunakan model GenProg, RSRepair, dan ARJA(Bian et al., 2021). Kemudian Constraint-Based Repair menggunakan model Semfix dan Angelix(Le Goues et al., 2019). selanjutnya, Template-Based Repair menggunakan model PAR dan NPEFix (Zhang, Fang, Zhang, et al., 2023), dan Teknik yang terakhir Learning-Based Repair menggunakan model Getafix, TFix, CodeBERT, dan CODET5(Zhang, Fang, Ma, et al., 2023). Pendekatan terakhir ini menjadi focus utama dalam penelitian APR modern karena model Pra-latih Language Models (PLM) memiliki kemampuan untuk mempelaajri pola perbaikan dari data histori dan menggeneralisasikannya ke konteks baru.

CodeT5 merupakan model pra-latih berbasis arsitektur Transformer yang dikembangkan secara khusus untuk berbagai tugas pemrograman, termasuk code summarization, code translation, code completion, hingga code repair(Y. Wang et al., 2021). Model ini mengadopsi pendekatan Text to Text Transfer Transformer (T5), namun dimodiifkasi agar dapat memproses representasi sintaksis dan semantic dari Bahasa pemrograman dengan lebih optimal. Salah satu keunggulan CodeT5 adalah kemampuannya dalam menghasilkan representasi yang lebih kontekstual. Selain itu, CodeT5 menggunakan corpus besar yang terdiri dari berbagai Bahasa pemrograman, seperti Java, Python, dan JavaScript sehingga memiliki potensi generalisasi yang lebih baik untuk tugas Bahasa pemrograman lainnya.

Di sisi lain, PLBART merupakan model yang dirancang dengan pendekatan sequence to sequence berbasis arsitektur BART, tetapi dimodifikasi untuk mempelajari pasangan teks-kode, seperti dokumentasi dan kode program(Ahmad et al., 2021). Fokus PLBART lebih condong

pada penerjemahan antara deskripsi dan kode, namun performanya juga menjanjikan untuk tugas perbaikan bug.

Baik CodeT5 maupun PLBART menunjukkan performa yang kompetitif dalam tugas *bug fixing*. CodeT5, dengan kemampuannya menghasilkan representasi kontekstual yang lebih kaya, mampu memahami pola sintaksis sekaligus makna semantik dalam kode, sehingga efektif dalam mendeteksi serta memperbaiki kesalahan logika maupun sintaks (Bui et al., 2021; Y. Wang et al., 2021). Sementara itu, PLBART yang dilatih pada pasangan teks–kode memungkinkan model ini memahami konteks perintah atau dokumentasi yang berkaitan dengan kode, sehingga dapat menawarkan perbaikan yang lebih relevan terhadap potongan kode yang bermasalah(Ahmad et al., 2021; N. Jiang et al., 2023b). Kombinasi pendekatan ini menjadikan keduanya unggul sebagai model pembantu otomatis dalam memperbaiki bug pada berbagai bahasa pemrograman.

Dalam lima tahun terakhir, penelitian mengenai code repair dalam bahasa Java mengalami perkembangan signifikan. Pendekatan seperti RepairLLaMA memperkenalkan teknik fine-tuning efisien menggunakan representasi kode optimal dan adapter yang disesuaikan, sehingga mampu memperbaiki lebih banyak bug dibanding baseline sebelumnya(Silva et al., 2023). Studi oleh (N. Jiang et al., 2023a) juga menunjukkan bahwa model bahasa kode (Code Language Models/CLM) yang dilatih khusus dengan data APR dapat memperbaiki 46% hingga 164% lebih banyak bug dibanding pendekatan deep learning klasik. Beberapa model seperti CodeBERT, GraphCodeBERT, dan CodeT5 telah banyak digunakan dalam penelitian, tetapi studi-studi tersebut umumnya hanya menguji satu model atau membandingkan dengan baseline klasik seperti GenProg, jKali, dan ARJA (Y. Wang et al., 2021)(Guo et al., 2021)(Feng et al., 2020). Sebagian

besar pada penelitian sebelumnya, seperti yang ditunjukkan dalam studi oleh (Silva et al., 2023) dan (N. Jiang et al., 2023a) mengungkapkan bahwa masih terdapat keterbatasan dalam eksplorasi metode fine-tuning dan representasi kode yang optimal dalam konteks APR. Misalnya, meskipun CodeT5 dan PLBART telah terbukti efektif dalam berbagai tugas generatif kode, perbandingan langsung antara keduanya secara spesifik untuk perbaikan kode Java, dengan mempertimbangkan representasi kode dan konteks bug (seperti *fault localization* atau *multi-location bugs*), masih belum banyak dilakukan secara sistematis.

Penelitian ini berfokus pada evaluasi 2 model pra-latih berbasis Transformer yang digunakan dalam konteks code repair, yakni CodeT5 dan PLBART. Kedua model ini dirancang khusus untuk memahami dan menghasilkan kode pemrograman, serta telah digunakan dalam berbagai studi sebelumnya dengan hasil yang bervariasi. Meskipun sama-sama mengadopsi arsitektur Transformer, CodeT5, dan PLBART memiliki strategi pra pelatihan, struktur model, dan data pelatihan yang berbeda, sehingga perlu dikaji lebih lanjut mengenai kinerjanya dalam tugas perbaikan bug. Oleh karena itu, penelitian ini bertujuan untuk melakukan studi perbandingan antara CodeT5 dan PLBART dalam tugas perbaikan bug otomatis pada kode pemrograman Java. Metrik evaluasi yang digunakan berbasis semantik kode, seperti kategori patch yang dihasilkan (valid, implausible, dan nonsensical). Melalui penelitian ini, diharapkan diperoleh pemahaman yang lebih mendalam mengenai efektifitas dan efisiensi masing-masing model dalam menangani masalah bug fixing secara otomatis.

B. Rumusan Masalah

Berdasarkan latar belakang permasalahan, rumusan masalah yang diperoleh, yaitu:

1. Bagaimana performa model CodeT5 dan PLBART dalam melakukan *bug fixing* pada script Bahasa pemrograman Java?
2. Model manakah yang memiliki kinerja lebih baik dalam memperbaiki bug berdasarkan klasifikasi hasil patch?
3. Rekomendasi apa yang dapat diberikan terkait pemilihan model untuk tipe kode atau skenario pengembangan tertentu?

C. Tujuan Penelitian

1. Mengevaluasi performa model CodeT5 dan PLBART dalam memperbaiki bug pada script Java.
2. Membandingkan kinerja CodeT5 dan PLBART berdasarkan klasifikasi hasil patch.
3. Memberikan rekomendasi praktis terkait pemilihan model yang sesuai untuk berbagai tipe kode dan skenario pengembangan perangkat lunak.

D. Manfaat Penelitian

1. Memperkaya literatur terkait Gambaran mengenai evaluasi performa model CodeT5 dan PLBART pada Bahasa pemrograman Java.
2. Menyediakan referensi dan pertimbangan teknis bagi pengembang perangkat lunak, peneliti, dan praktisi dalam memilih model PLM yang lebih tepat untuk diterapkan dalam system *Automated Program Repair* (APR) pada proyek berbasis Java.

E. Batasan Penelitian

1. Penelitian ini hanya dilakukan pada Bahasa pemrograman Java.

2. Dataset yang digunakan dibatasi pada dataset *bug fixing*.
3. Model yang digunakan terbatas pada CodeT5 dan PLBART.
4. Evaluasi yang digunakan menggunakan hasil klasifikasi patch ke dalam tiga kategori: valid, in-plausible, dan nonsensical.

BAB V
PENUTUP

A. Kesimpulan

Penelitian ini bertujuan untuk mengevaluasi dan membandingkan performa model CodeT5 dan PLBART dalam tugas perbaikan bug otomatis pada kode pemrograman Java. Berdasarkan hasil evaluasi yang dilakukan dengan menggunakan dataset HumanEval, kedua model mengalami peningkatan performa setelah dilakukan fine-tuning. Model PLBART menunjukkan kinerja yang lebih baik dibandingkan dengan CodeT5, terutama dalam hal metrik fungsional pass@10, yang meningkat dari 0 menjadi 3 setelah fine-tuning. Sementara itu, CodeT5 mengalami peningkatan dari 0 menjadi 1, meskipun hasil akhirnya tetap lebih rendah dibandingkan PLBART. Meskipun begitu, kedua model memiliki tantangan yang signifikan dalam menghasilkan patch yang valid dan dapat berfungsi sesuai harapan, dengan banyak hasil yang masuk ke dalam kategori non-sensical dan in-plausible.

Secara keseluruhan, penelitian ini menunjukkan bahwa fine-tuning memainkan peranan penting dalam meningkatkan performa model dalam tugas perbaikan bug otomatis. Namun, data pelatihan yang digunakan dalam penelitian ini, yang hanya mencakup 10.000 data bug, membatasi potensi peningkatan yang dapat dicapai oleh model. Oleh karena itu, peningkatan jumlah data pelatihan dapat membawa dampak yang lebih signifikan terhadap kinerja model.

Selain perbandingan, penelitian ini juga memberikan rekomendasi praktis. CodeT5 dapat digunakan untuk eksperimen awal atau kasus dengan kode sederhana karena lebih ringan dan mudah dijalankan pada sumber

daya terbatas. Sebaliknya, PLBART lebih direkomendasikan untuk bug yang lebih kompleks atau proyek dengan kebutuhan perbaikan kode lintas fungsi. Dengan demikian, hasil penelitian ini tidak hanya menunjukkan perbedaan performa antar model, tetapi juga menawarkan panduan pemilihan model sesuai kebutuhan pengembangan perangkat lunak.

B. Saran

Berdasarkan temuan ini, beberapa saran untuk penelitian selanjutnya adalah sebagai berikut:

1. Peningkatan Dataset Pelatihan: Penelitian ini terbatas pada penggunaan 10.000 data bug untuk pelatihan. Penambahan jumlah dataset yang lebih besar, seperti yang digunakan dalam penelitian sebelumnya (misalnya 143.000 data), dapat memberikan hasil yang lebih baik dan signifikan, serta meningkatkan kemampuan generalisasi model.
2. Eksplorasi Model Lain: Penelitian ini hanya membandingkan dua model, yaitu CodeT5 dan PLBART. Penelitian selanjutnya dapat menguji model lain seperti CodeBERT atau GraphCodeBERT untuk melihat apakah model lain dapat memberikan kinerja yang lebih baik dalam konteks perbaikan bug otomatis.
3. Pendekatan Hybrid: Menggabungkan metode fine-tuning dengan teknik lainnya seperti retrieval-augmented generation (RAG) atau penggunaan reinforcement learning untuk mengoptimalkan hasil perbaikan bug dapat menjadi arah penelitian yang menjanjikan.
4. Penggunaan Benchmark Lebih Beragam: Penelitian selanjutnya bisa mempertimbangkan penggunaan berbagai benchmark untuk menguji kinerja model, termasuk dataset yang lebih beragam dan

lebih besar untuk menangani berbagai jenis bug dan bahasa pemrograman.

Dengan melakukan pengembangan lebih lanjut pada aspek-aspek ini, diharapkan model-model yang ada dapat lebih efektif dalam menyelesaikan masalah perbaikan bug di dunia nyata, serta memberikan kontribusi yang lebih besar terhadap kemajuan di bidang rekayasa perangkat lunak otomatis.

DAFTAR PUSTAKA

Ahmad, W. U., Chakraborty, S., Ray, B., & Chang, K. W. (2021). *Unified Pre-training for Program Understanding and Generation. NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, 2655–2668. https://doi.org/10.18653/v1/2021.naacl-main.211

Anand, A., Gupta, A., Yadav, N., & Bajaj, S. (2024). A *Comprehensive Survey of AI-Driven Advancements and Techniques in Automated Program Repair and Code Generation*. 1(1), 1–19. http://arxiv.org/abs/2411.07586

Bian, Z., Blot, A., & Petke, J. (2021). *Refining Fitness Functions for Search-Based Program Repair. Proceedings - 2021 IEEE/ACM International Workshop on Automated Program Repair, APR* 2021, 1–8. https://doi.org/10.1109/APR52552.2021.00008

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., Brynjolfsson, E., Buch, S., Card, D., Castellon, R., Chatterji, N., Chen, A., Creel, K., Davis, J. Q., Demszky, D., … Liang, P. (2021). O*n the Opportunities and Risks of Foundation Models*. 1–214. http://arxiv.org/abs/2108.07258

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., … Amodei, D. (2020). *Language models are few-shot learners. Advances in Neural Information Processing Systems, 2020-Decem.*

Bui, N. D. Q., Wang, Y., & Hoi, S. C. H. (2021). *Learning to Debug with CodeT5.*

Cai, X., & Jiang, L. (2025). *Adapting Knowledge Prompt Tuning for Enhanced Automated Program Repair. Proceedings - 2025 IEEE International Conference on Software Analysis, Evolution and Reengineering*, SANER 2025, 360–371. https://doi.org/10.1109/SANER64311.2025.00041

Clevinger, A., & Dexheimer, V. (n.d.). *Dense-matter equation of state at zero & finite temperature.* 8–11.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: *Pre-training of deep bidirectional transformers for language understanding. NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference,* 1(Mlm), 4171–4186.

Eunseok, J. heo; hohyeon jeong; and, & Lee; (2022). *Patch It If You Can : Increasing the Efficiency of Patch*. 1–15.

Feng, Z., Guo, D., Tang, D., Duan, N., Feng, X., Gong, M., Shou, L., Qin, B., Liu, T., Jiang, D., & Zhou, M. (2020). *CodeBERT: A pre-trained model for programming and natural languages. Findings of the Association for Computational Linguistics Findings of ACL*: EMNLP 2020, January, 1536–1547. https://doi.org/10.18653/v1/2020.findings-emnlp.139

Ge, S., Zhang, Y., Liu, L., Zhang, M., Han, J., & Gao, J. (2024). *Model Tells You What To Discard: Adaptive Kv Cache Compression for Llms. 12th International Conference on Learning Representations*, ICLR 2024, 1–14.

Gharibi, R., Sadreddini, M. H., & Fakhrahmad, S. M. (2024). T5APR: *Empowering Automated Program Repair across languages through checkpoint ensemble. Journal of Systems and Software*, 214. https://doi.org/10.1016/j.jss.2024.112083

Gomez-ponce, J., Choi, T., Abbasi, N. A., Adame, A., & Alvarado, A. (2021). *Air-to-Ground Directional Channel Sounder With 64-antenna Dual-polarized Cylindrical Array*.

Guo, D., Ren, S., Lu, S., Feng, Z., Tang, D., Liu, S., Zhou, L., Duan, N., Svyatkovskiy, A., Fu, S., Tufano, M., Deng, S. K., Clement, C., Drain, D., Sundaresan, N., Yin, J., Jiang, D., & Zhou, M. (2021). *Graphcodebert: Pre-Training Code Representations With Data Flow. ICLR 2021 - 9th International Conference on Learning Representations*, 1–18.

Jiang, J., Xiong, Y., Zhang, H., Gao, Q., & Chen, X. (2018). *Shaping program repair space with existing patches and similar code. ISSTA 2018 - Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis, June*, 298–309. https://doi.org/10.1145/3213846.3213871

Jiang, N., Liu, K., Lutellier, T., & Tan, L. (2023a). Impact of Code Language Models on Automated Program Repair.

Jiang, N., Liu, K., Lutellier, T., & Tan, L. (2023b). *Impact of Code Language Models on Automated Program Repair. Proceedings - International Conference on Software Engineering*, 1430–1442. https://doi.org/10.1109/ICSE48619.2023.00125

Kouamo, S., Tangha, C., & Kouamo, O. (2020). *Reduction of False Rejection in an Authentication System by Fingerprint with Deep Neural Networks. Journal of Software Engineering and Applications*,

13(01), 1–13. https://doi.org/10.4236/jsea.2020.131001

Le-cong, T., Luong, D., Le, X. B. D., & Lo, D. (2023). *Invalidator : Automated Patch Correctness Assessment via Semantic and Syntactic Reasoning.* 1–20.

Le Goues, C., Pradel, M., & Roychoudhury, A. (2019). *Automated Program Repair. Communications of the ACM*, 62(12), 56–65. https://doi.org/10.1145/3318162

Le, H., Wang, Y., Gotmare, A. D., Savarese, S., & Hoi, S. C. H. (2022). *CodeRL: Mastering Code Generation through Pretrained Models and Deep Reinforcement Learning. Advances in Neural Information Processing Systems*, 35.

Li, H. (2023). *Improve Code Summarization via Prompt-Tuning CodeT5. Wuhan University Journal of Natural Sciences,* 28(6), 474–482. https://doi.org/10.1051/wujns/2023286474

Liu, K., Li, L., Koyuncu, A., Kim, D., Liu, Z., Klein, J., & Bissyandé, T. F. (2021). *A critical review on the evaluation of Automated Program Repair systems. Journal of Systems and Software,* 171. https://doi.org/10.1016/j.jss.2020.110817

Mei, H., & Zhang, L. (2018). *Can big data bring a breakthrough for software automation? Science China Information Sciences*, 61(5), 56101. https://doi.org/10.1007/s11432-017-9355-3

Mohiuddin, K., Welke, P., Alam, M. A., Martin, M., Alam, M. M., Lehmann, J., & Vahdati, S. (2023). *Retention Is All You Need. International Conference on Information and Knowledge Management*, Proceedings, Nips, 4752–4758. https://doi.org/10.1145/3583780.3615497

Nindel-Edwards, J., & Steinke, G. (2015). *A Full Life Cycle Defect*

*Process Model That Supports Defect Tracking, Software Product Cycles, And Test Iterations. Communications of the IIM*A, 6(1). https://doi.org/10.58729/1941-6687.1290

Salazar-jurado, E. H., Vilches-ponce, K., & Mora, M. (2022). *T OWARDS THE G ENERATION OF S YNTHETIC I MAGES OF P ALM V EIN P ATTERNS*: A R EVIEW.

Silva, A., Fang, S., & Monperrus, M. (2023). *RepairLLaMA: Efficient Representations and Fine-Tuned Adapters for Program Repair.* 1–15. http://arxiv.org/abs/2312.15698

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). *Attention is all you need. Advances in Neural Information Processing Systems, 2017-Decem(Nips)*, 5999–6009.

Wang, R., Chen, H., Zhou, R., Ma, H., Duan, Y., Kang, Y., Yang, S., Fan, B., & Tan, T. (2024). *LLM-Detector: Improving AI-Generated Chinese Text Detection with Open-Source LLM Instruction Tuning*. http://arxiv.org/abs/2402.01158

Wang, W., Wang, Y., Joty, S., & Hoi, S. C. H. (2023). *RAP-Gen: Retrieval-Augmented Patch Generation with CodeT5 for Automatic Program Repair. ESEC/FSE 2023 - Proceedings of the 31st ACM Joint Meeting European Software Engineering Conference and Symposium on the Foundations of Software Engineering,* Dl, 146–158. https://doi.org/10.1145/3611643.3616256

Wang, Y., Wang, W., Joty, S., & Hoi, S. C. H. (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*,

8696–8708. https://doi.org/10.18653/v1/2021.emnlp-main.685

Wu, Y., Jiang, N., Pham, H. V., Lutellier, T., Davis, J., Tan, L., Babkin, P., & Shah, S. (2023). *How Effective Are Neural Networks for Fixing Security Vulnerabilities. ISSTA 2023 - Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*, 1282–1294. https://doi.org/10.1145/3597926.3598135

Zhang, Q., Fang, C., Ma, Y., Sun, W., & Chen, Z. (2023). *A Survey of Learning-based Automated Program Repair. ACM Transactions on Software Engineering and Methodology*, 33(2). https://doi.org/10.1145/3631974

Zhang, Q., Fang, C., Zhang, T., Yu, B., Sun, W., & Chen, Z. (2023). *Gamma: Revisiting Template-Based Automated Program Repair Via Mask Prediction. Proceedings - 2023 38th IEEE/ACM International Conference on Automated Software Engineering*, ASE 2023, 535–547. https://doi.org/10.1109/ASE56229.2023.00063

Zhao, Z., & Fard, F. (2025). *Do Current Language Models Support Code Intelligence for R Programming Language? ACM Transactions on Software Engineering and Methodology,* 1(1). https://doi.org/10.1145/3735635

Zou, W., Li, Q., Ge, J., Li, C., Shen, X., Huang, L., & Luo, B. (2023). *A Comprehensive Evaluation of Parameter-Efficient Fine-tuning on Software Engineering Tasks.* 1(1), 1–32. http://arxiv.org/abs/2312.15614