Prosiding Seminar Nasional Ilmu Teknik
Volume 2 Nomor 2 Desember 2025
e-ISSN: 3063-5713; p-ISSN: 3063-4709, Hal 1160-1173
DOI: https://doi.org/10.61132/prosemnasproit.v2i2.156
Tersedia: https://prosiding.aritekin.or.id/index.php/PROSEMNASPROIT/

Optimasi Software Effort Estimation Menggunakan Random Forest

Maria Rosario Borroek[1*], Jasmir [2], Fachruddin[3], Marrylinteri [4], Yosefina Venus[5]
[1] Program Studi Sistem Informasi, Universitas Dinamika Bangsa, Indonesia
[2,3,5] Magister Sistem Informasi, Universitas Dinamika Bangsa, Indonesia
[4] Teknik Informatika, Universitas Dinamika Bangsa, Indonesia
Email: diamar_ros@yahoo.com[1], Jasmir@unama.ac.id.ac.id[2], fachruddin.stikom@gmail.com[3], Marrylinteri@unama.ac.id[4], ydifera@gmail.com[5]*

Alamat: Jl. Jend. Sudirman, The Hok, Kec. Jambi Sel., Kota Jambi, Jambi 36138
**Penulis Korespondensi: diamar_ros@yahoo.com*

Abstract*. Software development effort estimation is crucial as it is one of the key factors for successful software development. This research employs Random Forest to estimate software development effort. To achieve better results, the study combines the Random Forest method with Genetic Algorithm. The results show that the China dataset provides more accurate estimation compared to the Desharnais dataset, because the China dataset uses relevant feature selection for estimation..*

Keywords*: effort; software effort estimation, random forest, China dataset, Desharnais dataset.*

Abstrak. Estimasi usaha pengembangan perangkat lunak adalah sesuatu yang penting dilakukan karena merupakan salah satu kunci keberhasilan pengembangan perangkat lunak. Penelitian menggunakan Random Forest dalam mengestimasi usaha pengembangan perangkat lunak. Agar hasilnya lebih baik, penelitian mengkombinasikan metoda Random Forest dengan Genetic Algorthim. Hasilnya Dataset Cina memberikan hasil estimasi yang lebih akurat dibandingkan dengan dataset desharnais, dikarenakan pada dataset Cina menggunakan pemilihan fitur yang relevan untuk mengestimasinya.

Kata kunci: effort; estimasi perangkat lunak; Random Forest; Dataset Cina, dataset Derhanais.

1. LATAR BELAKANG

Estimasi usaha pengembangan perangkat lunak menjadi sangat penting untuk mencegah atau mengurangi kegagalan proyek, estimasi pada tahap awal siklus hidup perangkat lunak menjadi sangat diperlukan (Nashaat & Miller, 2025; Rankovic et al., 2021; Vanathi et al., 2024). Semakin awal estimasi dilakukan, semakin baik pengelolaan proyek yang dapat dilakukan (Park & Kim, 2020). Pentingnya estimasi awal terungkap ketika dibutuhkan untuk mengajukan penawaran pada proyek atau membuat komitmen kontrak antara pelanggan dan pengembang (Mahmood et al., 2022; Nassif et al., 2013). Hal tersebut penting dilakukan agar efisiensi dalam manajemen pengembangan perangkat lunak (Jadhav et al., 2022; Sharma & Chaudhary, 2023).

Effort adalah segala sumber daya dikeluarkan dalam mengembangkan perangkat lunak. *Effort* biasanya dievaluasi berdasarkan jumlah jam kerja seseorang atau jumlah uang yang dibutuhkan untuk mengimbangi pekerjaan tersebut (Sakhrawi et al., 2022; Sharma & Kushwaha, 2012; Tan et al., 2024). Dalam memprediksi ada banyak cara baik mengenai program perangkat lunak yang membantu menghitung perkiraan atau teknik spesifik yang memandu cara membuat prediksi ini. Pada metoda pengembangan ada beberapa metoda yang

Naskah Masuk: 18 Oktober 2025; Revisi: 29 Oktober 2025; Diterima: 29 November 2025; Terbit: 30 Desember 2025;

digunakan yakni: Model Agile yang fokusnya pada bagaimana menghasilkan perangkat lunak yang cepat, COCOMO yang focus kepada estimasi biaya proyek perangkat lunak berdasarkan ukuran dan kompleksitasnya, SEER-SEM yang fokusnya memperkirakan biaya perangkat lunak dengan mempertimbangkan berbagai faktor, termasuk ukuran proyek dan pengalaman tim, PUTNAM yang menggunakan data historis untuk memprediksi berapa lama proyek akan berlangsung berdasarkan ukuran dan kompleksitas perangkat lunak dan PRICE-S dan SLIM yang membantu dalam memperkirakan biaya perangkat lunak dengan menganalisis atribut dan karakteristik proyek yang berbeda (Denard et al., 2020; Mahmood et al., 2022; Nassif et al., 2013).

Banyak peneliti telah mencoba membuat model yang dapat memperkirakan upaya ini secara akurat(Suresh Kumar et al., 2020). Namun, pengembangan perangkat lunak selalu berubah. Bahasa dan metode pemrograman baru sedang dikembangkan, yang berarti bahwa teknik estimasi lama mungkin tidak berfungsi dengan baik lagi.

Metoda ML metoda yang digunakan untuk untuk memprediksi usaha pengembangan perangkat lunak diantaranya adalah Case Based Reasoning (CBR), Ensemble model, FAHP (Fuzzy analytic hierarchy process), Regression, RBFNN (feed forward neural network), fuzzy logic , ANN, Bayesian dan Decission Tree (DT), Least Square Support Vector Machine (LSSVM), Random Forest (RF), Support Vector Machine (SVM), Linear Regression (LR)(Nassif et al., 2019; Park & Kim, 2020; Sakhrawi et al., 2022)

Agar hasil Estimasi menjadi lebih akurat beberapa penelitian melakukan optimasi parameter (Khan et al., 2021). Optimasi dilakukan untuk membantu memilih fitur yang lebih berpengaruh dan meningkatkan akurasi estimasi. Beberapa metoda yang digunakan antara lain: *Genetic Algorithm* (GA), *Particle Swam Optimization* (PSO), *Genetic programming* (GP), *Artificial Bee Colony* (ABC)

2. KAJIAN TEORITIS

Estimasi Usaha Pengembangan perangkat Lunak

Estimasi usaha pengembangan perangkat lunak adalah proses yang digunakan untuk memperkirakan jumlah usaha yang diperlukan dalam pengembangan perangkat lunak (Phannachitta, 2020). Perkiraan ini mencakup berbagai aspek, seperti waktu pengerjaan, tenaga kerja yang dibutuhkan, serta biaya yang harus dikeluarkan. Estimasi pengembangan perangkat lunak sangat penting dalam manajemen proyek karena berperan dalam perencanaan sumber daya, penjadwalan, serta pengendalian anggaran agar proyek berjalan sesuai rencana (H. Liu et al., 2025; Shi et al., 2025).

Dalam melakukan estimasi, terdapat beberapa metode yang sering digunakan, seperti pendekatan berbasis pengalaman (expert judgment), metode algoritmik seperti COCOMO, serta teknik berbasis data historis dan machine learning (Y. Liu et al., 2025; Rosa & Jardine, 2025). Masing-masing metode memiliki kelebihan dan keterbatasan, sehingga pemilihan metode yang tepat bergantung pada karakteristik proyek dan data yang tersedia (Rashid et al., 2025; Singal et al., 2020).

Keakuratan estimasi pengembanagn perangkat lunak dipengaruhi oleh berbagai faktor, seperti tingkat kompleksitas proyek, pengalaman tim pengembang, serta ketidakpastian dalam spesifikasi perangkat lunak. Jika estimasi dilakukan secara tidak akurat, proyek dapat mengalami keterlambatan atau pembengkakan biaya yang berdampak pada efisiensi dan keberlanjutan pengembangan (Fernandez-Diego et al., 2020).

Untuk mengurangi risiko kesalahan dalam estimasi, disarankan untuk mengombinasikan berbagai teknik serta melakukan validasi secara berkala (Abdu et al., 2025). Dengan pendekatan ini, keandalan prediksi usaha dalam pengembangan perangkat lunak dapat meningkat, sehingga proyek dapat berjalan lebih efektif dan sesuai dengan target yang telah ditetapkan.

Machine Learning Network

Machine Learning (ML) kini menjadi teknologi yang banyak dimanfaatkan dalam pengembangan perangkat lunak, terutama untuk keperluan prediksi. Dengan kemampuannya dalam menganalisis data historis dan mengidentifikasi pola tersembunyi, ML membantu pengembang dalam menghasilkan estimasi yang lebih akurat terkait berbagai aspek perangkat lunak, seperti perhitungan usaha pengembangan, prediksi keandalan, deteksi bug, serta perkiraan kinerja system (Malhotra & Singh, 2025). Dalam rekayasa perangkat lunak, pendekatan berbasis ML berperan penting dalam meningkatkan efisiensi pengembangan, mengurangi potensi kesalahan, dan mempercepat pengambilan keputusan berbasis data.

Dalam penerapannya, ML dapat digunakan untuk berbagai keperluan prediksi dalam pengembangan perangkat lunak, seperti memperkirakan jumlah bug berdasarkan riwayat pengembangan, memprediksi kemungkinan keterlambatan proyek, atau mengestimasi beban sistem guna optimasi kinerja (Mahmood et al., 2020; Nashaat & Miller, 2025). Beberapa model pembelajaran mesin yang sering digunakan dalam bidang ini meliputi regresi linier, pohon keputusan, jaringan saraf tiruan (ANN), serta model ensemble seperti Random Forest dan Gradient Boosting. Dengan memilih algoritma yang sesuai dan menerapkan preprocessing data yang optimal, ML dapat menjadi alat yang efektif untuk meningkatkan kualitas serta akurasi estimasi dalam proses pengembangan perangkat lunak (Dai et al., 2025).

Random Forest

Random Forest (RF) adalah algoritma *ensemble learning* yang menggabungkan hasil dari beberapa *Decision Trees* untuk meningkatkan akurasi prediksi dan mengurangi *overfitting*. Konsep dasar dari RF adalah membangun banyak pohon keputusan (*decision trees*) secara independen dengan menggunakan subset acak dari data pelatihan dan fitur(Parmar et al., 2019),(Probst et al., 2019). Setiap pohon keputusan dalam hutan memberikan prediksi, dan hasil akhir diperoleh dengan menggabungkan prediksi dari seluruh pohon menggunakan metode *voting* untuk klasifikasi atau rata-rata untuk regresi. Pendekatan ini mengurangi variabilitas model dan meningkatkan kestabilan serta akurasi hasil prediksi. Pada penelitian peramalan dalam pengembangan software, RF unggul pada Dataset China dibandingkan 6 dataset lainnya, akan tetapi model stacking dengan RF menggunguli dibandingkan base model yang RF (A G et al., 2021)

Optimasi Optimasi Genetic Algorithm

Dalam pembelajaran mesin, beberapa eksperimen telah dilakukan oleh banyak peneliti dengan tujuan meningkatkan akurasi prediksi dengan mengkombinasikan sejumlah metoda (Karimi & Gandomani, 2021; Saljoughinejad & Khatibi, n.d.). Peneliti juga menambahkan sejumlah parameter tertentu dalam algoritma pembelajaran yang harus ditentukan sebelum proses pelatihan dimulai (Shah et al., 2020). Parameter ini dikenal sebagai hiperparameter, yang berbeda dari parameter model seperti bobot dan bias dalam jaringan saraf, yang diperoleh melalui proses pelatihan. Hiperparameter perlu ditetapkan terlebih dahulu karena berperan dalam mengontrol bagaimana model belajar dan beradaptasi dengan data (Nguyen & Liu, 2025; Villalobos-Arias et al., 2020).

Genetic Algorithm (GA) adalah sebuah teknik untuk optimasi global yang diusulkan oleh John Holland pada tahun 1970-an. GA didasarkan pada mekanisme genetik yang digunakan untuk memecahkan masalah optimasi [22], [30], [33]. Berikut ini adalah tahapan kunci utama dalam Algoritma Genetika: (i) inisialisasi populasi, (ii) evaluasi fitur kebugaran, dan (ii) generasi populasi baru. Algoritma genetika dengan gen yang disebut kromosom membawa informasi yang diproduksi secara acak. Kromosom menghasilkan populasi asli, yang memiliki jumlah individu yang konstan. Semua kromosom dievaluasi menggunakan fungsi fitness untuk memberikan solusi optimal yang paling mendekati [30]. GA menerapkan banyak operasi matematika seperti mutasi, crossover, dan fittest. Banyak masalah optimasi yang telah berhasil diselesaikan dengan menggunakan GA.

Beberapa peneliti mengkombinasi berbagai metoda dalam memprediksi Upaya dalam perangkat lunak. Metode CBR-GA membuktikan bahwa penerapan algoritma GA mampu

menemukan kombinasi parameter CBR yang paling optimal sehingga menghasilkan peningkatan akurasi (Hameed et al., 2023). Chahar mengkombinasi GA dengan ANN Dimana akurasi yang diperoleh 91,3% naik 8,9% dibandingan dari metoda yang lainnya (Chahar & Bhatia, 2022). Temuan ini memberikan manfaat nyata bagi manajer proyek, khususnya pada tahap perencanaan awal anggaran, karena dapat digunakan untuk memperkirakan kebutuhan usaha serta mendukung pengendalian biaya proyek secara lebih efektif (*PDF*, n.d.).

Evaluasi Kinerja Model

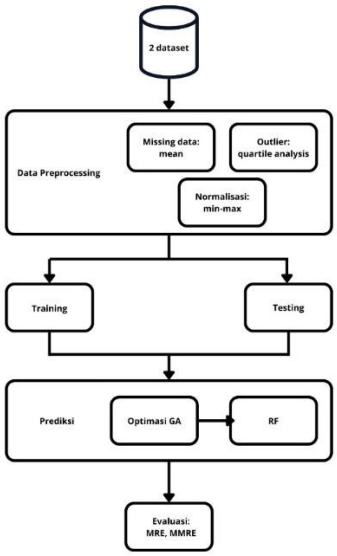
Evaluasi model prediksi dalam *machine learning* adalah proses menilai kinerja model yang telah dilatih untuk memprediksi kelas atau label tertentu dari data yang diberikan(Valero-Carreras et al., 2023). Model klasifikasi berfungsi untuk mengkategorikan data ke dalam kelas-kelas yang telah ditentukan sebelumnya, seperti 'ya' atau 'tidak', 'positif' atau 'negatif'. Evaluasi ini penting untuk memastikan model tidak hanya bekerja dengan baik pada data latih, tetapi juga dapat menggeneralisasi dengan baik pada data yang belum pernah dilihat. Beberapa metrik evaluasi sering digunakan untuk mengukur seberapa baik model memprediksi label yang benar, termasuk *Magnitude of Relative Error* (MRE) dan *Mean Magnitudo of Relative Error* (MMRE)

3. METODE PENELITIAN

Penelitian ini bertujuan untuk mengestimasi SEE dengan menggunakan RF yang dioptimasi dengan dengan GA. Adapun skema penelitiannya Adalah sebagai berikut:

Dataset

Pada penelitian dataset yang digunakan ada 2 yakni Cina dan Desharnais dataset dalam pengembangan perangkat lunak



Gambar 1. Model Yang Diusulkan

Preprocessing Data

Proses *preprocessing* data pada 2 dataset pengembangan perangkat lunak sangat penting untuk memastikan kualitas data sebelum diterapkan pada algoritma prediksi pada ML. Langkah-langkah yang dilakukan mencakup penanganan nilai hilang, konversi variabel kategorikal ke bentuk numerik, serta normalisasi fitur.

Dalam menangani *missing value*, digunakan metode rata-rata (*mean imputation*) untuk menggantikan nilai yang hilang agar tetap mencerminkan distribusi data asli. Untuk normalisasi, diterapkan teknik *Min-Max Scaling*, yang mengubah nilai fitur ke dalam rentang tertentu guna memastikan model tidak bias terhadap skala variabel. Sementara itu, deteksi dan penanganan *outlier* dilakukan menggunakan *quartile analysis* (IQR method) agar data tetap representatif tanpa dipengaruhi nilai ekstrem.

Pembagian Data (*Data Splitting*)

Dalam pengembangan perangkat lunak, pembagian data menjadi *training set* dan *testing set* merupakan langkah penting untuk mengevaluasi kinerja algoritma ML secara efektif. *Training set* digunakan untuk melatih model agar mampu mengenali pola dalam data, sedangkan *testing set* berfungsi untuk mengukur akurasi model terhadap data yang belum pernah dilihat sebelumnya. Pembagian ini membantu mencegah *overfitting*, yaitu kondisi di mana model terlalu menyesuaikan diri dengan data latih sehingga kurang mampu melakukan generalisasi pada data baru.

Rasio 70:30 menyediakan lebih banyak data untuk pengujian, sehingga lebih cocok untuk analisis performa model yang lebih mendalam. Pemilihan rasio ini bergantung pada ukuran dataset serta kebutuhan analisis yang dilakukan.

Estimasi Perangkat Lunak

Selanjutnya data yang sudah terstandarisai dilakukan estimasi dengan menggunakan RF yang selanjutnya akan dilakukan optimasi.

Evaluasi Kinerja Model

Evaluasi dilakukan dengan menggunakan metrik MRE dan MMRE untuk menguji model yang diusulkan.

4. HASIL DAN PEMBAHASAN

Analisa Dataset

Penelitian ini menggunakan 2 dataset yakni Cina dan Desharnais. Untuk dataset Cina terdiri 19 fitur yang akhirnya hanya 11 fitur yang berperan dalam pengembangan perangkat lunak, sedangkan untuk dataset Desharnais menggunakan semua fitur (11 fitur), dengan tipe

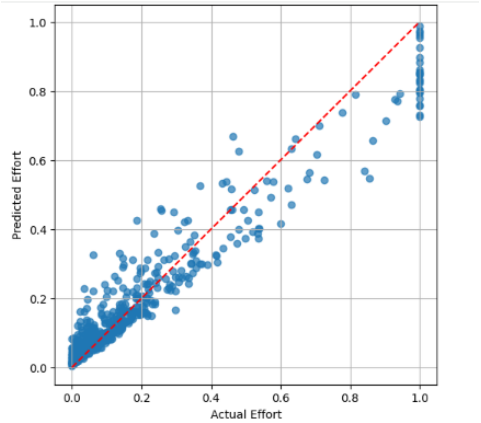
untuk semua fitur numerik, kecuali bahasa pemograman yang digunakan (dataset Desharnais). Adapujn fiturnya adalah sebagai berikut:

Tabel 1. Fitur Pada Dataset Cina dan Desharnais

NO	Fitur Dataset Cina	Fitur Dataset Desharnais
1	Input	TeamExp
2	Output	ManagerExp
3	Enquiry	YearEnd
4	File	Length
5	Interface	Effort
6	Add	Transaction
7	Change	Entities
8	Deleted	PointAdjust
9	Resource	Envergure
10	Duration	PontsNonAjust
11	Effortt	Language

Evaluasi Model Yang Diusulkan

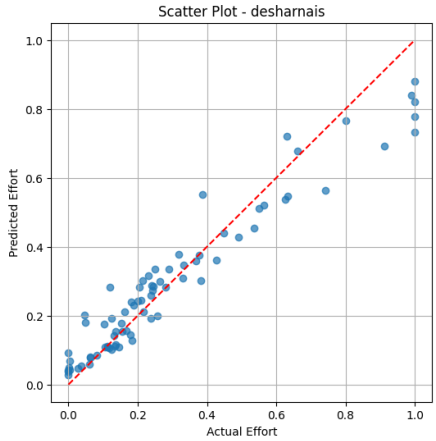
Dalam menganalisis aktual dan hasil prediksi, penulis menggunakan scatter plot.



Gambar 1. Scatter Plot Dataset Cina

Berdasarkan scatter plot yang ditampilkan, dapat dilihat hubungan antara usaha pengembangan aktual dengan usaha pengembangan yang merupakan hasil dari prediksi pada dataset Desharnais. Gambar 2 menunjukkan adanya hubungan positif antara kedua variabel tersebut, dimana penyebaran titik-titik biru cenderung mengikuti garis diagonal merah putus-putus yang merupakan garis prediksi ideal. Hal yang menarik untuk diperhatikan adalah sebagian besar data terkonsentrasi pada rentang usaha pengembangan perangkat lunak rendah, yang menunjukkan bahwa mayoritas proyek dalam dataset ini termasuk proyek berskala kecil hingga menengah. Pada rentang nilai tersebut, titik-titik data tersebar cukup rapat di sekitar garis referensi, sehingga dapat disimpulkan bahwa model prediksi memiliki tingkat akurasi yang cukup baik untuk proyek-proyek berukuran kecil. Namun demikian, seiring dengan meningkatnya nilai actual effort, pola penyebaran data mengalami perubahan yang cukup signifikan dimana pada rentang usaha pengembangan perangkat lunak tinggi, jumlah titik data

menjadi lebih sedikit dan tingkat variasinya semakin besar, dengan beberapa titik menyebar cukup jauh dari garis ideal baik di atas maupun di bawah garis tersebut. Kondisi ini mengindikasikan bahwa model mengalami kesulitan yang lebih besar dalam melakukan prediksi terhadap proyek-proyek berskala besar atau kompleks, dimana ketidakseimbangan jumlah data antara proyek kecil dan besar menyebabkan model kurang terlatih dengan baik untuk kasus-kasus proyek berskala besar. Adanya beberapa outlier, khususnya titik-titik yang berada jauh di atas garis referensi pada area usaha pengembangan perangkat lunak tinggi, menunjukkan terjadinya prediksi yang terlalu tinggi yang perlu dilakukan analisis lebih lanjut guna meningkatkan kualitas dan keandalan model dalam mengestimasi proyek-proyek dengan skala yang lebih besar



Gambar 2. Scatter Plot Dataset Desharnais

Pada dataset Desharnais untuk estimasi proyek perangkat lunak. Terlihat adanya korelasi positif yang kuat antara kedua variabel, di mana sebaran titik-titik biru cenderung mengikuti garis diagonal merah putus-putus yang merepresentasikan prediksi yang cukup akurat. Hal ini mengindikasikan bahwa model prediksi yang digunakan memiliki kemampuan yang cukup baik dalam mengestimasi usaha pengembangan perangkat lunak. Sebagian besar titik data berada relatif dekat dengan garis referensi, menunjukkan bahwa prediksi model tidak terlalu jauh menyimpang dari nilai aktual.

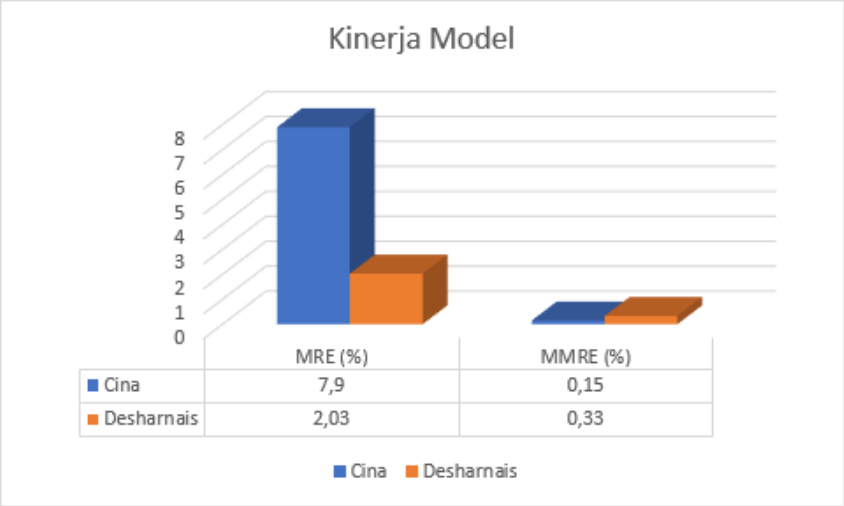
Namun demikian, terdapat beberapa hal yang perlu diperhatikan dari pola sebaran data. Pada range usaha pengembangan perangkat lunak rendah hingga menengah, prediksi cenderung lebih akurat dengan titik-titik yang lebih rapat ke garis ideal. Sebaliknya, pada range usaha pengembangan perangkat lunak tinggi, terlihat variabilitas yang lebih besar dengan beberapa titik yang menyebar cukup jauh dari garis referensi. Gejala ini mengindikasikan bahwa model mengalami kesulitan dalam memprediksi proyek-proyek besar atau kompleks dengan akurasi yang sama seperti proyek kecil. Adanya beberapa titik yang berada jauh di atas

atau di bawah garis diagonal juga menunjukkan keberadaan outlier atau kasus-kasus khusus yang mungkin memerlukan investigasi lebih lanjut untuk meningkatkan performa model secara keseluruhan.

Kedua model juga menunjukkan bahwa hubungan variasi antar data sangat baik dengan nilai R^2 sebesar 91%.

Kinerja Model

Penelitian ini menggunakan 2 dataset yakni Cina Desharnais, dimana hasilnya dapat dilihat pada gambar 4 dibawah ini:



Gambar 3. Kinerja Model (MRE dan MMRE)

Pada gambar 4, dapat diketahui bahwa Hasil pengujian menunjukkan bahwa model RF-GA mencapai kinerja yang lebih baik pada dataset China (MMRE 7,90%) dibandingkan dengan dataset Desharnais (MMRE 22,60%). Perbedaan kinerja ini dapat dijelaskan melalui proses pemilihan fitur yang diterapkan pada dataset China. Pada dataset China, fitur-fitur yang tidak relevan seperti PDR-AFT dan fitur lainnya telah dieliminasi melalui proses pemilihan fitur, sehingga model hanya menggunakan fitur-fitur yang berkontribusi signifikan terhadap prediksi. Sebaliknya, dataset Desharnais menggunakan seluruh fitur tanpa proses seleksi, yang berpotensi mengandung *noise* dan fitur-fitur yang tidak relevan, sehingga menurunkan akurasi model. Hal ini mengonfirmasi pentingnya tahap preprocessing, khususnya pemilihan fitur, dalam meningkatkan kinerja model estimasi usaha perangkat lunak

Perbandingan Kinerja Dengan Model

Peneliti juga membandingan kinerja model dengan peneliti terdahulu, dimana hasilnya terlihat pada tabel 4.

Tabel 1. Perbandingan Kinerja Model

Ref	Dataset	Metoda	MMRE %	MRE %
(Vanathi et al., 2024)	NASA, NASA 92 dan NASA 93	Fuzzy Logic	1,01	6,55
(Rashid et al., 2025)	COCOMO	ANN	22	-
(Vescan & Barac-Antonescu, 2025)	QUES and UIMS.	NN	6	-
(Abdelali et al., 2019)	ISBSG	RF	1,29	-
(Ali et al., 2023)	ISBSG	ANN	-	0,03
(Nhung et al., 2022)	Privat	Extension of Optimizing Correction Factors	-	6
Model yang Diusulkan	Cina	RF-GA	7,90	0,15
Model yang Diusulkan	Desharnais	RF-GA	2,03	0,33

Berdasarkan Tabel 3, model yang diusulkan untuk dataset yang memiliki fitur kecil memiliki tingkat eror yang rendah, sehingga bisa disimpulkan model ini handal untuk dataset yang memiliki dimensi rendah. Untuk dataset yang dimensi tinggi ANN ternyata lebih baik.

5. KESIMPULAN DAN SARAN

Penelitian ini menggunakan RF sebagai metoda prediksi dalam mengestimasi usaha pengembangan perangkat lunak, dimanauntuk mendapatkan hasil yang optimal maka digunakan GA untuk *hyper* parameternya. Untuk menguji model, maka digunakan 2 dataset yang berbeda yakni Cina dan Desharnais, dengan fitur dan tipe data yang berbeda, akan tetapi jumlah fitur kedua dataset ini sama. Hasil yang didapat adalah dataset Cina yang dilakukan seleksi fitur memiliki hasil yang lebih baik ()dibandingkan dataset Desharnais

Selanjutnya agar model ini lebih teruji, maka diperlukan datset yang berbeda yang memiliki banyak fitur dan data. Perlu juga menguji model dengan karkateristik yang memiliki missing value yang cukup banyak.

UCAPAN TERIMA KASIH

Peneliti mengucapkan terima kasih kepada Yayasan Dinamika Bangsa atas bantuan dalam pembiayaan penelitian ini.

DAFTAR REFERENSI

A G, P. V., K, A. K., & Varadarajan, V. (2021). Estimating Software Development Efforts Using a Random Forest-Based Stacked Ensemble Approach. *Electronics*, *10*(10), 1195. https://doi.org/10.3390/electronics10101195

Abdelali, Z., Mustapha, H., & Abdelwahed, N. (2019). Investigating the use of random forest in software effort estimation. *Procedia Computer Science*, *148*, 343–352. https://doi.org/10.1016/j.procs.2019.01.042

Abdu, A., Zhai, Z., Abdo, H. A., Lee, S., Al-masni, M. A., Gu, Y. H., & Algabri, R. (2025). Cross-project software defect prediction based on the reduction and hybridization of software metrics. *Alexandria Engineering Journal*, *112*, 161–176. https://doi.org/10.1016/j.aej.2024.10.034

Ali, S. S., Ren, J., Zhang, K., Wu, J., & Liu, C. (2023). Heterogeneous Ensemble Model to Optimize Software Effort Estimation Accuracy. *IEEE Access*, *11*, 27759–27792. https://doi.org/10.1109/ACCESS.2023.3256533

Chahar, V., & Bhatia, P. K. (2022). Performance Analysis of Software Test Effort Estimation using Genetic Algorithm and Neural Network. *International Journal of Advanced Computer Science and Applications*, *13*(10). https://doi.org/10.14569/IJACSA.2022.0131045

Dai, H., Xi, J., & Dai, H.-L. (2025). Improve cross-project just-in-time defect prediction with dynamic transfer learning. *Journal of Systems and Software*, *219*, 112214. https://doi.org/10.1016/j.jss.2024.112214

Denard, S., Ertas, A., Mengel, S., & Ekwaro-Osire, S. (2020). Development Cycle Modeling: Resource Estimation. *Applied Sciences*, *10*(14), 5013. https://doi.org/10.3390/app10145013

Fernandez-Diego, M., Mendez, E. R., Gonzalez-Ladron-De-Guevara, F., Abrahao, S., & Insfran, E. (2020). An Update on Effort Estimation in Agile Software Development: A Systematic Literature Review. *IEEE Access*, *8*, 166768–166800. https://doi.org/10.1109/ACCESS.2020.3021664

Hameed, S., Elsheikh, Y., & Azzeh, M. (2023). An optimized case-based software project effort estimation using genetic algorithm. *Information and Software Technology*, *153*, 107088. https://doi.org/10.1016/j.infsof.2022.107088

Jadhav, A., Kaur, M., & Akter, F. (2022). Evolution of Software Development Effort and Cost Estimation Techniques: Five Decades Study Using Automated Text Mining Approach. *Mathematical Problems in Engineering*, *2022*, 1–17. https://doi.org/10.1155/2022/5782587

Karimi, A., & Gandomani, T. J. (2021). Software development effort estimation modeling using a combination of fuzzy-neural network and differential evolution algorithm. *International Journal of Electrical and Computer Engineering (IJECE)*, *11*(1), 707. https://doi.org/10.11591/ijece.v11i1.pp707-715

Khan, M. S., Jabeen, F., Ghouzali, S., Rehman, Z., Naz, S., & Abdul, W. (2021). Metaheuristic Algorithms in Optimizing Deep Neural Network Model for Software Effort Estimation. *IEEE Access*, *9*, 60309–60327. https://doi.org/10.1109/ACCESS.2021.3072380

Liu, H., Li, M., Cheng, J. C. P., Anumba, C. J., & Xia, L. (2025). Actual construction cost prediction using hypergraph deep learning techniques. *Advanced Engineering Informatics*, *65*, 103187. https://doi.org/10.1016/j.aei.2025.103187

Liu, Y., Meng, Q., Chen, K., & Shen, Z. (2025). ALB-TP: Adaptive Load Balancing based on Traffic Prediction using GRU-Attention for Software-Defined DCNs. *Journal of Network and Computer Applications*, *236*, 104103. https://doi.org/10.1016/j.jnca.2024.104103

Mahmood, Y., Kama, N., & Azmi, A. (2020). A systematic review of studies on use case points and expert-based estimation of software development effort. *Journal of Software: Evolution and Process*, *32*(7), e2245. https://doi.org/10.1002/smr.2245

Mahmood, Y., Kama, N., Azmi, A., Khan, A. S., & Ali, M. (2022). Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation. *Software: Practice and Experience*, *52*(1), 39–65. https://doi.org/10.1002/spe.3009

Malhotra, R., & Singh, P. (2025). DHG-BiGRU: Dual-attention based hierarchical gated BiGRU for software defect prediction. *Information and Software Technology*, *179*, 107646. https://doi.org/10.1016/j.infsof.2024.107646

Nashaat, M., & Miller, J. (2025). Refining software defect prediction through attentive neural models for code understanding. *Journal of Systems and Software*, *220*, 112266. https://doi.org/10.1016/j.jss.2024.112266

Nassif, A. B., Azzeh, M., Idri, A., & Abran, A. (2019). Software Development Effort Estimation Using Regression Fuzzy Models. *Computational Intelligence and Neuroscience*, *2019*, 1–17. https://doi.org/10.1155/2019/8367214

Nassif, A. B., Ho, D., & Capretz, L. F. (2013). Towards an early software estimation using log-linear regression and a multilayer perceptron model. *Journal of Systems and Software*, *86*(1), 144–160. https://doi.org/10.1016/j.jss.2012.07.050

Nguyen, X. D. J., & Liu, Y. A. (2025). Methodology for hyperparameter tuning of deep neural networks for efficient and accurate molecular property prediction. *Computers & Chemical Engineering*, *193*, 108928. https://doi.org/10.1016/j.compchemeng.2024.108928

Nhung, H. L. T. K., Van Hai, V., Silhavy, R., Prokopova, Z., & Silhavy, P. (2022). Parametric Software Effort Estimation Based on Optimizing Correction Factors and Multiple Linear Regression. *IEEE Access*, *10*, 2963–2986. https://doi.org/10.1109/ACCESS.2021.3139183

Park, B. K., & Kim, R. Y. C. (2020). Effort Estimation Approach through Extracting Use Cases via Informal Requirement Specifications. *Applied Sciences*, *10*(9), 3044. https://doi.org/10.3390/app10093044

Parmar, A., Katariya, R., & Patel, V. (2019). A Review on Random Forest: An Ensemble Classifier. *Lecture Notes on Data Engineering and Communications Technologies*, *26*, 758–763. https://doi.org/10.1007/978-3-030-03146-6_86

PDF. (n.d.).

Phannachitta, P. (2020). On an optimal analogy-based software effort estimation. *Information and Software Technology*, *125*, 106330. https://doi.org/10.1016/j.infsof.2020.106330

Probst, P., Wright, M. N., & Boulesteix, A. L. (2019). Hyperparameters and tuning strategies for random forest. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *9*(3), 1–15. https://doi.org/10.1002/widm.1301

Rankovic, N., Rankovic, D., Ivanovic, M., & Lazic, L. (2021). A New Approach to Software Effort Estimation Using Different Artificial Neural Network Architectures and Taguchi Orthogonal Arrays. *IEEE Access*, *9*, 26926–26936. https://doi.org/10.1109/ACCESS.2021.3057807

Rashid, C. H., Shafi, I., Khattak, B. H. A., Safran, M., Alfarhood, S., & Ashraf, I. (2025). ANN-based software cost estimation with input from COCOMO: CANN model. *Alexandria Engineering Journal*, *113*, 681–694. https://doi.org/10.1016/j.aej.2024.11.042

Rosa, W., & Jardine, S. (2025). A novel method to early agile effort estimation through functional initiatives. *Journal of Systems and Software*, *223*, 112302. https://doi.org/10.1016/j.jss.2024.112302

Sakhrawi, Z., Sellami, A., & Bouassida, N. (2022). Software enhancement effort estimation using correlation-based feature selection and stacking ensemble method. *Cluster Computing*, *25*(4), 2779–2792. https://doi.org/10.1007/s10586-021-03447-5

Saljoughinejad, R., & Khatibi, V. (n.d.). *A New Optimized Hybrid Model Based on COCOMO to Increase the Accuracy of Software Cost Estimation*.

Shah, M. A., Jawawi, D. N. A., Isa, M. A., Younas, M., Abdelmaboud, A., & Sholichin, F. (2020). Ensembling Artificial Bee Colony With Analogy-Based Estimation to Improve Software Development Effort Prediction. *IEEE Access*, *8*, 58402–58415. https://doi.org/10.1109/ACCESS.2020.2980236

Sharma, A., & Chaudhary, N. (2023). Prediction of Software Effort by Using Non-Linear Power Regression for Heterogeneous Projects Based on Use case Points and Lines of code. *Procedia Computer Science*, *218*, 1601–1611. https://doi.org/10.1016/j.procs.2023.01.138

Sharma, A., & Kushwaha, D. S. (2012). Estimation of Software Development Effort from Requirements Based Complexity. *Procedia Technology*, *4*, 716–722. https://doi.org/10.1016/j.protcy.2012.05.116

Shi, J., Lian, Y., Salzmann, C., & Jones, C. N. (2025). Adaptive data-driven prediction in a building control hierarchy: A case study of demand response in Switzerland. *Energy and Buildings*, *333*, 115498. https://doi.org/10.1016/j.enbuild.2025.115498

Singal, P., Kumari, A. C., & Sharma, P. (2020). Estimation of Software Development Effort: A Differential Evolution Approach. *Procedia Computer Science*, *167*, 2643–2652. https://doi.org/10.1016/j.procs.2020.03.343

Suresh Kumar, P., Behera, H. S., K, A. K., Nayak, J., & Naik, B. (2020). Advancement from neural networks to deep learning in software effort estimation: Perspective of two decades. *Computer Science Review*, *38*, 100288. https://doi.org/10.1016/j.cosrev.2020.100288

Tan, A. J. J., Chong, C. Y., & Aleti, A. (2024). REARRANGE: Effort estimation approach for software clustering-based remodularisation. *Information and Software Technology*, *176*, 107567. https://doi.org/10.1016/j.infsof.2024.107567

Valero-Carreras, D., Alcaraz, J., & Landete, M. (2023). Comparing two SVM models through different metrics based on the confusion matrix. *Computers and Operations Research*, *152*(April 2022), 106131. https://doi.org/10.1016/j.cor.2022.106131

Vanathi, D., Anusha, K., Ahilan, A., & Salinda Eveline Suniram, A. (2024). Software cost and effort estimation using dragonfly whale optimized multilayer perceptron neural network. *Alexandria Engineering Journal*, *103*, 30–37. https://doi.org/10.1016/j.aej.2024.04.043

Vescan, A., & Barac-Antonescu, D. (2025). Software maintainability prediction based on change metric using neural network models. *Engineering Applications of Artificial Intelligence*, *144*, 110032. https://doi.org/10.1016/j.engappai.2025.110032

Villalobos-Arias, L., Quesada-López, C., Guevara-Coto, J., Martínez, A., & Jenkins, M. (2020). Evaluating hyper-parameter tuning using random search in support vector machines for software effort estimation. *Proceedings of the 16th ACM International Conference on Predictive Models and Data Analytics in Software Engineering*, 31–40. https://doi.org/10.1145/3416508.3417121