Penggunaan Metrik *Software Defect* dalam Pengembangan Perangkat Lunak: Literature Review

Ayu Kusuma Dewi[1], Hilma Yasmin Azmina[2] dan [3]Yuni Sugiarti
[1,2,]Sistem Informasi, Universitas Islam Negeri Syarif Hidayatullah Jakarta, Indonesia
Email: [1]ayu.kusumadewi22@mhs.uinjkt.ac.id, [2]hilma.yasmin22@mhs.uinjkt.ac.id, [3]yuni.sugiarti@uinjkt.ac.id

Abstrak - Penggunaan metrik pada software defect adalah sebagai alat pengukur untuk membantu mengidentifikasi risiko dalam memprediksi cacat dalam perangkat lunak. Metrik perangkat lunak membantu berbagai pelaku industri agar memastikan produk akhir berkualitas tinggi. Metrik ini mampu melakukan mitigasi risiko sejak awal hingga akhir proses pengembangan perangkat lunak dan membantu dalam pencegahan dan perbaikan cacat perangkat lunak. Tujuan penulisan ini adalah untuk mengetahui pemilihan metrik, teknik dan model yang sesuai sebelum melakukan pengujian pada perangkat lunak. Metodologi yang digunakan dalam penulisan ini adalah studi literatur. Hasil penulisan tersebut adalah untuk mengetahui bahwa terdapat banyak jenis metrik yang dapat digunakan dalam suatu penelitian. Pada permasalahan yang ada mengenai pemilihan jenis metrik untuk cacat perangkat lunak, langkah yang perlu dilakukan adalah dengan mengidentifikasi permasalahan tersebut. Lalu dilakukannya diskusi mengenai jalan keluar yang tepat dengan pemangku kepentingan untuk menjadi jalan keluar dari sebuah permasalahan.

Kata Kunci – Cacat Perangkat Lunak, Metrik, SLR

Abstract - The use of metrics for software defects is as a measuring tool to help identify risks in predicting defects in software. Software metrics help various industry players ensure high-quality end products. This metric is able to mitigate risks from the beginning to the end of the software development process and assists in preventing and repairing software defects. The purpose of this research is to determine the selection of appropriate metrics, techniques and models before testing the software. The methodology used in this research is literature study. The results of this research are to find out that there are many types of metrics that can be used in research. In the existing problem regarding selecting the type of metric for software defects, the step that needs to be taken is to identify the problem. Then a discussion is held regarding the appropriate solution with stakeholders to provide a solution to a problem.

Keywords – Software Defect, Metrics, SLR

I. PENDAHULUAN

Perangkat lunak merupakan program yang terdiri dari serangkaian instruksi yang dijalankan oleh komputer untuk melaksanakan tugas tertentu. Bentuk perangkat lunak dapat berupa data elektronik yang disimpan di komputer serta berupa program atau instruksi yang akan dieksekusi oleh komputer [1]. Agar dapat terbentuk, perangkat lunak memerlukan baris-baris kode yang dibentuk berdasarkan bahasa pemrogramannya dan ditulis oleh seorang pemrogram (programmer) agar dapat dikenali oleh komputer. [Bimbingan pengenalan perangkat lunak komputer kepada siswa – siswi smp djojoredjo][2]

Dalam proses pengembangan perangkat lunak, tidak terlepas kemungkinan bahwa akan terjadi suatu kesalahan yang membuat kode tidak berjalan dengan semestinya. Baik karena kesalahan logika mau pun kesalahan semantik. Mitigasi risiko dan prediksi kesalahan sejak awal hingga akhir proses pengembangan ini sangat penting dilakukan untuk mengoptimasi pengerjaan proyek dan menghindari pengeluaran biaya berlebih karena kesalahan dapat dihindari [3]. Untuk memenuhi kebutuhan ini, terdapat suatu alat ukur kuantitatif dan sistematis yang dikenal sebagai metrik untuk memprediksi kesalahan perangkat lunak (software defect). Metrik ini membantu mengidentifikasi risiko dan memprediksi kesalahan, sehingga pencegahan dan perbaikan dapat dilakukan lebih awal [4]. *Software defect* (*bug*) sendiri merupakan kesalahan yang dapat menyebabkan perangkat lunak mengalami kegagalan dan sangat sulit untuk dihindari dalam proses desain desain dan pengembangan perangkat lunak [5].

Berbagai penelitian telah dilakukan untuk mengevaluasi efektivitas berbagai metrik dalam memprediksi kesalahan perangkat lunak. Misalnya, penelitian [6] membandingkan metrik ukuran, metrik kompleksitas siklomatis CC, dan metrik Chidamber dan Kemerer dari metrik berorientasi objek. Namun, penelitian ini terbatas hanya pada ketiga metrik tersebut dan tidak mencakup teknik yang lebih umum digunakan saat ini. Penelitian lain [7] menganalisis metrik LoC, Kompleksitas Siklomatik, metrik Halstead, metrik Chidamber-Kemerer, dan metrik proses, namun hasil dari kinerja metrik yang ditemukan tidak selalu konsisten dan memerlukan penelitian lebih lanjut.

Penelitian lebih lanjut [8] dilakukan untuk memprediksi kesalahan pada perangkat lunak melalui analisis dataset, model, kerangka kerja, dan metrik kinerja. Namun, penelitian ini memiliki rentang waktu yang terbatas karena hanya menganalisis metrik dan teknik dari tahun 2017 hingga 2021. Selain itu, penelitian oleh Sourabh Pal dan Alberto Sillitti [9] berfokus pada model CPDP menggunakan metrik produk, tetapi terbatas pada jenis proyek dan dataset yang bersifat open-source.

Meninjau penelitian-penelitian sebelumnya yang masih memungkinkan untuk ditingkatkan, penelitian ini dilakukan dengan tujuan utama untuk mengidentifikasi metrik-metrik yang digunakan dalam memprediksi kesalahan dalam perangkat lunak yang tidak hanya terbatas pada jenis metrik tertentu atau model tertentu serta menemukan kekurangan

dari setiap penelitian yang diulas untuk membantu menilai efektifitas dan membantu pemilihan metrik di masa yang akan datang. Penelitian ini akan menggali informasi berdasarkan penelitian-penelitian terdahulu menggunakan metode *systematic literature review*.

II. METODE PENELITIAN

Pada tinjauan sistematis literatur, pendekatan metodologi yang digunakan adalah studi literatur. Metodologi ini bertujuan untuk mengumpulkan informasi dan sumber data yang relevan dengan topik yang dibahas dalam penelitian [10]. Penelitian ini juga didapatkan melalui sumber-sumber terkait seperti jurnal akademik, buku, laporan penelitian, artikel dan website.

Pada metodologi tinjauan literatur, beberapa tahap prosedural diikuti oleh peneliti untuk memperoleh informasi penelitian terkait dan melakukan analisis [11]. Tahapan pertama yang dilakukan oleh penelitian ini adalah menyiapkan alat atau sumber daya untuk membantu dalam pencarian informasi penelitian. Alat bantu dapat terdiri dari berbagai jenis, pada penelitian ini peneliti menggunakan Mendeley, Google Scholar, Google Translate dan Typeset.

Tahapan kedua adalah proses pencarian paper menggunakan Google Scholar. Google Scholar merupakan suatu platform pengindeksan yang cukup populer untuk makalah akademik, buku, dan berbagai karya berkualitas tinggi lainnya yang telah diterbitkan. Hal ini memungkinkan integrasi yang baik dan kemampuan pencarian yang efisien. Sehingga, pengguna dapat menilai dampak kutipan dari masing-masing artikel yang sesuai. Google Scholar menampilkan kumpulan publikasi yang dihasilkan oleh peneliti, dengan mempertimbangkan faktor-faktor seperti kelengkapan isi artikel, penulis, jurnal tempat artikel diterbitkan, dan seberapa sering artikel tersebut dikutip dalam karya ilmiah lainnya. Hasil teratas yang dianggap paling relevan akan secara konsisten ditampilkan di halaman awal [12]. Pada penelitian ini, peneliti melakukan pencarian untuk jurnal menggunakan kata kunci *"Software Defect".*

Tahapan ketiga adalah membuka file terkait menggunakan mendeley. Mendeley adalah alat perangkat lunak yang berguna untuk mengelola basis data karya ilmiah. Mahasiswa memiliki kemampuan untuk menangani kutipan secara efektif ketika menggunakan Mendeley. Dalam kapasitas penyelenggara kutipan, mahasiswa memiliki kapasitas untuk mendokumentasikan rincian sumber referensi individu dalam platform Mendeley terpadu, kemudian membuatnya mudah untuk mengutip atau menyinggung sumber pada berbagai kesempatan sepanjang karya tertulis mereka [13] Membuka file mendeley yang berbentuk .ris lalu secara otomatis referensi pada penelitian akan tersedia sesuai dengan file yang sebelumnya telah di analisis.

Tahapan keempat adalah setelah jurnal penelitian telah rampung, peneliti membuat abstrak bahasa indonesia dan bahasa inggris. Pada abstrak bahasa inggris penulis menggunakan Google Translate sebagai tools pembantu penelitian. Google translate adalah alat bantu penerjemah gratis yang dibuat oleh Google, memfasilitasi terjemahan cepat kalimat, dokumen, dan situs web dari satu bahasa ke bahasa lainnya.

Tahapan kelima adalah finishing, setelah proses jurnal telah rampung dari judul,abstrak hingga referensi, selanjutnya peneliti melakukan pemilihan jurnal yang sesuai dengan focus and scope.Peneliti juga menambahkan data yang di cari melalui web dan Typeset.

III. HASIL DAN PEMBAHASAN

Tabel 1. Literature review

Sumber	*Metode Penelitian*	*Hasil Pembahasan*	*Limitasi Penelitian*
[14]	Metode ini menggunakan data processing, oversampling dan self training semi supervised learning.	Penelitian ini memperkenalkan model prediksi keparahan cacat perangkat lunak menggunakan self-training semi-supervised learning. Juga mengusulkan lima ukuran proyek untuk mengevaluasi dampaknya: faktor risiko, persentase anggaran terselamatkan, waktu layanan tersisa, dan waktu layanan gratis.	Pengembangan model prediksi keparahan cacat perangkat lunak dengan pendekatan self-training semi-terawasi memiliki keterbatasan: tidak dapat digeneralisasi ke model lain, penggunaan pengklasifikasi pohon keputusan mungkin tidak optimal, dan evaluasi model terbatas pada ukuran khusus proyek.
[15]	Metode penelitian ini menggunakan teknik seleksi fitur dan pembelajaran mesin untuk membangun model prediksi cacat perangkat lunak dengan	Tidak ada metode seleksi atau konstruksi fitur yang selalu unggul untuk prediksi cacat perangkat lunak. Kinerja optimal tergantung pada algoritma pembelajaran mesin yang digunakan,	Terbatasnya informasi mengenai proyek tersebut dan tidak ada informasi tentang uji statistik khusus yang digunakan untuk menganalisis.

	metrik dari repositori NASA dan Eclipse. Data mencakup tiga kategori utama: metrik dimensi, metrik kompleksitas, dan metrik orientasi objek.	dengan evaluasi menggunakan akurasi, presisi, recall, dan area di bawah kurva.	
[16]	Penelitian ini menggunakan Logistic Regression untuk menganalisis metrik perangkat lunak dari berbagai proyek dan memodelkan hubungan antara metrik-metrik tersebut dan kemungkinan kecacatan. Metode ini dirancang untuk mengidentifikasi pola yang dapat memprediksi kecacatan perangkat lunak	Penelitian menunjukkan bahwa pendekatan CPDP-OSS lebih efisien dibandingkan metode pesaing, menghasilkan prediksi cacat perangkat lunak yang lebih akurat dan meningkatkan efisiensi serta efektivitas dalam aplikasi dunia nyata. Pendekatan ini memberikan solusi yang lebih baik dan andal untuk memprediksi serta mengelola cacat perangkat lunak.	Penelitian ini terbatas pada tiga hingga empat jenis metrik perangkat lunak tertentu, yang membatasi variasi dan kompleksitas metrik yang dianalisis oleh model. Aspek lain dari metrik perangkat lunak mungkin tidak dipertimbangkan, sehingga mempengaruhi keakuratan dan generalitas model.
	lintas proyek.		
[17]	Penelitian ini menggunakan teknologi transformator untuk memodelkan pola urutan panggilan metode dalam kode sumber perangkat lunak dari repositori publik. Metodologi mencakup pengolahan data urutan panggilan, pengembangan model prediksi dengan pembelajaran mesin, dan evaluasi kinerja menggunakan akurasi, presisi, recall, dan F1-score.	Hasil penelitian menunjukkan bahwa prediksi berdasarkan urutan pemanggilan metode lebih baik dibandingkan prediksi berdasarkan efek tingkat kelas, dengan nilai Mean Absolute Error (MAE) 8% lebih rendah. Hasil ini menegaskan keunggulan pendekatan berbasis urutan dalam menganalisis dan memprediksi cacat perangkat lunak.	Dalam penelitian ini, fokus terbatas pada analisis dan prediksi cacat perangkat lunak pada tingkat entitas kasar seperti kelas, file, dan paket. Perlu penelitian lebih lanjut untuk mengembangkan metode yang lebih komprehensif dalam memprediksi berbagai jenis cacat perangkat lunak.
[18]	Penelitian ini menggunakan dua metode untuk memprediksi kerusakan perangkat lunak:	Model yang diusulkan berhasil melampaui skema yang sudah ada dalam akurasi, Area Under the Curve (AUC), dan	Ketidakseimbangan kelas dalam penelitian ini membuat data cenderung miring. Regresi Logistik dan ansambel pohon

	Logistic Regression yang memperti mbangkan biaya dan model prediksi berbasis kumpulan pohon keputusan. Penelitian ini mengguna kan data dari berbagai fitur perangkat lunak untuk mengeval uasi seberapa baik kedua model ini dalam mengident ifikasi potensi kerusakan.	perolehan. Evaluasi dilakukan menggunakan 11 kumpulan data yang berbeda, menunjukkan performa yang signifikan dan konsisten dalam memprediksi berbagai kasus potensial dalam konteks penelitian.	keputusan sulit menangani kelas minoritas, berdampak pada kinerja dan akurasi prediksi yang lebih condong pada mayoritas kelas. Tantangan evaluasi prediksi kerusakan perangkat lunak tetap signifikan.
[19]	Penelitian ini mengguna kan teknik clustering untuk mengelom pokkan dan membandi ngkan metrik dari berbagai perusahaa n guna mempredi ksi cacat perangkat lunak lintas perusahaa n. Metode pengatura n berat	Penelitian menunjukkan bahwa metode baru lebih unggul dalam memprediksi cacat perangkat lunak dibandingkan dengan CCDP arus utama, dievaluasi dengan data dari 12 proyek NASA dan PROMISE. Metode ini memberikan prediksi yang lebih akurat dan dapat diandalkan dalam	Dari hasil limitasi yang diungkapkan dalam penelitian ini, tantangan yang perlu diperhatikan adalah kehilangan informasi dari data akibat proses pencocokan metrik yang mungkin tidak mencerminkan kompleksitas dan detail data asli secara akurat. Penggunaan langsung metrik asli juga bisa
	badan juga diimpleme ntasikan untuk menyesuai kan bobot metrik, meningkat kan validitas dan akurasi prediksi.	mengidentifik asi masalah perangkat lunak.	menyebabkan inkonsistensi dalam dimensi yang digunakan dalam analisis.
[20]	Penelitian ini menganali sis korelasi antara metrik perangkat lunak dan tingkat keparahan cacat, mengguna kan pendekata n BPSO untuk mengident ifikasi subset fitur yang relevan dalam prediksi. Data berasal dari repositori perangkat lunak proyek-proyek berbeda untuk mengemb angkan pemahama n tentang hubungan ini.	Dari penelitian ini, empat pendekatan dalam pemilihan fitur untuk analisis dievaluasi untuk membandingk an cara mereka mengestimasi tumpang tindih fitur. Hasilnya memberikan pemahaman lebih dalam tentang efektivitas pendekatan ini dan menimbulkan pertanyaan baru untuk penelitian lanjutan dalam analisis fitur untuk prediksi perangkat lunak.	Tantangan terkait dengan pendekatan statistik dan berbasis mesin dalam menetapkan ambang batas untuk derivasi metrik perangkat lunak memerlukan validasi yang lebih cermat, terutama untuk bahasa pemrograman yang berbeda. Setiap bahasa pemrograman memiliki karakteristik unik yang dapat mempengaruhi perhitungan dan interpretasi metrik perangkat lunak.
[21]	Penelitian ini mengguna	Mayoritas penelitian prediksi cacat	Keterbatasan utama studi ini adalah

	kan pertanyaan penelitian dan kriteria penilaian untuk memilih makalah-makalah yang relevan. Pendekatan sistematis ini menyediakan pemahaman mendalam dalam literatur prediksi cacat perangkat lunak menggunakan pembelajaran mesin untuk aplikasi seluler.	perangkat lunak aplikasi seluler fokus pada Android (48%), dengan algoritma seperti Naive Bayes, SVM, Regresi Logistik, Jaringan Saraf Tiruan, dan Decision Trees paling umum digunakan. Ini menunjukkan dominasi penelitian pada platform Android dan preferensi terhadap algoritma-algoritma utama dalam pengembangan model prediktif.	terbatasnya repositori dan kumpulan data untuk prediksi kerusakan aplikasi seluler. Ada potensi untuk penelitian lebih lanjut dengan menggunakan teknik pembelajaran tanpa pengawasan dan semi-pengawasan untuk mengatasi masalah ini.
[22]	Penelitian ini menggunakan kombinasi metrik untuk memprediksi kesalahan pada perangkat lunak. Tiga jenis metrik utama yang digunakan adalah metrik produk, metrik perubahan, dan metrik	Metrik yang dikombinasikan meningkatkan hasil prediksi kesalahan pada perangkat lunak. Hasil *recall* tertinggi diraih oleh model Random Forest sedangkan model Logistic Regression memberikan performa paling baik ketika menggunakan kombinasi	Waktu dan tenaga untuk menguji komponen sistem terbatas sehingga pengujian secara menyeluruh tidak dapat benar-benar bisa dilakukan. Selain itu, biaya tinggi juga menghambat penelitian untuk melakukan eksplorasi lebih jauh.
	profil. Value yang hilang diatasi terlebih dahulu sebelum model digunakan.	dari ketiga metrik dalam hal nilai area di bawah kurva (AUC).	
[23]	Penelitian ini menggunakan metrik proses dan kode untuk memprediksi kesalahan perangkat lunak di tingkat kelas (class-level) dan menggunakan teknik pemilihan fitur untuk menilai efektivitas dari algoritma machine learning.	Metrik proses mempunyai tingkat akurasi yang lebih tinggi dibandingkan dengan metrik kode karena mampu memberikan pandangan yang lebih komprehensif mengenai proses pengembangan dan aspek pengembangan lainnya. Dengan menggabungkan metrik OO dalam proses pemilihan fitur, model dapat menangkap paradigma pemrograman dengan lebih baik.	Model prediksi kesalahan yang dilakukan mengabaikan metrik berorientasi objek lainnya. Selain itu, peneliti di awal proses pengembangan hanya fokus pada memprediksi masalah perangkat lunak.
[24]	Penelitian ini melakukan ekstraksi token Abstrak Sintaks Pohon (AST) sebagai pengujian vektor metrik dari *source code* untuk mewakili	Hasil penelitian menunjukkan bahwa model CCFT-CNN meningkatkan rata-rata ukuran F untuk prediksi kerusakan sebesar 2% dibandingkan dengan model dasar, yang menunjukkan bahwa penggunaan pengelompok	Penelitian ini fokus pada prediksi kesalahan pada perangkat lunak menggunakan pengujian metrik dan pengelompokan korelasi, tetapi tidak dilakukan eksplorasi mengenai integrasi jenis metrik lainnya

	kode semantik. Penelitian ini bertujuan untuk mengusulkan model baru Correlation Clustering berdasarkan pengujian metrik untuk memprediksi kerusakan perangkat lunak.	an korelasi dalam pengujian metric untuk prediksi kerusakan perangkat lunak termasuk efektif untuk dilakukan.	yang dapat membantu meningkatkan akurasi prediksi.
[25]	Penelitian ini menggunakan model CCFT-CNN untuk memprediksi kesalahan pada perangkat lunak, yang menggabungkan pengelompokan korelasi metrik pengujian Pohon Sintaks Abstrak (AST) dengan Jaringan Neural Konvolusional (CNN).	Evaluasi yang dilakukan menggunakan dataset dari empat proyek perangkat lunak menunjukkan bahwa metrik cost-effectiveness dapat memprioritaskan aturan yang mampu menemukan kesalahan lebih banyak dari metrik konvensional lainnya yang ditunjukkan seperti confidence dan odds ratio.	Penelitian ini terbatas pada penggunaan metrik cost-effectiveness untuk aturan asosiasi dalam memprediksi kesalahan pada perangkat lunak yang mungkin tidak berlaku jika kita menggunakan teknik prediksi atau domain lainnya. Evaluasi juga tidak mewakili keberagaman lingkungan pengembangan perangkat lunak lainnya.
[26]	Penelitian ini menggunakan metode eksperimen untuk mengetahui dampak jarak antar instansi terhadap kinerja tiga teknik yang berbasis SMOTE untuk memprediksi kesalahan pada perangkat lunak menggunakan metrik jarak (*distance metric*).	Penelitian ini menunjukkan bahwa metrik jarak memberikan dampak yang sangat kecil terhadap kinerja teknik SMOTE untuk memprediksi kesalahan pada perangkat lunak. Meskipun demikian, nilai Probability of Detection (PD) dan Probability of False alarm (PF) dalam SMOTE sangat dipengaruhi oleh jarak instansi yang dipilih.	Penelitian ini tidak mengeksplorasi faktor-faktor lain yang mungkin dapat kinerja teknik SMOTE. Selain itu, penelitian ini terbatas pada menyelidiki hubungan antara jarak instansi yang dipilih dengan kinerja teknik SMOTE sehingga aspek-aspek penting lainnya mungkin terlewat.
[27]	Penelitian ini dilakukan menggunakan metode seleksi fitur pearson untuk mengatasi masalah redundansi data yang muncul dalam proses pelatihan model. Teknik transfer learning dengan basis kompensasi metrik digunakan untuk mengatasi variasi ketika distribusi nilai metrik dilakukan	Hasil penelitian menunjukkan bahwa model prediksi kesalahan perangkat lunak yang dibuat menggunakan metode seleksi fitur pearson memiliki peningkatan kinerja. Model ini menunjukkan hasil yang lebih baik dalam hal area di bawah kurva (AUC) yang menunjukkan peningkatan tingkat akurasi prediksi.	Keterbatasan pada penelitian ini adalah tidak didiskusikannya secara mendalam dan spesifik mengenai kriteria yang dibuat ketika ingin memilih sumber proyek dan target untuk melakukan prediksi kesalahan antar proyek yang dapat mempengaruhi generalisabilitas dari metode seleksi fitur pearson.

	sehingga mampu meningkatkan tingkat akurasi model.		
[28]	Penelitian ini menggunakan pendekatan pembelajaran metrik dengan jarak yang berbasis cost-sensitive learning (CSL) dengan tujuan utama untuk mengatasi ketidakseimbangan kelas yang sering muncul dalam model prediksi kesalahan perangkat lunak dan diintegrasikan dengan large margin distribution machine (LDM) untuk meningkatkan hasil akurasi prediksi kesalahan.	Penelitian ini memberikan hasil yang mengonfirmasi bahwa model CS-ILDM memiliki kinerja untuk memprediksi kesalahan yang lebih unggul dibandingkan dengan model-model prediksi kesalahan lain yang pernah ada. Model ini juga efektif dalam mengurangi biaya yang diperlukan dalam mengatasi kesalahan dalam memprediksi dan mengurangi dampak kelas yang tidak seimbang dalam dataset.	Penelitian ini tidak menjelaskan kriteria yang digunakan untuk memilih proyek sumber dan target apa yang ingin dicapai dalam memprediksi kesalahan yang ada pada perangkat lunak dan tidak melakukan eksplorasi lebih dalam mengenai integrasi dengan teknik atau metode lainnya.
[29]	Penelitian ini menggunakan beberapa metrik baru yang disebut sebagai metrik perubahan teragregasi. Metrik ini dibuat dengan menggabungkan data-data seluruh perubahan yang dilakukan pada perangkat lunak melalui urutan yang kronologis. Kerangka kerja juga digunakan untuk mengekstrak semua perubahan yang terjadi pada versi modul proyek yang dipilih.	Penelitian menunjukkan bahwa eksperimen yang dilakukan terhadap proyek java menunjukkan bahwa set fitur yang diperluas dapat meningkatkan stabilitas dan kinerja model klasifikasi yang biasanya digunakan dalam memprediksi kesalahan pada perangkat lunak.	Penelitian ini hanya berfokus pada proyek Java *open-source* yang membatasi generalisabilitas hasil lainnya jika digunakan ke proyek lain. Selain itu, kualitas hasil eksperimen sangat bergantung pada dataset yang digunakan.

Tabel hasil *literature review* di atas menunjukkan beberapa metrik yang ditemukan paling efektif dalam memprediksi kesalahan yang ada dalam proses pengembangan perangkat lunak. Metrik-metrik yang ditemukan adalah sebagai berikut.

a. Risk-Factor (RF): Risk-Factor mampu menilai tingkat keparahan risiko yang mungkin terjadi dan menentukan prioritas dalam menyelesaikan masalah sehingga sumber daya dapat dialokasikan dengan lebih efektif. Namun, fokus utama dari metrik ini adalah menilai implikasi biaya, tidak mempertimbangkan faktor-faktor lain yang mempengaruhi proyek.
b. Semantic Metrics: Metrik ini menunjukkan peningkatan kinerja dalam memprediksi kesalahan antar proyek (CDPD) hingga lebih dari 50% sehingga dapat dijadikan sebagai salah satu pilihan.
c. Correlated Quality Metrics: Metrik ini membantu memilih fitur diskriminatif untuk meningkatkan akurasi dan kekuatan model prediksi kesalahan. Namun, metrik ini bergantung pada metrik kualitas yang diekstrak dari repository sumber kode dan data yang tidak seimbang dapat mempengaruhi efektivitas dari metrik.

d. Clustering-Based Metric Matching: Metrik ini mampu mengatasi inkonsistensi metrik dan perbedaan distribusi data antara proyek sumber dan target sehingga kinerja dari prediksi defect meningkat.
e. Object-Oriented Metrics: OO Metrics sangat efektif dalam mengidentidikasi modul yang rentan terhadap kesalahan dalam aplikasi mobile. Namun, kefokusan pada karakteristik struktural kode yang dilakukan oleh metrik rentan untuk mengabaikan faktor yang lainnya.
f. Combination of Change Metrics & Profile Metrics: Metrik ini merupakan metrik kombinasi yang dapat digunakan untuk mencapai nilai di bawah kurva terbaik dengan model Logistic Regression dan mencapai nilai recall tertinggi dengan model Random Forest. Namun, metrik ini masih kurang akurat dan memungkinkan terjadinya bias dalam proses analisis.

IV. KESIMPULAN DAN SARAN

Dari tinjauan 16 artikel jurnal, ditemukan bahwa metrik seperti Risk-Factor (RF), Semantic Metrics, Correlated Quality Metrics, Clustering-Based Metric Matching, Object-Oriented Metrics, dan kombinasi Change Metrics & Profile Metrics efektif dalam memprediksi kesalahan perangkat lunak. Efektivitas metrik-metrik ini bergantung pada teknik dan model yang diterapkan, seperti logistic regression dan Random Forest. Penggunaan kombinasi metrik yang tepat dapat meningkatkan akurasi prediksi, meskipun beberapa metrik memiliki keterbatasan seperti ketergantungan pada data sumber kode dan potensi bias analisis. Pemilihan metrik harus disesuaikan dengan kebutuhan spesifik proyek untuk meningkatkan efisiensi dan akurasi prediksi kesalahan.

DAFTAR PUSTAKA

[1] F. ADIPATI, "PERANGKAT LUNAK," 2020. [ONLINE]. AVAILABLE: HTTPS://API.SEMANTICSCHOLAR.ORG/CORPUSID:241169448

[2] M. F. ALBARI ET AL., "BIMBINGAN PENGENALAN PERANGKAT LUNAK KOMPUTER KEPADA SISWA-SISWI SMP DJOJOREDJO," ABDI JURNAL PUBLIKASI, VOL. 1, NO. 6, PP. 510–514, 2023.

[3] J. RAHMADI, "PENGUKURAN SOFTWARE METRIC TERHADAP APLIKASI SISTEM AKUNTANSI INSTANSI BERBASIS AKRUAL UNTUK PENGEMBANGAN SISTEM INFORMASI KEMENTERIAN PEMUDA DAN OLAHRAGA: ARRAY," JURNAL ILMIAH KOMPUTASI, VOL. 18, NO. 2, PP. 107–118, 2019.

[4] R. PARLIKA, D. C. M. WIJAYA, H. KHARIONO, AND R. A. FERNANDA, "STUDI LITERATUR PERBANDINGAN ANTARA METODE LOC, COCOMO, FPA DALAM RANAH SOFTWARE METRIC," JURNAL PENDIDIKAN INFORMATIKA DAN SAINS, VOL. 9, NO. 1, PP. 66–74, 2020.

[5] F. WANG, "SOFTWARE DEFECT FAULT INTELLIGENT LOCATION AND IDENTIFICATION METHOD BASED ON DATA MINING," IN JOURNAL OF PHYSICS: CONFERENCE SERIES, IOP PUBLISHING, 2022, P. 012012.

[6] I. A. RAHMAN, M. A. PERSADA, AND Y. SUGIARTI, "METRIC IN PREDICTING ERROR IN SOFTWARE: SYSTEMATIC LITERATURE REVIEW," JURNAL PERANGKAT LUNAK, VOL. 5, NO. 3, PP. 251–258, 2023.

[7] T. O. OLALEYE, O. T. AROGUNDADE, S. MISRA, A. ABAYOMI-ALLI, AND U. KOSE, "PREDICTIVE ANALYTICS AND SOFTWARE DEFECT SEVERITY: A SYSTEMATIC REVIEW AND FUTURE DIRECTIONS," SCI PROGRAM, VOL. 2023, NO. 1, P. 6221388, 2023.

[8] Y. Z. BALA, P. A. SAMAT, K. Y. SHARIF, AND N. MANSHOR, "CURRENT SOFTWARE DEFECT PREDICTION: A SYSTEMATIC REVIEW," IN 2022 APPLIED INFORMATICS INTERNATIONAL CONFERENCE (AIIC), IEEE, 2022, PP. 117–121.

[9] S. PAL AND A. SILLITTI, "CROSS-PROJECT DEFECT PREDICTION: A LITERATURE REVIEW," IEEE ACCESS, VOL. 10, PP. 118697–118717, 2022.

[10] B. A. HABSY, "SENI MEMEHAMI PENELITIAN KULIATATIF DALAM BIMBINGAN DAN KONSELING: STUDI LITERATUR," JURNAL KONSELING ANDI MATAPPA, VOL. 1, NO. 2, PP. 90–100, 2017.

[11] M. RIZKY AND Y. SUGIARTI, "PENGUNAAN METODE SCRUM DALAM PENGEMBANGAN PERANGKAT LUNAK: LITERATURE REVIEW," JOURNAL OF COMPUTER SCIENCE AND ENGINEERING (JCSE), VOL. 3, NO. 1, PP. 41–48, 2022.

[12] F. FADHILATURRAHMI, E. ERLINAWATI, AND R. ANANDA, "WORKSHOP SINTA 2 DAN GOOGLE SCHOOLAR DI UNIVERSITAS PAHLAWAN TUANKU TAMBUSAI," JURNAL ABDIDAS, VOL. 1, NO. 4, PP. 203–209, 2020.

[13] H. HANAFIAH, R. S. SAURI, D. MULYADI, AND O. ARIFUDIN, "PELATIHAN SOFTWARE MENDELEY DALAM PENINGKATAN KUALITAS ARTIKEL ILMIAH BAGI MAHASISWA," JURNAL KARYA ABDI MASYARAKAT, VOL. 5, NO. 2, PP. 213–220, 2021.

[14] R. SADAM, "TOWARDS DEVELOPING AND ANALYSING METRIC-BASED SOFTWARE DEFECT SEVERITY PREDICTION MODEL," ARXIV PREPRINT ARXIV:2210.04665, 2022.

[15] M. CANAPARO, E. RONCHIERI, AND G. BERTACCINI, "SOFTWARE DEFECT PREDICTION: A STUDY ON SOFTWARE METRICS USING STATISTICAL AND MACHINE LEARNING METHODS," IN PROCEEDINGS OF SCIENCE, 2022, PP. 21–25.

[16] Y. ZHONG, K. SONG, S. LV, AND P. HE, "AN EMPIRICAL STUDY OF SOFTWARE METRICS DIVERSITY FOR CROSS-PROJECT DEFECT PREDICTION," MATH PROBL ENG, VOL. 2021, NO. 1, P. 3135702, 2021.

[17] F. YANG, Y. HUANG, H. XU, P. XIAO, AND W. ZHENG, "FINE-GRAINED SOFTWARE DEFECT PREDICTION BASED ON THE METHOD-CALL SEQUENCE," COMPUT INTELL NEUROSCI, VOL. 2022, NO. 1, P. 4311548, 2022.

[18] A. ALI, N. KHAN, M. ABU-TAIR, J. NOPPEN, S. MCCLEAN, AND I. MCCHESNEY, "DISCRIMINATING FEATURES-BASED COST-SENSITIVE APPROACH FOR SOFTWARE DEFECT PREDICTION," AUTOMATED SOFTWARE ENGINEERING, VOL. 28, PP. 1–18, 2021.

[19] Y. SHAO, J. ZHAO, X. WANG, W. WU, AND J. FANG, "RESEARCH ON CROSS-COMPANY DEFECT PREDICTION METHOD TO IMPROVE SOFTWARE SECURITY," SECURITY AND COMMUNICATION NETWORKS, VOL. 2021, NO. 1, P. 5558561, 2021.

[20] G. MAUŠA, T. GALINAC GRBAC, L. BREZOČNIK, P. VILI, AND M. HERIČKO, "SOFTWARE METRICS AS IDENTIFIERS OF DEFECT OCCURRENCE SEVERITY," IN 8TH WORKSHOP SOFTWARE QUALITY ANALYSIS, MONITORING, IMPROVEMENT, AND APPLICATIONS (SQAMIA 2019), 2019, PP. 1–9.

[21] M. JORAYEVA, A. AKBULUT, C. CATAL, AND A. MISHRA, "MACHINE LEARNING-BASED SOFTWARE DEFECT PREDICTION FOR MOBILE APPLICATIONS: A SYSTEMATIC LITERATURE REVIEW," SENSORS, VOL. 22, NO. 7, P. 2551, 2022.

[22] A. KOMALASARI AND M. Z. C. CANDRA, "IMPROVING DEFECT PREDICTION USING COMBINATION OF SOFTWARE METRICS," IN 2022 INTERNATIONAL CONFERENCE ON DATA AND SOFTWARE ENGINEERING (ICODSE), IEEE, 2022, PP. 89–94.

[23] A. ZAIM, J. AHMAD, N. H. ZAKARIA, G. E. SU, AND H. AMNUR, "SOFTWARE DEFECT PREDICTION FRAMEWORK USING HYBRID SOFTWARE METRIC," JOIV: INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION, VOL. 6, NO. 4, PP. 921–930, 2022.

[24] K. K. SHARMA, A. SINHA, AND A. SHARMA, "SOFTWARE DEFECT PREDICTION USING DEEP LEARNING BY CORRELATION CLUSTERING OF TESTING METRICS," INTERNATIONAL JOURNAL OF ELECTRICAL AND COMPUTER ENGINEERING SYSTEMS, VOL. 13, NO. 10, PP. 953–960, 2022.

[25] K. NISHIURA, T. KASAGI, AND A. MONDEN, "A COST-EFFECTIVENESS METRIC FOR ASSOCIATION RULE MINING IN SOFTWARE DEFECT PREDICTION," IN 2023 CONGRESS IN COMPUTER SCIENCE, COMPUTER ENGINEERING, & APPLIED COMPUTING (CSCE), IEEE, 2023, PP. 2615–2620.

[26] S. FENG, J. KEUNG, P. ZHANG, Y. XIAO, AND M. ZHANG, "THE IMPACT OF THE DISTANCE METRIC AND MEASURE ON SMOTE-BASED TECHNIQUES IN SOFTWARE DEFECT PREDICTION," INF SOFTW TECHNOL, VOL. 142, P. 106742, 2022.

[27] J. CHEN, X. WANG, S. CAI, J. XU, J. CHEN, AND H. CHEN, "A SOFTWARE DEFECT PREDICTION METHOD WITH METRIC COMPENSATION BASED ON FEATURE SELECTION AND TRANSFER LEARNING," FRONTIERS OF INFORMATION TECHNOLOGY & ELECTRONIC ENGINEERING, VOL. 23, NO. 5, PP. 715–731, 2022.

[28] C. JIN, "SOFTWARE DEFECT PREDICTION MODEL BASED ON DISTANCE METRIC LEARNING," SOFT COMPUT, VOL. 25, NO. 1, PP. 447–461, 2021.

[29] L. ŠIKIĆ, P. AFRIĆ, A. S. KURDIJA, AND M. ŠILIĆ, "IMPROVING SOFTWARE DEFECT PREDICTION BY AGGREGATED CHANGE METRICS," IEEE ACCESS, VOL. 9, PP. 19391–19411, 2021.