BAB II

TINJAUAN PUSTAKA

Berdasarkan latar belakang, penelitian ini berfokus pada mengadaptasi dan mengoptimalkan metodologi prediksi performa. Optimalisasi ini bertujuan untuk memastikan model dapat melakukan generalisasi secara akurat SPEC CPU2017. Perbedaan antara teknologi / metode yang digunakan pada penelitian dari penulis dan penelitian Lu et al. dapat dilihat pada tabel 2.1:

Tabel 2.1 Teknologi atau Metode yang dikembangkan

No	Teknologi/Metode	Deskripsi Lu et al.	Deskripsi Penulis	Status
1	Dataset	SPEC CPU2006 = 48,129 data, SPEC CPU2017 = 1,941 data (Total ~ 50, 000 data).	Seluruh data pada SPEC CPU2017 = 45,249 data.	Pengembangan
2	Jenis *Benchmark*	CINT2017, CFP2017, CINT2017rate, CFP2017rate (dipisahkan per subset).	CINT2017, CFP2017, CINT2017rate, CFP2017rate (dipisahkan per subset).	Adopsi.
3	Fitur Input	System, Number of Cores, Number of Chips, Number of Cores per Chip, Number of Threads per Core, Processor, Processor MHz, Processor Characteristics, 1st Level Cache, 2nd Level Cache, 3rd Level Cache, Memory, Compiler.	Benchmark, Processor, # Cores, # Chips, # Enabled Threads Per Core, Processor MHz, 1st Level Cache, 2nd Level Cache, 3rd Level Cache, Memory, Storage, Operating System, File System, Compiler.	Pengembangan
4	Fitur Output	Result.	Base Result dan Peak Result	Pengembangan
5	Preprocessing Konversi	Konversi string ke numerik (pengelompokkan prosesor berdasarkan	Konversi string ke numerik (pengelompokkan prosesor berdasarkan	Pengembangan

		family-model type-iteration dan diberikan *class number*, pengelompokan memori berdasarkan total memori (GB) dan jumlah modul, dan compiler diberikan *class number* serta diurut berdasarkan versinya).	*family-model type-iteration* dan diberikan *class number*, pengelompokan memori berdasarkan total memori (GB) dan jumlah modul, dan compiler diberikan *class number* serta diurut berdasarkan versinya, Operating System, Storage, File System, dan Cache) yang dapat adaptif untuk spesifikasi terbaru. Fitur kategorikal di-encode menggunakan Target Mean Encoding.	
6	Preprocessing Scaling	Tidak disebutkan	Transformasi menggunakan logaritmik dan menggunakan StandardScaler.	Pengembangan
7	Analisis fitur	Principal Component Analysis (PCA) menggunakan FactoMineR untuk kombinasi fitur numerik dan kategorikal.	Menggunakan Spearman Correlation Filtering, yang kemudian dilanjuti dengan Recursive Feature Elimination with Cross-Validation (RFECV)	Pengembangan
8	Split data Testing:Training	50:50	5-Fold dengan rasio Training:Testing 80:20 per fold. Dilakukan dalam setiap fold.	Pengembangan
9	Model & Struktur	Artificial Neural Networks (ANN) dengan 13 neuron input, 3 hidden layer (50-100-50), 1 neuron output, dan 50 epoch.	RandomForestRegressor. Model dilatih dengan parameter n_estimators=50 dan max_depth=15.	Pengembangan
10	Algoritma Pelatihan	*Backpropagation*	*Bootstrap Aggregation*	Pengembangan

		dengan *train-test-revise approach.*	(*Bagging*) dan *Decision Tree*.	
11	Pengujian Model	Membandingkan Result dari dataset dengan hasil prediksi Result dari model.	Evaluasi menggunakan 5-Fold Cross-Validation dengan metrik MAPE dan R^2 pada nilai target. Dilakukan pengujian generalisasi data uji yang dipisah dari dataset utama.	Adopsi

2.1 Justifikasi Solusi

Justifikasi solusi disusun berdasarkan perbandingan metodologi antara penelitian Lu et al. dan metodologi yang diterapkan dari penelitian ini [1]. Setiap aspek teknologi/metode dievaluasi untuk menentukan apakah perlu dilakukan pengebangan atau adopsi. Berikut rincian dari justifikasi solusi berdasarkan tabel perbandingan diatas:

2.1.1 Dataset (Pengembangan)

Menurut Lu et al., Penelitian sebelumnya menggunakan dataset lama (SPEC CPU2006) dan jumlah data yang kecil pada dataset baru (SPEC CPU2017) [1]. Penelitian ini menggunakan seluruh data SPEC CPU2017 untuk menggantikan dataset yang lama. Peningkatan jumlah data ini berguna untuk memperkuat generalisasi model agar dapat mempelajari spesifikasi perangkat keras yang terbaru [8][15][12].

2.1.2 Jenis Benchmark (Adopsi)

Lu et al. sebelumnya memisahkan dataset berdasarkan tipe benchmark (CINT2017, CFP2017, CINT2017rate, CFP2017rate). Metode pemisahan ini dipertahankan karena efektif meningkatkan akurasi prediksi. Karakteristik beban kerja *Integer* dan *Floating Point* berbeda, hingga melatih model terpisah untuk setiap jenis benchmark mencegah bias dan *noise* pada proses pembelajaran [1][15].

2.1.3 Fitur Input (Pengembangan)

Meskipun fitur yang digunakan sama dengan 13 fitur yang di-*list* oleh Lu et al. [1], perluasan fitur (Penambahan Storage, File System, Operating System) ini dilakukan dataset SPEC CPU2017 menyediakan fitur lebih banyak dibandingkan generasi sebelumnya [15]. Fitur tersebut dipakai agar model mampu menangkap interaksi kompleks antara *hardware* dan *software* pada spesifikasi terbaru, dan dapat mengatasi resiko kegagalan memprediksi kinerja pada spesifikasi yang berbeda [13]. Dengan ini, fitur tambahan bertujuan untuk meningkatkan stabilitas prediksi pada pengujian perangkat keras dengan konfigurasi sistem yang diluar dari dataset.

2.1.4 Fitur Output (Pengembangan)

Meskipun terjadi penyesuaian nama, Lu et al. menggunakan fitur output "Result" [1], sedangkan penelitian ini menggunakan "Peak Result" Kedua dataset ini memiliki dua fitur output; SPEC CPU2006 bernama "Result" dan "Baseline", sedangkan SPEC CPU2017 bernama "Peak Result" dan "Base Result". "Baseline" dan "Base Result" bermakna hasil benchmark dengan performa yang standar, sedangkan "Result" dan "Peak Result" bermakna hasil benchmark dengan performa yang maksimal. Penggunaan "Base Result" dan "Peak Result" ini lebih relevan untuk studi kasus perbandingan performa prosesor guna mengetahui potensi maksimalnya [9][12][16].

2.1.5 Preprocessing Konversi (Pengembangan)

Penelitian terdahulu memiliki keterbatasan konversi *string* ke numerik hanya bekerja pada CPU yang ada di dataset [1]. Penelitian ini mengembangkan metodologi yang adaptif dan mampu konversi *string* dari CPU yang belum pernah dilihat pada model. Ini adalah solusi inti untuk menjawab masalah generalisasi model pada spesifikasi perangkat keras terbaru [15][17]. Selanjutnya, fitur kategorikal di-*encode* menjadi numerik menggunakan *Target Mean Encoding*, dengan tujuan memanfaatkan informasi dari *Peak Result* yang secara signifikan

meningkatkan robustness dan transferability model pada spesifikasi yang belum dikenal [18][19].

2.1.6 Preprocessing Scaling (Pengembangan)

Lu et al. tidak menyebutkan penggunaan teknik *scaling* [1]. Oleh karena itu, penelitian ini mengimplementasikan dua langkah *data preprocessing* untuk meningkatkan kualitas fitur: Pertama, menggunakan transformasi logaritmik yang memiliki distribusi *skewed* untuk normalisasi data. Kedua, menerapkan StandardScaler sebagai langkah terakhir dalam *preprocessing pipeline* untuk menstandarisasi fitur agar memiliki nilai rata-rata (mean) 0 dan varians 1 [15].

2.1.7 Analisis Fitur (Pengembangan)

Penelitian sebelumnya menggunakan metode PCA dan FactoMineR untuk kombinasi fitur untuk kombinasi fitur numerik dan kategorikal [1]. Untuk pengembangan selanjutnya, digunakan Spearman Correlation Filtering karena kemampuannya untuk ukuran nonparametrik untuk hubungan monoton antar fitur, menjadikan metode yang kuat untuk data spesifikasi *hardware* yang cenderung *skewed* [15][19]. Kemudian, dilanjutkan dengan Recursive Feature Elimination with Cross-Validation (RFECV). RFECV adalah metode *wrapper* yang menggunakan *cross-validation* untuk secara sistematis mengidentifikasi subset fitur optimal, memastikan *feature set* akhir yang digunakan memiliki relevansi dan daya prediksi yang maksimal [15][21].

2.1.8 Split Data Testing:Training (Pengembangan)

Penelitian sebelumnya menggunakan pembagian data *Training:Testing* dengan rasio 50:50 [1]. Untuk pengembangannya, digunakan validasi 5-fold Cross-Validation dengan rasio *Training:Testing* sebesar 80:20 per *fold*. Metode 5-Fold ini lebih bagus karena memastikan seluruh data digunakan untuk pelatihan dan pengujian secara bergantian, menghasilkan estimasi performa model yang tidak bias dan lebih andal [15][22].

2.1.9 Model & Struktur (Pengembangan)

Penelitian sebelumnya menggunakan model ANN dengan konfigurasi 13 input neuron, 50-100-50 *hidden layer*, 1 output neuron, dan 50 epoch. Untuk pengembangannya, model ini menggunakan model RF dengan parameter n_estimators = 50, dan max_depth=15. Model ini dipilih karena merupakan model *tree-based* yang stabil, efektif menangani data non-linear, interaksi fitur kompleks, dan kurang rentan terhadap *overfitting* [15][23][24].

2.1.10 Algoritma Pelatihan (Pengembangan)

Penelitian sebelumnya menggunakan algoritma *backpropagation* (*train-test-revise*) pada model ANN. Sebagai pengembangan metode, digunakan *Bootstrap aggregation* (*Bagging*) dan *Decision Tree* pada model RF. *Bagging* melatih banyak pohon keputusan secara independen pada subset data acak, kemudian menggabungkan prediksi melalui rata-rata. Metode ini efektif mengurangi varians dan meningkatkan stabilitas serta generalisasi akurasi prediksi [15][23][24].

2.1.11 Pengujian Model (Adopsi)

Penelitian sebelumnya umumnya melakukan pengujian model dengan membandingkan nilai benchmark aktual dengan hasil prediksi model untuk menilai hubungan antara spesifikasi perangkat keras dan performa yang dihasilkan. Penelitian ini mengadopsi pendekatan sebelumnya dengan membandingkan nilai aktual dan hasil prediksi menggunakan data uji yang dipisahkan dari dataset utama [1][15].

2.2 Tinjauan Teori

2.2.1 Benchmark

Benchmark adalah proses menjalankan program atau beban kerja pada sistem komputasi, seperti laptop dan PC, dengan tujuan mendapatkan dan mengukur indikator kinerja. Pendekatan ini secara fundamental memberikan evaluasi akurat mengenai performa perangkat pada beban kerja spesifik [25]. Penggunaan benchmark sangat krusial untuk memahami potensi kinerja perangkat dalam berbagai skenario beban kerja [3].

Contoh aplikasi benchmark populer meliputi PCMark10 dan 3DMark Time Spy. PCMark10 dirancang untuk mencakup berbagai jenis pekerjaan komputasi, mulai dari aktivitas sehari-hari, produktivitas kantor, hingga pengujian multimedia [6]. Sementara itu, 3DMark Time Spy fokus pada pengujian kinerja *rendering* 3D sistem [4].

Selain benchmark konvensional, terdapat pula benchmark berbasis *machine learning*. Metode ini memanfaatkan dataset yang berisi spesifikasi perangkat keras dan mengaplikasikan model *machine learning* untuk memprediksi skor kinerja, dengan membandingkan hasil prediksi dengan skor dari dataset yang ada [1].

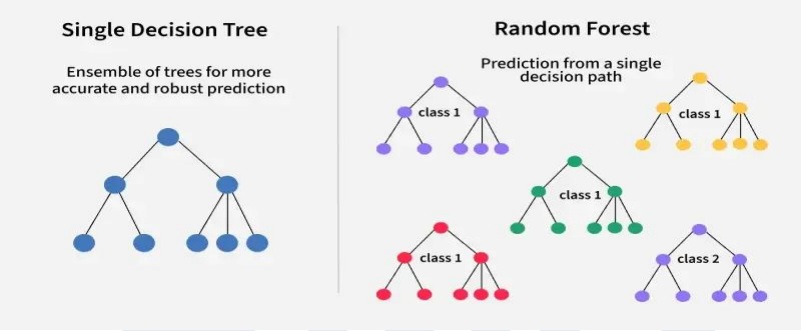
Perbedaan mendasar antara aplikasi benchmark populer dan metode berbasis *machine learning* terletak pada pendekatan dan efisiensinya. Aplikasi populer seperti PCMark10 dan 3DMark Time Spy memerlukan eksekusi fisik dari serangkaian pengujian yang kompleks dan memakan waktu, seringkali hingga puluhan menit bahkan berjam-jam, untuk menghasilkan skor performa. Proses ini melibatkan penggunaan sumber daya komputasi secara langsung pada sistem yang diuji [4][6].

Sebaliknya, benchmark berbasis *machine learning* menawarkan solusi yang jauh lebih cepat dan efisien. Alih-alih menjalankan tes secara langsung, metode ini memanfaatkan model prediktif yang telah dilatih menggunakan dataset besar berisi spesifikasi *hardware* dan skor *benchmark* historis. Dengan demikian, prediksi performa dapat

diperoleh secara instan hanya dengan memasukkan parameter spesifikasi perangkat, tanpa perlu eksekusi yang memakan waktu. Ini memungkinkan evaluasi yang cepat, bahkan untuk konfigurasi yang belum pernah diuji secara fisik, meskipun ada pengorbanan pada akurasi dibandingkan dengan pengujian langsung.

2.2.2 Random Forest

Random Forest (RF) merupakan model yang dibangun dari kumpulan *decision trees* yang bekerja sama untuk membuat prediksi. Kemudian, metode ini mencocokkan pohon-pohon tersebut pada dataset dan rata-rata hasil prediksinya untuk meningkatkan akurasi dan mengurangi *overfitting*. Setiap pohon cenderung *overfitting* pada sebagian data, tetapi varians keseluruhan dan jumlah *overfitting* dikurangi dengan rata-rata hasil dari beragam pohon [15][23][24].



Gambar 2.1 Diagram RF