TUGAS AKHIR – EF234801

ANALISIS PERBANDINGAN KINERJA DETEKSI DAN PENGENALAN DOKUMEN SCAN MENGGUNAKAN MODEL OCR OPEN-SOURCE

DANIEL HERMAWAN

NRP 5025201087

Dosen Pembimbing

Dini Adni Navastara, S.Kom., M.Sc.

NIP 198510172015042001

Prof. Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM.

NIP 198106202005011003

Program Studi S1 Teknik Informatika

Departemen Teknik Informatika

Fakultas Teknologi Elektro dan Informatika Cerdas (FTEIC)

Institut Teknologi Sepuluh Nopember

Surabaya

2026

TUGAS AKHIR – EF234801

ANALISIS PERBANDINGAN KINERJA DETEKSI DAN PENGENALAN DOKUMEN SCAN MENGGUNAKAN MODEL OCR OPEN-SOURCE

DANIEL HERMAWAN

NRP 5025201087

Dosen Pembimbing

Dini Adni Navastara, S.Kom., M.Sc.

NIP 198510172015042001

Prof. Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM.

NIP 198106202005011003

Program Studi S1 Teknik Informatika

Departemen Teknik Informatika

Fakultas Teknologi Elektro dan Informatika Cerdas (FTEIC)

Institut Teknologi Sepuluh Nopember

Surabaya

2026

FINAL PROJECT – EF234801

CHARACTER DETECTION AND RECOGNITION IN SCANNED DOCUMENTS PERFORMANCE COMPARATIVE ANALYSIS USING OPEN-SOURCE OCR MODEL

DANIEL HERMAWAN

NRP 5025201087

Advisor

Dini Adni Navastara, S.Kom., M.Sc.

NIP 198510172015042001

Prof. Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM.

NIP 198106202005011003

Study Program Informatics

Department of Informatics

Faculty of ELECTICS

Institut Teknologi Sepuluh Nopember

Surabaya

2026

LEMBAR PENGESAHAN

ANALISIS PERBANDINGAN KINERJA DETEKSI DAN PENGENALAN DOKUMEN SCAN MENGGUNAKAN MODEL OCR OPEN-SOURCE

TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat

memperoleh gelar Sarjana Komputer pada

Program Studi S-1 Teknik Informatika

Departemen Teknik Informatika

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Oleh : **DANIEL HERMAWAN**

NRP. 5025201087;

Disetujui oleh Tim Penguji Tugas Akhir :

1. Dini Adni Navastara, S.Kom., M.Sc. — Pembimbing
2. Prof. Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM. — Ko-pembimbing
3. Prof. Dr. Eng. Nanik Suciati, S.Kom., M.Kom. — Penguji
4. Imam Mustafa Kamal, S.ST., Ph.D. — Penguji

SURABAYA

Januari, 2026

Halaman ini sengaja dikosongkan.

APPROVAL SHEET

CHARACTER DETECTION AND RECOGNITION IN SCANNED DOCUMENTS PERFORMANCE COMPARATIVE ANALYSIS USING OPEN-SOURCE OCR MODEL

FINAL PROJECT

Submitted to fulfill one of the requirements

for obtaining a Bachelor of Computer Science degree at

Undergraduate Study Program of Informatics

Department of Informatics

Faculty of Intelligent Electrical and Informatics Technology

Institut Teknologi Sepuluh Nopember

By: **DANIEL HERMAWAN**

NRP. 5025201087

Approved by Final Project Examiner Team:

1. Dini Adni Navastara, S.Kom., M.Sc. — Advisor
2. Prof. Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM. — Co-advisor
3. Prof. Dr. Eng. Nanik Suciati, S.Kom., M.Kom. — Examiner 1
4. Imam Mustafa Kamal, S.ST., Ph.D. — Examiner 2

SURABAYA

January, 2026

This page is intentionally left blank.

PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini :

Nama mahasiswa / NRP	:	Daniel Hermawan / 5025201087
Program studi	:	S-1 Teknik Informatika
Dosen Pembimbing 1 / NIP	:	Dini Adni Navastara, S.Kom., M.Sc. / 198510172015042001
Dosen Pembimbing 2 / NIP	:	Prof. Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM. / 198106202005011003

dengan ini menyatakan bahwa Tugas Akhir dengan judul "**ANALISIS PERBANDINGAN KINERJA DETEKSI DAN PENGENALAN DOKUMEN SCAN MENGGUNAKAN MODEL OCR OPEN-SOURCE**" adalah hasil karya sendiri, bersifat orisinal, dan ditulis dengan mengikuti kaidah penulisan ilmiah.

Bilamana di kemudian hari ditemukan ketidaksesuaian dengan pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Institut Teknologi Sepuluh Nopember.

Mengetahui
Dosen Pembimbing 1

Dini Adni Navastara, S.Kom., M.Sc.

NIP. 198510172015042001

Dosen Pembimbing 2

Prof. Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM.
NIP. 198106202005011003

Surabaya, 20 Januari 2026
Mahasiswa

Daniel Hermawan
NRP. 5025201087

Halaman ini sengaja dikosongkan.

STATEMENT OF ORIGINALITY

The undersigned:

Student Name / Student ID	:	Daniel Hermawan / 5025201087
Study Program	:	S-1 Teknik Informatika
Advisor / Employee ID	:	Dini Adni Navastara, S.Kom., M.Sc. / 198510172015042001
Co-advisor / Employee ID	:	Prof. Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM. / 198106202005011003

hereby declares that the Final Project entitled "**CHARACTER DETECTION AND RECOGNITION IN SCANNED DOCUMENTS PERFORMANCE COMPARATIVE ANALYSIS USING OPEN-SOURCE OCR MODEL**" is my own work, is original, and was written in accordance with the rules of scientific writing.

If any discrepancies with this statement are found in the future, I am willing to accept sanctions in accordance with the provisions of Institut Teknologi Sepuluh Nopember.

Acknowledged
Advisor 1

Dini Adni Navastara, S.Kom., M.Sc.

NIP. 198510172015042001

Co-advisor

Prof. Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM.
NIP. 198106202005011003

Surabaya, 20 January 2026
Student

Daniel Hermawan
NRP. 5025201087

This page is intentionally left blank.

PERNYATAAN KODE ETIK
PENGGUNAAN AI GENERATIF

Code of Conduct Statement: Generative AI or AI-Assisted Usage

Saya yang bertanda tangan di bawah ini:
I, the undersigned:

Nama Mahasiswa / NRP *Full Name / Student ID*	:	Daniel Hermawan / 5025201087
Program Studi *Study Program*	:	S-1 Teknik Informatika
Judul Tugas Akhir *Final Project Title*	:	Analisis Perbandingan Kinerja Deteksi dan Pengenalan Dokumen Scan Menggunakan Model OCR Open-Source

dengan ini menyatakan bahwa pada Tugas Akhir dengan judul di atas tersebut:
hereby declare that in the Final Project with the above title:

No.	**Pernyataan** *Statement*	(✓)
1	Saya hanya menggunakan AI generatif sebagai alat bantu untuk memperbaiki tata bahasa. AI generatif tidak digunakan untuk membuat isi Tugas Akhir *I only used generative AI as a tool to improve the readability or language of the text in my Final Project. It was not used to generate a complete text of my work.*	✓
2	Saya telah memeriksa dan/atau memperbaiki seluruh bagian dari Tugas Akhir saya yang dibantu oleh AI generatif agar sesuai dengan baku mutu penulisan karya ilmiah. *I have reviewed and refined all aspects of my work that generative AI assists with, ensuring it adheres to the standards of academic writing.*	✓
3	Saya tidak menggunakan AI generatif untuk pembuatan data primer, grafik dan/atau tabel pada Tugas Akhir saya. *I did not use generative AI to generate primary data, figures, and/or tables in my work.*	✓
4	Saya telah memberikan atribusi/pengakuan terhadap alat AI yang digunakan, secara rinci pada suatu bagian pada lampiran. *I have acknowledged the use of generative AI in any part of the work in the specific appendix page.*	✓
5	Saya memastikan tidak ada plagiarisme, termasuk hal yang berasal dari penggunaan AI generatif. *I have ensured that there is no plagiarism issue in the work, including any parts generated by AI.*	✓

Surabaya, 20 Januari 2026

Mahasiswa

Daniel Hermawan
NRP. 5025201087

Halaman ini sengaja dikosongkan.

ABSTRAK

ANALISIS PERBANDINGAN KINERJA DETEKSI DAN PENGENALAN DOKUMEN SCAN MENGGUNAKAN MODEL OCR OPEN-SOURCE

Nama Mahasiswa / NRP	**:**	**Daniel Hermawan / 5025201087**
Departemen	**:**	**Teknik Informatika FTEIC – ITS**
Dosen Pembimbing 1	**:**	**Dini Adni Navastara, S.Kom., M.Sc.**
Dosen Pembimbing 2	**:**	**Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM**

Abstrak

Akurasi *Optical Character Recognition* (OCR) merupakan komponen penting dalam digitalisasi dokumen akademik, namun kemampuannya seringkali menurun jauh saat berhadapan dengan dokumen scan yang memiliki *noise*. Penelitian ini mencoba mencari cara terbaik untuk mengatasi masalah tersebut dengan membandingkan tiga pendekatan utama. Pertama, dilakukan pengujian kemampuan empat model OCR yang bekerja menggunakan arsitektur model masing-masing (*standalone*), baik untuk bagian deteksi (detektor) maupun bagian pengenalan (*recognizer*), untuk dijadikan sebagai acuan awal (*baseline*) kinerja dasar model OCR. Kedua, dilakukan pengujian dengan pendekatan OCR *Hybrid* di mana *Hybrid* memiliki arti penggabungan bagian deteksi (detektor) dari satu model dengan bagian pengenalan (*recognizer*) dari model lain, untuk melihat keberhasilan untuk kombinasi ini bisa bekerja lebih baik. Ketiga, kami juga menguji sebuah langkah perbaikan gambar (pra-pemrosesan) dengan harapan untuk bisa memperbaiki dan meningkatkan kinerja OCR *Hybrid* yang kurang baik. Saat diuji sendiri-sendiri (*standalone*), tidak ada satu pun model yang sempurna karena selalu ada *trade-off* antara kecepatan, akurasi, dan ketahanan terhadap *noise*. Ketika model-model tersebut digabungkan (OCR *Hybrid*), hasilnya ternyata tidak bisa ditebak. Kebanyakan kombinasi justru membuat hasilnya lebih buruk, namun juga terdapat beberapa kombinasi yang kinerjanya baik. Terakhir, langkah perbaikan gambar (pra-pemrosesan) terbukti sangat berguna untuk memperbaiki kombinasi OCR *Hybrid* yang gagal, tetapi juga bisa merusak hasil dari pasangan model yang sudah berhasil. Penelitian ini menunjukkan bahwa tidak ada satu cara yang paling baik dan benar untuk semua kondisi.

Kata Kunci : OCR, OCR *Hybrid*, Pra-pemrosesan Gambar, *Word Error Rate* (WER), *Noise* Dokumen, DocTR, Tesseract OCR, Paddle OCR, Easy OCR.

Halaman ini sengaja dikosongkan.

ABSTRACT

CHARACTER DETECTION AND RECOGNITION IN SCANNED DOCUMENTS PERFORMANCE COMPARATIVE ANALYSIS USING OPEN-SOURCE OCR MODEL

Student Name / NRP	**:**	**Daniel Hermawan / 5025201087**
Department	**:**	**Teknik Informatika FTEIC – ITS**
Advisor 1	**:**	**Dini Adni Navastara, S.Kom., M.Sc.**
Advisor 2	**:**	**Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D., IPM**

Abstract

The accuracy of *Optical Character Recognition* (OCR) is a crucial component in the digitization of academic documents, but its performance often suffers significantly when faced with noisy document scans. This study attempts to find the best way to address this issue by comparing three main approaches. First, we tested the capabilities of four OCR models working with their respective model architectures (*standalone*), for both the detection (detector) and recognition (*recognizer*) parts, to serve as a baseline for basic OCR model performance. Second, we tested a *Hybrid* OCR approach, where *Hybrid* means combining the detection (detector) part of one model with the recognition (*recognizer*) part of another model, to see if this combination can perform better. Third, we also tested an image enhancement step (pre-processing) in the hope of improving and enhancing the poor performance of *Hybrid* OCR. When tested individually (*standalone*), none of the models were perfect because there is always a trade-off between speed, accuracy, and resistance to *noise*. When these models were combined (*Hybrid* OCR), the results were unpredictable. While most combinations actually worsened the results, some performed well. Finally, the image enhancement (pre-processing) step proved very useful for improving failed *Hybrid* OCR combinations, but it could also damage the results of successful model pairs. This research shows that there is no single best approach for all situations.

Keywords : *OCR*, *Hybrid OCR, Image Pre-processing*, *Hybrid OCR*, *Word Error Rate* (WER), *Document Noise*, DocTR, TesseractOCR, PaddleOCR, EasyOCR.

This page is intentionally left blank.

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa atas rahmat dan karunia-Nya, penulis dapat menyelesaikan Tugas Akhir ini dengan judul **"Analisis Perbandingan Kinerja Deteksi dan Pengenalan Dokumen Scan Menggunakan Model OCR Open Source"** sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Komputer di Institut Teknologi Sepuluh Nopember.

Penulisan Tugas Akhir ini tidak akan terlaksana dengan baik tanpa bimbingan, dukungan, dan motivasi dari berbagai pihak. Oleh karena itu, penulis ingin menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. **Ibu Dini** selaku **Pembimbing 1** dan **Bapak Ary** selaku **Pembimbing 2** atas bimbingan, saran, serta kesabaran dalam mengarahkan penulis selama proses penyusunan Tugas Akhir ini.
2. **Ibu Nanik** dan **Bapak Kamal** selaku **Penguji Sidang** yang telah memberikan masukan, kritik, dan saran konstruktif demi perbaikan karya ini.
3. Keluarga, teman-teman, dan rekan-rekan seperjuangan yang selalu memberikan dukungan moral dan semangat kepada penulis.

Penulis menyadari bahwa Tugas Akhir ini masih jauh dari sempurna. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun untuk perbaikan di masa mendatang. Semoga hasil penelitian ini dapat bermanfaat bagi perkembangan ilmu pengetahuan dan teknologi.

Hormat saya,
Daniel Hermawan

Halaman ini sengaja dikosongkan.

DAFTAR ISI

LEMBAR PENGESAHAN i
APPROVAL SHEET iii
PERNYATAAN ORISINALITAS v
STATEMENT OF ORIGINALITY vii
PERNYATAAN KODE ETIK PENGGUNAAN AI GENERATIF ix
ABSTRAK xi
ABSTRACT xiii
KATA PENGANTAR xv
DAFTAR ISI xvii
DAFTAR GAMBAR xix
DAFTAR TABEL xxiii
DAFTAR KODE SUMBER xxv
BAB 1 PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4 Tujuan 2
1.5 Manfaat 2
BAB 2 TINJAUAN PUSTAKA 3
2.1 Hasil Penelitian Terdahulu 3
2.2 Dasar Teori 8
2.2.1 Scan Dokumen Akademik 8
2.2.2 *Deep learning* 8
2.2.3 Paddle OCR 9
2.2.4 DocTR (Document Text Recognition) 10
2.2.5 Tesseract OCR 11
2.2.6 Easy OCR 12
2.2.7 Generasi Data Tergradasi 13
2.2.8 Metrik Evaluasi Model OCR 14
BAB 3 METODOLOGI 17
3.1 Metode yang digunakan 17
3.1.1 Persiapan Dataset 17

3.1.2 Pengujian Bagian 1 (Model OCR *Standalone*) 20
3.1.3 Pengujian Bagian 2 (Model OCR *Hybrid*) 20
3.1.4 Pengujian Bagian 3 (dengan Pra-pemrosesan Data) 21
3.1.5 Evaluasi dan Analisis Kinerja Model OCR 22
3.2 Peralatan Pendukung 23
3.3 Implementasi Sistem 25
3.3.1 Persiapan Lingkungan, Pemuatan Dataset, dan Inisialisasi Model 25
3.3.2 Alur Kerja Evaluasi 26
3.3.3 Menjalankan Semua Pengujian 29
BAB 4 Hasil dan Pembahasan 31
4.1 Hasil Penelitian 31
4.1.1 Lingkungan Implementasi 31
4.1.2 Kinerja Model Baseline OCR pada Data Bersih 31
4.1.3 Kinerja Model OCR pada Data dengan *Noise* 33
4.1.4 Hasil Evaluasi OCR *Hybrid* (Detektor Paddle OCR) 34
4.1.5 Hasil Evaluasi OCR *Hybrid* (Detektor DocTR) 36
4.1.6 Hasil Evaluasi OCR *Hybrid* (Detektor Tesseract OCR) 38
4.1.7 Hasil Evaluasi OCR *Hybrid* (Detektor Easy OCR) 40
4.1.8 Hasil Uji Coba OCR *Hybrid* yang Ditingkatkan dengan Pra-pemrosesan (Detektor Paddle OCR) 42
4.1.9 Hasil Uji Coba OCR *Hybrid* yang Ditingkatkan dengan Pra-pemrosesan (Detektor Easy OCR) 44
4.1.10 Hasil Uji Coba OCR *Hybrid* yang Ditingkatkan dengan Pra-pemrosesan (Detektor Tesseract OCR) 46
4.1.11 Hasil Uji Coba OCR *Hybrid* yang Ditingkatkan dengan Pra-pemrosesan (Detektor DocTR) 48
4.2 Pembahasan 50
4.2.1 Analisis Kinerja dan Karakteristik Model *Standalone* 50
4.2.2 Analisis Kinerja OCR *Hybrid* 52
4.2.3 Analisis Kinerja Model OCR *Hybrid* dengan Pra-pemrosesan 72
BAB 5 Kesimpulan dan Saran 79
5.1 Kesimpulan 79
5.2 Saran 80
DAFTAR PUSTAKA 81
LAMPIRAN 83
BIODATA PENULIS 85

DAFTAR GAMBAR

Gambar 2.1 Arsitektur Neural Networks dalam Pengenalan Karakter 9
Gambar 2.2 Arsitektur Paddle OCR 10
Gambar 2.3 Arsitektur DocTR 11
Gambar 2.4 Arsitektur Tesseract OCR 12
Gambar 2.5 Arsitektur Easy OCR 13
Gambar 2.6 Contoh Hasil Augmentasi Gambar 14
Gambar 3.1 Diagram Alir Deteksi dan Pengenalan Karakter Dokumen *Scan* 17
Gambar 3.2 Perbandingan Hasil Gambar Setelah Augmentasi 18
Gambar 3.3 Jenis-Jenis *Noise* yang Diterapkan pada Generasi Data Tergradasi 19
Gambar 3.4 Jenis *Noise* pada Dataset Dokumen Tergradasi 25
Gambar 4.1 Kinerja Baseline Model OCR pada Skenario Original 32
Gambar 4.3 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario Original (Detektor Paddle OCR) 35
Gambar 4.4 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario *Noise* Tinggi (Detektor Paddle OCR) 35
Gambar 4.5 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario Original (Detektor DocTR) 37
Gambar 4.6 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario *Noise* Tinggi (Detektor DocTR) 37
Gambar 4.7 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario Original (Detektor Tesseract OCR) 39
Gambar 4.8 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario *Noise* Tinggi (Detektor Tesseract OCR) 39
Gambar 4.9 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario Original (Detektor Easy OCR) 41
Gambar 4.10 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario *Noise* Tinggi (Detektor Easy OCR) 41
Gambar 4.11 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario Original (Detektor Paddle OCR) 43
Gambar 4.12 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario *Noise* Tinggi (Detektor Paddle OCR) 43
Gambar 4.13 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario Original (Detektor Easy OCR) 45

Gambar 4.14 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario *Noise* Tinggi (Detektor Easy OCR) 45
Gambar 4.15 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario Original (Detektor Tesseract OCR) 47
Gambar 4.16 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario *Noise* Tinggi (Detektor Tesseract OCR) 47
Gambar 4.17 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario Original (Detektor DocTR) 49
Gambar 4.18 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario *Noise* Tinggi (Detektor DocTR) 49
Gambar 4.19 Perbandingan WER dan Waktu Pemrosesan Model *Standalone* pada Skenario Bersih 50
Gambar 4.20 Peningkatan *Word Error Rate* (WER) Model OCR *Standalone* dari Skenario Original ke Skenario *Noise* Tinggi 51
Gambar 4.21 Perbandingan Kinerja Model OCR *Standalone* vs. OCR *Hybrid* pada Skenario Original 52
Gambar 4.22 Heatmap Kinerja OCR *Hybrid* pada Skenario Original 54
Gambar 4.23 Perbandingan Kinerja Model OCR *Standalone* vs. OCR *Hybrid* pada Skenario *Noise* Tinggi 55
Gambar 4.24 Heatmap Kinerja OCR *Hybrid* pada Skenario *Noise* Tinggi 57
Gambar 4.25 Perbandingan Hasil Evaluasi OCR *Standalone* (Acuan), OCR *Hybrid* Terbaik, dan OCR *Hybrid* Terburuk pada Skenario Original 58
Gambar 4.26 Perbandingan Hasil Deteksi *Bounding box* Detektor Easy OCR (OCR *Hybrid* Terbaik) vs. Detektor DocTR (OCR *Hybrid* Terburuk) 61
Gambar 4.27 Hasil Deteksi oleh Detektor Easy OCR dan Pengenalan Teks oleh *Recognizer* Tesseract OCR 62
Gambar 4.28 Hasil Deteksi oleh Detektor DocTR dan Pengenalan Teks oleh *Recognizer* Tesseract OCR 64
Gambar 4.29 Perbandingan Hasil Evaluasi OCR *Standalone* (Acuan), OCR *Hybrid* Terbaik, dan OCR *Hybrid* Terburuk pada Skenario Original 65
Gambar 4.30 Perbandingan Hasil Deteksi *Bounding box* Detektor Easy OCR (OCR *Hybrid* Terbaik) vs. Detektor Paddle OCR (OCR *Hybrid* Terburuk) 68
Gambar 4.31 Hasil Deteksi oleh Detektor Easy OCR dan Pengenalan Teks oleh *Recognizer* Tesseract OCR pada Skenario *Noise* Tinggi 70
Gambar 4.32 Hasil Deteksi oleh Detektor Paddle OCR dan Pengenalan Teks oleh *Recognizer* DocTR pada Skenario *Noise* Tinggi 72
Gambar 4.33 Perbandingan Gambar Sebelum dan Sesudah Pra-pemrosesan 73
Gambar 4.34 Perbandingan Kinerja Model OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* dengan Pra-pemrosesan pada Skenario Original 74

Gambar 4.35 Heatmap Kinerja OCR *Hybrid* dengan Pra-pemrosesan pada Skenario Original 75
Gambar 4.36 Perbandingan Kinerja Model OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* dengan Pra-pemrosesan pada Skenario *Noise* Tinggi 76
Gambar 4.37 Heatmap Kinerja OCR *Hybrid* dengan Pra-pemrosesan pada Skenario *Noise* Tinggi 77

Halaman ini sengaja dikosongkan.

DAFTAR TABEL

Tabel 2.1 Perbandingan Studi Terkait *Optical Character Recognition* (OCR) 6
Tabel 3.1 Informasi Peralatan Pendukung..... 23
Tabel 3.2 Informasi Perangkat Lunak Pendukung 23
Tabel 4.1 Spesifikasi Lingkungan Eksperimen 31
Tabel 4.2 Kinerja Baseline Model OCR pada Skenario Original 32
Tabel 4.3 Kinerja Model OCR pada Skenario *Noise*..... 33
Tabel 4.4 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* (Detektor Paddle OCR) . 34
Tabel 4.5 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* (Detektor DocTR) 36
Tabel 4.6 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* (Detektor Tesseract OCR) 38
Tabel 4.7 Perbandingan Kinerja *Standalone* vs. OCR *Hybrid* (Detektor Easy OCR) 40
Tabel 4.8 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* Awal vs. OCR *Hybrid* Ditingkatkan (Detektor Paddle OCR) 42
Tabel 4.9 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* Awal vs. Ditingkatkan (Detektor Easy OCR) 44
Tabel 4.10 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* Awal vs. Ditingkatkan (Detektor Tesseract OCR) 46
Tabel 4.11 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* Awal vs. Ditingkatkan (Detektor DocTR)..... 48
Tabel 4.12 Perbandingan Hasil Pengenalan Teks OCR *Standalone* vs. OCR *Hybrid* Terbaik vs. OCR *Hybrid* Terburuk pada Skenario Original 59
Tabel 4.13 Perbandingan Hasil Pengenalan Teks OCR *Standalone* vs. OCR *Hybrid* Terbaik pada Skenario *Noise* Tinggi..... 66
Tabel 4.14 Perbandingan Hasil Pengenalan Teks OCR *Standalone* vs. OCR *Hybrid* Terburuk pada Skenario *Noise* Tinggi..... 67

Halaman ini sengaja dikosongkan.

DAFTAR KODE SUMBER

Kode Sumber 3.1 Insialisasi Proyek dan Persiapan Dokumen.....25
Kode Sumber 3.2 Inisialisasi dan Konfigurasi Model OCR 26
Kode Sumber 3.3 Fungsi Penambahan Pra-pemrosesan 27
Kode Sumber 3.4 Keseluruhan Alur Kerja Evaluasi.....28
Kode Sumber 3.5 Fungsi Utama Pengujian Model OCR.....30

Halaman ini sengaja dikosongkan.

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Digitalisasi arsip merupakan langkah penting bagi institusi akademik seperti Institut Teknologi Sepuluh Nopember (ITS) untuk meningkatkan aksesibilitas dan efisiensi pengelolaan informasi. Teknologi *Optical Character Recognition* (OCR) menjanjikan kemampuan untuk mengubah halaman dokumen menjadi data teks yang dapat diolah. Namun, hal ini seringkali terhalang oleh sebuah tantangan yang menghalangi untuk terwujudnya sistem tersebut, yaitu kualitas dari dokumen *scan*. Sebagian besar arsip akademik hadir dalam bentuk dokumen yang terdegradasi, mengandung berbagai jenis *noise* seperti butiran acak, pencahayaan tidak merata, hingga bayangan dari jilidan buku. Degradasi inilah yang menjadi penghalang utama karena menurunkan akurasi dan performa sistem. Oleh karena itu, diperlukan sebuah jalan keluar untuk permasalahan ini.

Untuk mengatasi masalah pada dokumen yang kualitasnya menurun, langkah paling dasar adalah menguji dulu kemampuan setiap model OCR yang ada (*standalone*). Tujuannya agar kita tahu apa kelebihan dan kekurangan dari masing-masing model tersebut, yang akan kita jadikan sebagai acuan. Setelah itu, ada dua cara utama yang bisa dicoba untuk mendapatkan hasil yang lebih baik. Cara pertama, fokus pada modelnya (*model-centric*). Artinya, kita mencoba membuat model OCR itu sendiri menjadi lebih pintar. Wujud nyata dari cara ini dalam penelitian adalah sistem OCR *Hybrid* (gabungan). Konsepnya adalah menggabungkan bagian pendeteksi teks dari satu model dengan bagian pengenal teks dari model lain, dengan harapan pasangan gabungan ini bisa bekerja lebih akurat. Cara kedua, fokus pada datanya (*data-centric*). Prinsipnya sederhana: hasil yang bagus datang dari gambar yang bagus. Jadi, bukannya mengubah modelnya, cara ini justru fokus memperbaiki kualitas gambar dokumen sebelum dibaca oleh OCR. Hal ini dilakukan dengan menerapkan serangkaian langkah perbaikan gambar yang disebut pra-pemrosesan, yang tujuannya adalah membersihkan gambar dari *noise* atau gangguan lainnya.

Terdapat penelitian sebelumnya yang mencoba menggunakan OCR dalam dunia akuntansi untuk otomasi dan meningkatkan cara kerja pemrosesan dokumen (Suhadja, A. et al. , 2020). Meskipun OCR terbukti bisa sangat mempercepat tugas-tugas repetitif, seperti memproses ribuan faktur setiap bulan, tantangan pun muncul. Masalah utamanya adalah kualitas dan sifat dokumen yang sangat bervariasi, baik itu dokumen yang berasal dari kertas yang di-scan maupun yang sudah berbentuk digital dari sistem akuntansi. Tantangan yang lebih spesifik dalam bidang akuntansi adalah variasi penulisan nama perusahaan dan daftar barang pada faktur, yang seringkali membuat sistem kesulitan untuk mencocokkannya dengan kamus kata yang sudah tersimpan. Meskipun demikian, jika tantangan-tantangan ini berhasil diatasi, potensi OCR untuk membuat pemrosesan dokumen di bidang akuntansi menjadi jauh lebih efisien sangatlah besar. Penelitian yang dibahas di sini bertujuan untuk mengisi celah tersebut dengan melakukan analisis perbandingan yang sangat lengkap dan mendalam.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dari penelitian ini adalah sebagai berikut :

1. Bagaimana kinerja dari berbagai model OCR *standalone* (seperti DocTR, Tesseract OCR, Paddle OCR) saat dihadapkan pada dokumen akademik dengan berbagai jenis dan tingkat *noise*?
2. Apakah pendekatan OCR *Hybrid*, yang menggabungkan beberapa model deteksi dan pengenalan dari model yang berbeda merupakan strategi yang efektif untuk meningkatkan akurasi OCR dibandingkan dengan model OCR *Standalone*?
3. Bagaimana efektivitas sebuah *pipeline* pra-pemrosesan gambar terhadap dampak *noise* dokumen *scan*, dan bagaimana kinerjanya jika dibandingkan dengan pendekatan OCR *Hybrid* maupun model OCR *Standalone*?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam Penelitian ini memiliki beberapa batasan, diantaranya sebagai berikut :

1. Bahasa pemrograman yang digunakan adalah Python
2. Framework dan library yang digunakan adalah Numpy, Pandas, Matplotlib, PyTorch, OpenCV, Tesseract OCR, Easy OCR, DocTR, dan Paddle OCR
3. Alat yang digunakan sebagai environment nya adalah Google Colab, Kaggle Notebook, dan Visual Studio Code

1.4 Tujuan

Tujuan dari pengerjaan Penelitian ini yaitu :

1. Menganalisis kinerja dari model OCR *Standalone* terhadap dataset dokumen *scan* yang bersih dan juga yang diberi *noise*.
2. Melakukan evaluasi pendekatan OCR *Hybrid* dengan menguji berbagai kombinasi model deteksi dan pengenalan (*recognizer*) untuk memahami dampaknya terhadap performa secara keseluruhan.
3. Menerapkan *pipeline* pra-pemrosesan gambar sebagai solusi alternatif untuk meningkatkan akurasi OCR.

1.5 Manfaat

Penelitian ini diharapkan dapat memberikan manfaat, beberapa diantaranya adalah :

1. Bagi Mahasiswa, memperoleh kemudahan dalam mendapatkan informasi tentang kurikulum, jadwal perkuliahan, dan kegiatan akademik lainnya.
2. Bagi Dosen, mencari referensi atau informasi akademik yang diperlukan untuk keperluan penelitian dan sebagainya.
3. Bagi Staff Administrasi, mempermudah proses pencarian informasi dalam dokumen scan, seperti data administrasi, regulasi, atau dokumen lain yang relevan.
4. Bagi Institusi Pendidikan, menjadi tonggak pengembangan teknologi dalam pengelolaan informasi akademik.
5. Bagi Perkembangan Teknologi, memberikan wawasan tentang potensi penerapan algoritma *deep learning* dalam pengenalan karakter dalam dokumen scan untuk berbagai keperluan, termasuk bidang Pendidikan.

BAB 2 TINJAUAN PUSTAKA

2.1 Hasil Penelitian Terdahulu

Penelitian dalam pengenalan karakter pada dokumen scan semakin memusatkan perhatian pada implementasi algoritma *deep learning*, dengan fokus utama pada pemanfaatan *Convolutional Neural Networks* (CNN). Penggunaan CNN sebagai pendekatan utama telah membawa kemajuan signifikan dalam meningkatkan akurasi dan efisiensi pengenalan karakter. Dalam konteks ini, penelitian juga menekankan pentingnya pengolahan dan normalisasi data citra sebagai langkah penting. Proses ini berperan penting dalam memastikan kinerja optimal dari algoritma *deep learning*, memungkinkan model untuk mengenali dan memproses karakter dengan akurasi tinggi. Implementasi CNN dalam pengenalan karakter pada dokumen scan membuka potensi baru untuk aplikasi teknologi OCR (*Optical Character Recognition*) yang lebih canggih. Penekanan pada pengolahan dan normalisasi data citra juga menunjukkan peran penting dalam menangani variasi kualitas dan karakteristik visual yang dapat ditemui dalam dokumen hasil pemindaian. Keseluruhan, pendekatan ini mencerminkan evolusi terkini dalam teknologi pengenalan karakter, memperkuat dasar untuk pengembangan sistem OCR yang lebih unggul dalam menangani berbagai jenis dokumen (Sridhar, S., et al., 2021).

Penggunaan pendekatan *deep learning* untuk pengenalan aksara Nusantara telah menjadi topik yang menarik dalam penelitian terkini. Penelitian terbaru yang dilakukan oleh Prasetiadi, A., et al. (2022) mengusulkan pendekatan yang inovatif dengan mengaplikasikan berbagai arsitektur seperti *Convolutional Neural Network* (CNN), ConvMixer, dan Visual Transformer. Hasil dari penelitian ini menunjukkan peningkatan signifikan dalam akurasi pengenalan aksara Nusantara dari dokumen scan. Pendekatan ConvMixer dan Visual Transformer terbukti memberikan hasil yang menjanjikan dalam mengatasi variasi kompleksitas karakter dan struktur aksara yang unik. Dalam konteks ini, penggunaan berbagai arsitektur *deep learning* menawarkan kemungkinan baru dalam pengenalan karakter, terutama untuk skrip yang kurang umum seperti aksara Nusantara.

Integrasi chatbot dalam pengelolaan informasi akademik telah menjadi fokus penelitian yang semakin meningkat. Beberapa penelitian, seperti yang dilakukan oleh Miller et al. (2023), menyoroti potensi chatbot dalam menyediakan akses yang lebih mudah terhadap informasi akademik bagi mahasiswa, dosen, dan staf universitas. Hasil penelitian ini menunjukkan bahwa penggunaan chatbot dapat memfasilitasi proses pencarian dan akses informasi akademik seperti jadwal kuliah, pengumuman, atau informasi mengenai fasilitas universitas secara lebih efisien. Integrasi chatbot dalam sistem informasi akademik juga meningkatkan interaksi pengguna dengan sistem, memberikan solusi yang lebih adaptif dan responsif terhadap kebutuhan informasi.

Tren terkini dalam teknologi pengenalan karakter dan integrasi *chatbot* menunjukkan perkembangan yang signifikan dalam beberapa tahun terakh Penelitian oleh Blecher, L., et al. (2023) merangkum inovasi terbaru dalam teknologi pengenalan karakter. Tinjauan ini menggambarkan adopsi teknologi Visual Transformer dalam meningkatkan akurasi pengenalan karakter. Selain itu, integrasi *chatbot* semakin berkembang dalam kemampuan bahasa alami dan respons yang lebih kontekstual. Hal ini menunjukkan arah baru yang menarik dalam pengembangan teknologi pengenalan karakter dan *chatbot* yang lebih adaptif dan efisien dalam menyediakan informasi akademik.

Terdapat sebuah penelitian yang berjudul Applications of integration of AI-*based Optical Character Recognition* (OCR) and *Generative* AI in *Document Understanding and Processing* yang membahas tentang prospek dari teknologi OCR dipadukan dengan teknologi lain seperti *Generative* AI (Abdelaziz, T. A. I., & Fazil, U., 2023). Penelitian ini menggambarkan kemajuan dalam penggunaan *Optical Character Recognition* (OCR) berbasis kecerdasan buatan (AI) dan kecerdasan buatan generatif dalam pemahaman dan pemrosesan dokumen. Penerapan teknologi ini terjadi pada beberapa tahap, dimulai dari pemindaian dan digitalisasi dokumen menggunakan OCR hingga kategorisasi dan organisasi dat. Model *Generative* AI kemudian memberikan kontribusi dalam menghasilkan ringkasan dokumen, mendeteksi dan memperbaiki kesalahan OCR, serta mengekstrak informasi spesifik. Penerapan teknologi ini juga mencakup bidang seperti penerjemahan bahasa, analisis sentimen, dan pembuatan dokumen. Secara keseluruhan, integrasi OCR dan *Generative* AI membawa perubahan transformasional dalam cara kita memahami dan memproses dokumen, memperbaiki efisiensi, akurasi, dan aksesibilitas informasi.

Pendekatan *hybrid* dalam ekstraksi informasi dari dokumen digital dan hasil pemindaian menjadi fokus utama dan dengan menggabungkan teknologi *Optical Character Recognition* (OCR) dengan model bahasa besar (*Large Language Models*) untuk menghasilkan keluaran terstruktur yang tidak hanya akurat tetapi juga kontekstual (Sinha, R., & Rekha, R. B., 2025). Sistem yang dikembangkan memanfaatkan alat OCR seperti Tesseract, DocTR, dan Google Vision API untuk dokumen hasil scan, serta perpustakaan seperti PyMuPDF dan pdfplumber untuk dokumen digital. Proses ekstraksi informasi dilakukan dalam beberapa tahap, dimulai dari pra-pemrosesan, ekstraksi teks mentah, hingga menggunakan LLM untuk membentuk pasangan *key-value* yang relevan. Evaluasi sistem menunjukkan peningkatan signifikan dalam aspek akurasi, ketahanan terhadap *layout* kompleks. Penelitian ini juga menyoroti pentingnya *confidence score* dalam hasil ekstraksi untuk mendukung aplikasi lanjutan seperti *indexing* otomatis dan penilaian berbasis teks. Pendekatan yang diusulkan menawarkan solusi yang lebih fleksibel dan adaptif dalam konteks pemrosesan dokumen yang bervariasi secara struktur dan format.

Penggunaan pendekatan gabungan CNN dan LSTM dalam pengenalan tulisan tangan menjadi inti dari penelitian ini. Model CNN dimanfaatkan untuk mengekstraksi fitur spasial dari gambar teks, sedangkan LSTM dipakai untuk memproses urutan karakter berdasarkan informasi spasial tersebut. Pendekatan ini dirancang untuk mengatasi tantangan utama dalam pengenalan tulisan tangan, seperti variasi bentuk huruf, kelengkungan garis tulisan, dan konektivitas antar karakter. Hasil evaluasi menunjukkan bahwa model ini mampu mencapai tingkat akurasi yang lebih tinggi dibanding metode tradisional. Penelitian ini juga menekankan pentingnya tahap pra-pemrosesan seperti normalisasi, binarisasi, dan penghalusan kontur tulisan dalam meningkatkan kinerja model. Dengan kombinasi kedua arsitektur ini, sistem pengenalan tulisan tangan menjadi lebih adaptif dan mampu menangani berbagai gaya tulisan, termasuk yang tidak standar. Pendekatan ini berkontribusi pada peningkatan kapabilitas OCR modern, khususnya untuk aplikasi di bidang digitalisasi arsip tulisan tangan (Kumar, J. P., & Dharshan, H. D., 2025).

Terdapat salah satu penelitian yang memfokuskan diri pada digitalisasi dokumen sejarah berbahasa daerah menggunakan pendekatan OCR berbasis AI (Vanitha et al., 2025). Sistem yang dikembangkan menggabungkan metode pra-pemrosesan canggih, model *deep learning* seperti CNN dan Transformer, serta modul terjemahan multibahasa untuk menghasilkan

representasi digital yang terstruktur dan dapat dicari. Tantangan seperti variasi tulisan tangan, teks yang memudar, dan keterbatasan dukungan bahasa oleh OCR konvensional diatasi melalui strategi pelatihan model pada dataset multibahasa dan penerapan teknik koreksi kesalahan berbasis NLP. Selain itu, sistem ini mendukung fitur tambahan seperti klasifikasi dokumen, pelabelan metadata, serta antarmuka pengguna berbasis web yang interaktif dan dilengkapi *chatbot* (Chat in PDF) untuk eksplorasi isi dokumen. Arsitektur ini memungkinkan skalabilitas melalui *cloud computing* serta dukungan *edge processing* untuk akses di daerah terbatas. Hasil evaluasi menunjukkan bahwa pendekatan terintegrasi ini mampu meningkatkan aksesibilitas dan pelestarian dokumen historis dalam bahasa daerah secara signifikan. Solusi ini dirancang tidak hanya untuk pelestarian arsip, tetapi juga untuk mendemokratisasi akses terhadap warisan budaya melalui teknologi yang inklusif dan berkelanjutan.

Demi mengatasi minimnya dukungan teknologi terhadap bahasa-bahasa minoritas yang sumber datanya terbatas, penelitian ini berfokus pada digitalisasi dokumen berbahasa lokal Sánchez et al., 2024). Dokumen tersebut adalah dokumen berbahasa lokal Peru yang ditulis dalam empat bahasa pribumi yaitu, Asháninka, Shipibo-Konibo, Yanesha, dan Yine. Tim peneliti mengembangkan dataset teranotasi yang berisi 454 halaman hasil scan dan menerapkan OCR menggunakan Google Vision serta Tesseract. Meskipun kedua sistem tidak secara eksplisit mendukung bahasa-bahasa tersebut, pengenalan berbasis huruf Latin tetap dapat dimanfaatkan. Untuk meningkatkan akurasi hasil ekstraksi, peneliti merancang modul koreksi berbasis alignment dan menerapkan pendekatan pasca-OCR menggunakan model pembelajaran berurutan seperti *SingleSource* dan *Ensemble spell checkers*. Hasil evaluasi menunjukkan bahwa kombinasi koreksi berbasis pembelajaran dan pengecekan linguistik dapat secara signifikan menurunkan nilai *Character Error Rate* (CER) dan *Word Error Rate* (WER). Pendekatan ini menjadi bukti bahwa teknologi OCR dapat diperluas secara efektif ke bahasa yang terpinggirkan, sekaligus menjadi kontribusi penting dalam upaya pelestarian warisan budaya dan linguistik.

Penelitian oleh Rakshit et al. (2023) mengusulkan *pipeline* end-to-end yang terdiri dari kombinasi teknologi OCR dan NLP untuk meningkatkan akurasi pengenalan karakter, khususnya dalam dokumen cetak dan tulisan tangan. Pendekatan ini mencakup dua modul utama: modul pertama memproses dokumen melalui segmentasi baris, klasifikasi tipe teks (cetak atau tulisan tangan), dan penerapan OCR menggunakan TrOCR atau PP-OCR, sedangkan modul kedua melakukan post-processing terhadap keluaran OCR menggunakan model NLP seperti ByT5, BART, dan Alpaca-LORA. Hasil evaluasi menunjukkan bahwa TrOCR lebih unggul dibanding PP-OCR pada sebagian besar dataset kecuali pada teks plat nomor. NLP pasca-OCR mampu secara signifikan menurunkan nilai CER dan WER, contohnya pada penggunaan Alpaca-LORA pada data sintetik berhasil menurunkan WER dari 0.455 menjadi 0.045 dan CER dari 0.124 menjadi 0.005. Penelitian ini juga menggarisbawahi peran augmentasi data dan strategi pembetulan ejaan dalam meningkatkan kinerja sistem. Dengan menggabungkan pengolahan visual dan semantik, *pipeline* ini menghadirkan pendekatan yang lebih adaptif terhadap variasi input dalam digitalisasi dokumen nyata.

Untuk memperoleh pemahaman yang lebih mendalam mengenai perkembangan teknologi *Optical Character Recognition* (OCR) dan pendekatan-pendekatan terkini yang digunakan dalam berbagai studi, penulis melakukan telaah terhadap sepuluh penelitian relevan. Studi-studi ini mencakup beragam pendekatan mulai dari penggunaan *Convolutional Neural Network* (CNN), Long Short-Term Memory (LSTM), hingga integrasi dengan model bahasa besar dan

generative AI. Perbandingan dilakukan berdasarkan aspek dataset yang digunakan, arsitektur model, performa utama, hingga keunggulan masing-masing pendekatan. Selain itu, analisis terhadap celah penelitian juga disajikan untuk menunjukkan kontribusi potensial dari penelitian ini dalam konteks yang lebih luas. Rangkuman dari hasil telaah pustaka tersebut disajikan pada Tabel 2.1 berikut ini.

Tabel 2.1 Perbandingan Studi Terkait *Optical Character Recognition* (OCR)

Judul	**Dataset**	**Model Terbaik**	**Performa**	**Analisis Gap**
Digitization of Document and Information Extraction using OCR	Dokumen hasil scan & digital (tidak disebutkan spesifik)	Gabungan OCR + LLM	Akurasi tinggi, efisiensi proses meningkat dengan ekstraksi semantik	Tidak fokus pada *noise* atau pengaruh kualitas visual buruk
Handwritten Text Recognition Using *Deep learning*: A CNN-LSTM Approach	Dataset tulisan tangan (tidak disebutkan spesifik)	CNN-LSTM	Akurasi tinggi dalam tulisan tangan; unggul dari metode tradisional	Tidak mempertimbangkan *noise* atau gangguan visual nyata
AI-Powered OCR for Digitizing Historical Document in Regional Languages	Dokumen berbahasa lokal & historis	*Pipeline* AI terintegrasi	Meningkatkan aksesibilitas, klasifikasi, dan pelabelan dokumen	Belum uji granular terhadap skenario *noise* atau OCR modular
Unlocking Knowledge with OCR-Driven Document Digitization for Peruvian Indigenous Languages	454 halaman dokumen 4 bahasa pribumi Peru	OCR + koreksi linguistik berbasis pembelajaran	Penurunan CER dan WER signifikan pasca-koreksi	Tidak mencakup deteksi teks modular atau evaluasi terhadap jenis *noise*

Deep learning Approaches for Nusantara Scripts OCR	Aksara Nusantara (Jawa, Bali, Bugis); >12.000 data hasil augmentasi	ConvMixer, Visual Transformer	Akurasinya 93–100% (karakter) & 96% (klasifikasi)	Tidak menangani *noise* atau variasi kualitas citra pemindaian
A Novel *Pipeline* for Improving OCR through Post-processing Using NLP	Born-digital, IAM, License Plate, Bing Quotes	TrOCR + Alpaca-LORA	CER: 0.124→0.005, WER: 0.455→0.045 (Alpaca)	Tidak memasukkan *noise* visual, fokus pasca-OCR saja
Character Recognition Using *Deep learning*	EMNIST (extended with other language characters)	CRNN (CNN + Bi-LSTM)	Akurasi tinggi dengan CNN + LSTM, CRNN sebagai pendekatan utama	Tidak mempertimbangkan *noise* atau variasi visual ekstrem seperti glare, blur, dan distorsi cetakan
Nougat: Neural Optical Understanding for Academic Documents	arXiv, PMC, IDL	Nougat base (350M)	Edit distance : 0.071, BLEU : 89.1, F1 Score : 93.1	Fokus pada PDF ilmiah born-digital, belum menangani dokumen rusak, buram, atau hasil scan berkualitas rendah
Applications of Integration of AI-based OCR and Generative AI in Document Understanding and Processing	Tidak disebutkan spesifik; umum untuk dokumen cetak dan tulisan tangan	OCR + LLM *pipeline*	Efisiensi dalam klasifikasi, ringkasan, koreksi kesalahan, dan ekstraksi data	Deskriptif dan konseptual, tidak menyajikan hasil kuantitatif atau pengujian performa model
OCR-free Document Understanding Transformer	Tobacco-3482, RVL-CDIP	Donut	Performa tinggi tanpa OCR tradisional, murni visual-semantic	Tidak relevan untuk pengenalan karakter dari scan *noise-heavy*; fokus pada struktur dan pemahaman dokumen bukan OCR murni

2.2 Dasar Teori

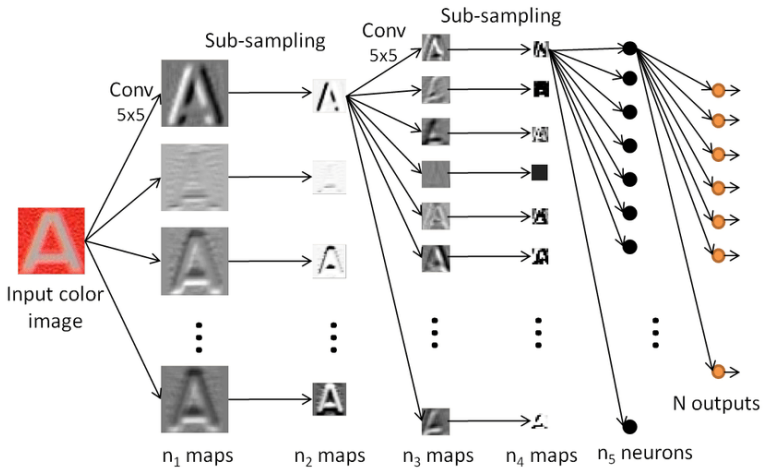
Dasar teori merupakan penjelasan yang mendetail mengenai konsep dan prinsip yang menjadi dasar dalam sistem pada model pengenalan *image to text* serta memberikan gambaran tentang keseluruhan implementasi dari sistem yang sedang dibangun. Dalam pengembangan sistem ini, penggunaan dasar teori sangat penting untuk memastikan bahwa sistem yang dibangun dapat berfungsi dengan baik dan sesuai dengan tujuan penggunaannya. Informasi tentang penggunaan dasar teori ini juga membantu proses perancangan dan pengembangan sistem dari model kecerdasan ini secara efektif dan efisien. Berikut adalah penjelasan singkat mengenai poin-poin dasar teori terkait pengembangan sistem deteksi dan pengenalan *image to text*.

2.2.1 Scan Dokumen Akademik

Dokumen scan akademik merupakan hasil dari pemindaian dokumen fisik ke dalam format digital, sering berisi materi akademik seperti jurnal ilmiah, buku, catatan kuliah, atau dokumen administratif institusi pendidikan. Proses pemindaian menghasilkan gambar teks dalam format PDF atau gambar raster (seperti TIFF atau JPEG) yang menjadi sumber data untuk pengenalan karakter. Isu utama terkait dokumen scan adalah variasi kualitas gambar, termasuk kejelasan teks, ukuran font, serta gangguan seperti noda atau bayangan yang dapat menghambat pengenalan karakter secara akurat. Dokumen scan akademik memiliki struktur dan konten kompleks yang beragam, dari tata letak hingga jenis teks dan format informasi, termasuk teks biasa hingga formula matematika atau notasi ilmiah. Beberapa dokumen memiliki halaman dengan kolom, tabel, atau gambar yang membutuhkan pendekatan khusus dalam pengenalan karakter. Ukuran yang beragam, termasuk dokumen berhalaman ratusan, menuntut manajemen data efisien dan pemrosesan cepat. Dalam pengembangan sistem pengenalan karakter, faktor-faktor seperti skala data, manajemen memori, dan optimisasi algoritma penting untuk menangani volume data yang besar dengan efisien. Pemahaman mendalam tentang struktur, karakteristik, dan kompleksitas dokumen scan akademik menjadi dasar kunci dalam mengembangkan sistem yang dapat mengenali karakter dengan akurat serta memahami konteks informasi akademik di dalamnya.

2.2.2 *Deep learning*

Deep learning merupakan cabang dari machine learning yang memusatkan perhatian pada pembelajaran dari representasi data yang mendalam. Dalam konteks pengenalan karakter dalam dokumen scan, pendekatan ini menonjol karena kemampuannya untuk secara otomatis mempelajari fitur-fitur kompleks dari data visual, seperti teks dalam gambar. Algoritma-algoritma dalam *deep learning*, seperti *Convolutional Neural Network*s (CNN), telah menjadi pilihan yang populer dalam mengatasi masalah pengenalan karakter dari gambar. CNN memanfaatkan struktur jaringan yang terdiri dari beberapa lapisan konvolusi dan lapisan terkait untuk secara efisien mengenali pola-pola lokal dalam gambar, memungkinkan mereka untuk diaplikasikan dengan baik dalam kasus pengenalan teks dari dokumen scan.



Gambar 2.1 Arsitektur Neural Networks dalam Pengenalan Karakter

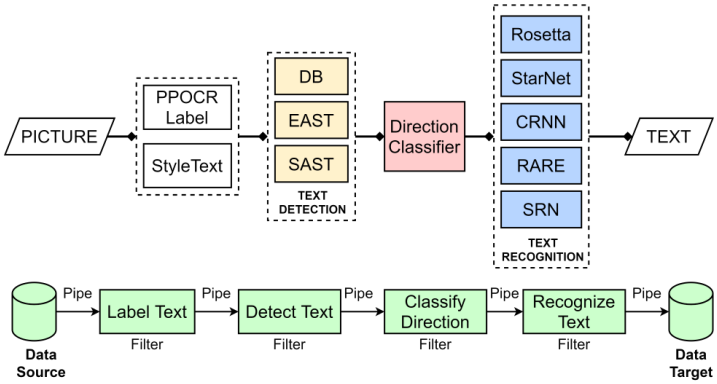
Keunggulan utama *deep learning*, khususnya dalam konteks pengenalan karakter, terletak pada kemampuannya dalam mempelajari representasi yang semakin tinggi dari data. Dalam hal ini, jaringan neural dalam *deep learning* mampu secara otomatis mengekstraksi fitur-fitur yang relevan dari gambar teks tanpa memerlukan proses ekstraksi manual yang rumit. Penggunaan metode *deep learning* juga mampu meningkatkan kinerja sistem dalam pengenalan karakter dengan peningkatan yang signifikan dalam akurasi dan efisiensi pengolahan data visual, yang merupakan kunci utama dalam pengenalan teks dari dokumen scan.

2.2.3 Paddle OCR

Paddle OCR adalah sebuah toolkit OCR *open-source* yang dikembangkan oleh Baidu dan berbasis pada framework *deep learning* PaddlePaddle. Keunggulan utama PaddleOCR terletak pada koleksi algoritmanya yang kaya dan modular, yang mencakup berbagai model untuk deteksi maupun pengenalan teks. Untuk deteksi teks, PaddleOCR sering mengimplementasikan algoritma canggih seperti DBNet. Cara kerjanya cukup unik, alih-alih langsung menggambar kotak pembatas di sekitar teks, DBNet pertama-tama membuat sebuah peta probabilitas untuk menebak piksel mana saja yang merupakan bagian dari sebuah teks. Setelah itu, algoritma ini dengan cerdas menentukan garis batas antara area teks dengan latar belakang. Dengan menggabungkan kedua informasi ini, DBNet dapat menghasilkan *bounding box* dengan presisi, bahkan mampu melingkupi teks yang melengkung atau tidak beraturan. Pendekatan ini membuat deteksi teks menjadi sangat akurat untuk berbagai tata letak dokumen.

Sementara untuk pengenalan teks, arsitektur yang umum digunakan adalah CRNN (*Convolutional Recurrent Neural Network*), yang mengombinasikan kekuatan CNN untuk ekstraksi fitur visual dan RNN (seperti LSTM) untuk pemodelan sekuensial karakter. Bagian pertama, CNN (*Convolutional*), bertindak sebagai "mata" dari model. Ia memindai potongan gambar teks dan mengekstraksi fitur-fitur visual penting seperti garis, lengkungan, dan sudut yang membentuk setiap karakter. Hasil dari CNN ini bukanlah teks, melainkan serangkaian fitur gambar yang kemudian diteruskan ke bagian kedua, yaitu RNN (R*ecurrent Neural Network*). Bagian RNN ini, biasanya berjenis LSTM, berfungsi sebagai otak yang memahami urutan. Ia menganalisis rangkaian fitur dari CNN secara sekuensial, sama seperti manusia membaca dari kiri ke kanan, sehingga ia dapat memahami konteks antar karakter. Kombinasi mata (CNN) yang melihat bentuk dan otak (RNN) yang memahami urutan inilah yang membuat arsitektur CRNN sangat efektif dalam mengenali berbagai jenis teks. Semua keunggulan ini,

ditambah dengan dukungan untuk lebih dari 80 bahasa dan model-model *pre-trained* yang sangat dioptimalkan, menjadikan PaddleOCR sebagai pilihan yang kuat dan seimbang.

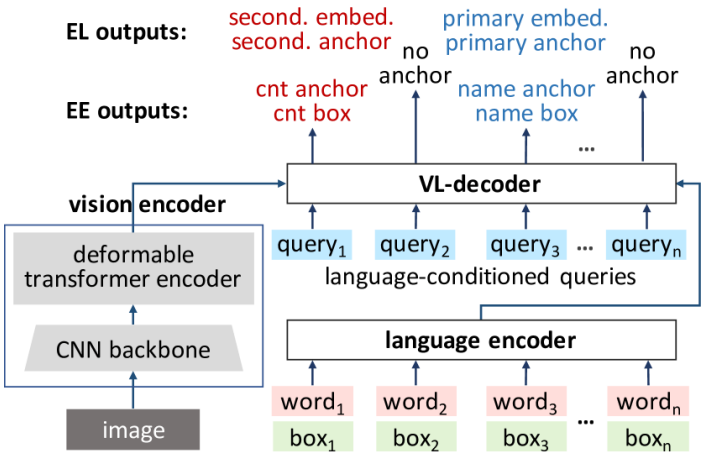


Gambar 2.2 Arsitektur Paddle OCR

2.2.4 DocTR (Document Text Recognition)

DocTR adalah model OCR *open-source* modern yang dikembangkan oleh Mindee, dirancang khusus untuk tugas pemahaman dokumen (*document understanding*). Model deteksi pada DocTR dirancang secara spesifik untuk mengatasi tantangan pada dokumen dengan tata letak yang kompleks, seperti formulir, faktur, atau tabel. Arsitekturnya dimulai dengan fondasi yang mirip dengan model lain, yaitu menggunakan CNN (*Convolutional Neural Network*) sebagai tulang punggung (*backbone*) untuk mengekstraksi fitur-fitur visual dasar dari gambar. Namun, keunggulan DocTR terletak pada penggunaan arsitektur berbasis Transformer. Transformer ini memungkinkan model untuk melihat gambar secara, tidak hanya per area kecil. Dengan kemampuan ini, model dapat memahami hubungan antara satu blok teks dengan blok teks lainnya, sehingga ia sangat mampu dalam membedakan mana judul, mana isi tabel, dan mana catatan kaki, bahkan jika posisinya tidak beraturan.

Pada tahap pengenalan teks, DocTR menunjukkan keunggulan yang paling menonjol dengan sepenuhnya meninggalkan pendekatan CRNN tradisional dan beralih ke arsitektur Transformer *end-to-end*. Prosesnya dibagi menjadi dua bagian utama. Pertama, Vision Transformer (ViT) Encoder, yang menerima potongan gambar teks. ViT memecah gambar tersebut menjadi kepingan-kepingan kecil dan menganalisis hubungan antar kepingan tersebut untuk memahami fitur dari seluruh kata atau baris secara bersamaan. Informasi ini kemudian dikirim ke bagian kedua, yaitu Transformer Decoder. *Decoder* inilah yang bertugas menghasilkan urutan karakter. Saat menghasilkan setiap huruf, *decoder* tidak hanya melihat fitur gambar dari *encoder*, tetapi juga melihat huruf-huruf yang telah ia tulis sebelumnya. Kemampuan ini membuatnya sangat pintar dalam memahami konteks kata dan menghindari kesalahan eja, sehingga menghasilkan yang sangat akurat.

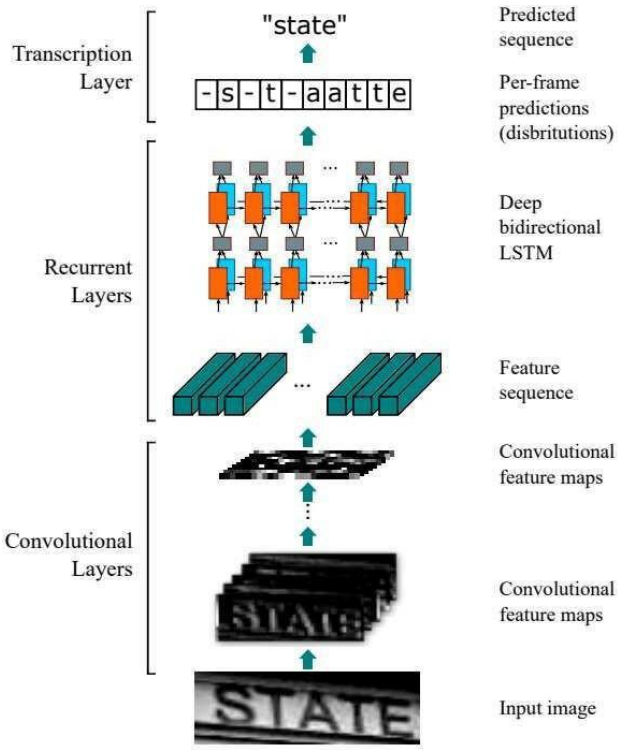


Gambar 2.3 Arsitektur DocTR

2.2.5 Tesseract OCR

Tesseract adalah salah satu mesin OCR *open-source* yang paling lama ada, paling dikenal, dan paling banyak digunakan, yang pengembangannya awalnya dipelopori oleh Hewlett-Packard dan kini dikelola oleh Google. Versi modern dari Tesseract (versi 4 dan setelahnya) menggunakan arsitektur berbasis CNN (*Convolutional Neural Network*), secara spesifik LSTM (*Long Short-Term Memory*), yang merupakan jenis dari RNN (*Recurrent Neural Network*). Model deteksi pada Tesseract menggunakan pendekatan yang disebut *Layout Analysis*. Sebelum mencoba mengenali karakter apa pun, Tesseract pertama-tama berusaha memahami struktur keseluruhan dari dokumen. Proses ini dimulai dengan mengidentifikasi blok-blok besar tulisan, seperti paragraf atau kolom. Setelah blok-blok tersebut ditemukan, algoritma akan bekerja untuk menemukan setiap baris teks di dalamnya. Terakhir, ia akan mencoba memisahkan setiap kata pada baris tersebut berdasarkan spasi antar karakter. Hasil akhir dari proses deteksi ini adalah serangkaian potongan gambar yang telah rapi menjadi baris-baris atau kata-kata. Pendekatan yang terstruktur dan bertahap inilah yang membuat proses deteksi Tesseract sangat cepat dan efisien.

Setelah halaman berhasil dipecah menjadi baris-baris teks, Tesseract menggunakan mesin pengenal (*recognizer*) berbasis LSTM. Mirip dengan arsitektur CRNN, prosesnya diawali dengan CNN yang bertugas mengekstraksi fitur-fitur visual dari gambar baris teks. Keunggulan utama Tesseract terletak pada penggunaan BiLSTM (*Bidirectional Long-Short Term Memory*). Artinya, model ini membaca fitur gambar tidak hanya dari kiri ke kanan, tetapi juga dari kanan ke kiri secara bersamaan. Dengan menganalisis urutan dari dua arah, model mendapatkan pemahaman konteks yang jauh lebih superior. Sebagai contoh, ia dapat lebih akurat menebak sebuah huruf di awal kata dengan melihat huruf-huruf yang mengikutinya. Kemampuan inilah yang membuat Tesseract sangat tangguh dan memiliki dukungan untuk lebih dari 100 bahasa.

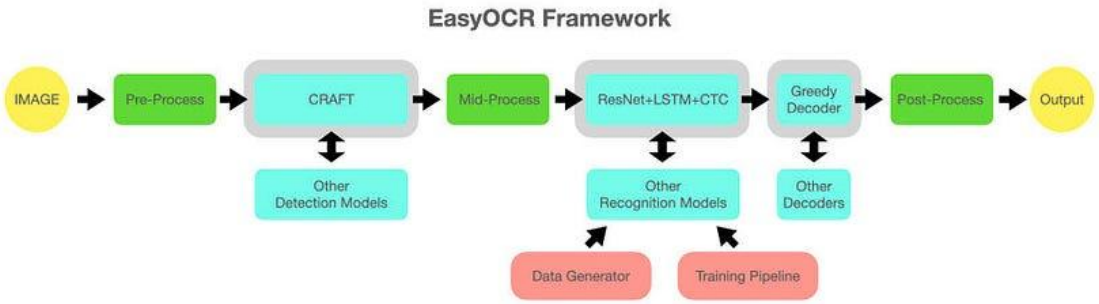


Gambar 2.4 Arsitektur Tesseract OCR

2.2.6 Easy OCR

Easy OCR, yang dikembangkan oleh Jaided AI, adalah *library* Python yang dirancang dengan satu filosofi utama yaitu, kemudahan penggunaan. Sesuai namanya, ia memungkinkan pengembang untuk mengimplementasikan fungsionalitas OCR dengan beberapa baris kode saja. Untuk mendeteksi lokasi teks, EasyOCR mengandalkan sebuah model yang sangat populer bernama CRAFT (*Character-Region Awareness for Text Detection*). Keunikan CRAFT terletak pada kemampuannya untuk mengenali wilayah per karakter individual, bukan langsung mencari satu kotak untuk satu kata. Ia menghasilkan dua jenis "peta panas" (*heatmap*). Peta pertama menandai area di mana kemungkinan besar terdapat sebuah karakter. Peta kedua, yang disebut *affinity score*, menandai "kedekatan" atau ikatan antar karakter tersebut. Dengan menggabungkan informasi ini, CRAFT dapat secara efektif mengelompokkan karakter-karakter yang berdekatan menjadi satu kata. Pendekatan yang fokus pada level karakter ini membuat CRAFT sangat andal dalam mendeteksi teks dengan berbagai ukuran dan orientasi, termasuk teks yang sedikit melengkung.

Sementara untuk pengenalan teks, Easy OCR mengandalkan arsitektur CRNN (*Convolutional Recurrent Neural Network*) yang populer dan efisien. Prosesnya diawali oleh ResNet, sebuah jenis CNN yang sangat kuat, yang bertugas mengekstraksi fitur-fitur visual dari gambar teks. Fitur-fitur ini kemudian diolah secara berurutan oleh LSTM, yang berfungsi untuk memahami konteks dan urutan dari karakter-karakter tersebut. Bagian terakhir dan yang menjadi kunci adalah CTC Loss (*Connectionist Temporal Classification*). CTC adalah algoritma pintar yang mengambil hasil dari LSTM dan secara otomatis menerjemahkannya menjadi teks akhir yang bersih. Ia mampu menangani masalah umum seperti karakter yang berulang atau spasi yang tidak sempurna, sehingga proses pelatihan dan pengenalan menjadi jauh lebih sederhana. Kombinasi dari komponen-komponen yang sudah matang inilah yang membuat Easy OCR dapat memberikan hasil yang baik.



Gambar 2.5 Arsitektur Easy OCR

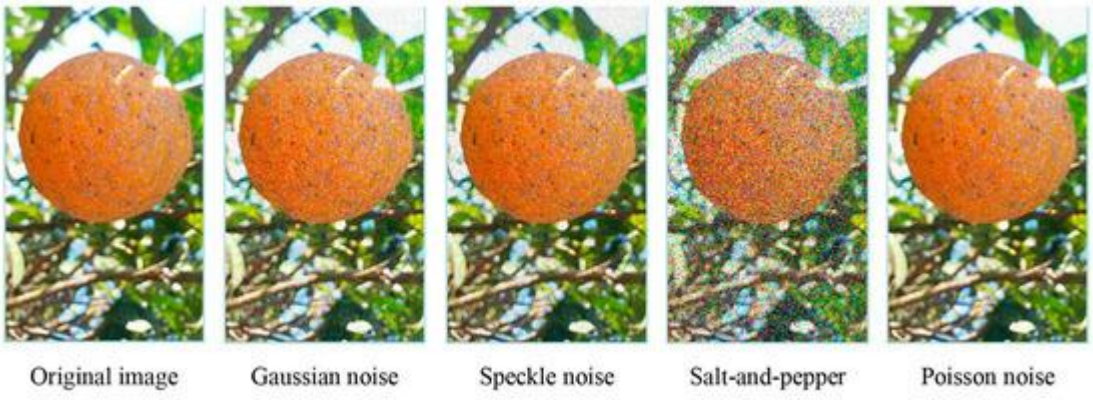
2.2.7 Generasi Data Tergradasi

Kinerja sebuah model OCR pada dokumen digital yang bersih seringkali tidak mencerminkan kemampuannya di dunia nyata. Dokumen hasil pindaian dari arsip fisik seringkali memiliki banyak kerusakan, seperti kertas yang menguning, noda, bayangan, atau hasil pindaian yang kurang fokus. Untuk dapat menguji seberapa tangguh setiap model OCR dalam menghadapi kondisi tersebut, penelitian ini tidak hanya menggunakan *dataset* asli yang bersih, tetapi juga membuat versi rusaknya secara buatan. Proses ini disebut degradasi buatan, di mana kami secara sengaja menambahkan berbagai jenis gangguan atau *noise* ke setiap gambar. Proses penambahan gangguan ini dilakukan secara terprogram menggunakan *library* OpenCV. Setiap jenis *noise* dipilih untuk meniru masalah spesifik yang sering terjadi saat proses pemindaian dokumen.

- ***Gaussian Noise*** : Gangguan ini menambahkan variasi acak pada kecerahan setiap piksel, menghasilkan efek yang mirip seperti semut atau bintik-bintik halus pada siaran televisi lama. Tujuannya adalah untuk mensimulasikan gangguan yang dihasilkan oleh sensor pemindai (*scanner*) berkualitas rendah atau saat pemindaian dilakukan dalam kondisi cahaya yang kurang baik.
- ***Salt & Pepper Noise*** : Jenis *noise* ini menambahkan titik-titik piksel berwarna hitam dan putih secara acak pada gambar. Ini adalah cara yang efektif untuk meniru kerusakan fisik kecil, seperti percikan tinta, debu yang menempel pada kaca pemindai, atau bintik-bintik pada kertas tua.
- ***Poisson Noise* (*Blur*)** : Dengan menerapkan filter *blur*, kami mensimulasikan kondisi di mana hasil pindaian tidak fokus. Hal ini bisa terjadi jika pemindai tidak dikalibrasi dengan baik atau jika kertas sedikit bergerak selama proses pemindaian, sehingga tepian karakter menjadi tidak tajam.
- **Pencahayaan Tidak Merata (*Lighting Gradient*)** : Gangguan ini mensimulasikan munculnya area gelap atau bayangan pada gambar, sebuah masalah yang sangat umum terjadi saat memindai buku tebal. Bayangan yang muncul di dekat jilidan buku dapat membuat kontras teks menjadi sangat rendah, sehingga menjadi tantangan besar bagi sistem OCR untuk membacanya.

Dengan menggabungkan dan mengatur tingkat kerusakan dari berbagai jenis gangguan di atas, penelitian ini berhasil menciptakan tiga skenario *noise* yang berbeda, yaitu rendah, sedang, dan tinggi. Pembuatan *dataset* pengujian ini memungkinkan kami untuk melakukan pengukuran terhadap penurunan kinerja setiap model seiring memburuknya kualitas gambar.

Dengan demikian, kami dapat memperoleh pemahaman yang lebih realistis dan akurat tentang kekuatan dan kinerja masing-masing model OCR saat dihadapkan pada tantangan dokumen di dunia nyata.



Gambar 2.6 Contoh Hasil Augmentasi Gambar

2.2.8 Metrik Evaluasi Model OCR

Dua metrik fundamental yang digunakan adalah CER (*Character Error Rate*) dan WER (*Word Error Rate*). Keduanya mengukur tingkat kesalahan, namun pada granularitas yang berbeda. CER mengukur tingkat kesalahan pada level karakter individual dengan membandingkan jumlah karakter yang salah terbaca dengan total karakter dalam teks referensi. Sebagai contoh, jika teks referensi adalah "metodologi" dan hasil OCR adalah "metodlogi" (karakter 'o' hilang), maka nilai CER-nya adalah 0,1.

$$CER = \frac{S + D + I}{N}$$

Di mana :

S = Jumlah Substitusi

D = Jumlah Penghapusan

I = Jumlah Penyisipan

N = Jumlah karakter dalam teks referensi (alias kebenaran dasar)

Penyebut N juga dapat dihitung dengan: N = S + D + C (di mana C = jumlah karakter yang benar)

Output persamaan ini menunjukkan persentase karakter dalam teks referensi yang diprediksi secara salah dalam output OCR. Semakin rendah nilai CER (dengan 0 sebagai skor sempurna), semakin baik kinerja model OCR. Di sisi lain, WER menilai kesalahan pada level kata, yang lebih mencerminkan keterbacaan dari sudut pandang manusia. Sebagai ilustrasi, jika teks referensi adalah "analisis data dengan metode" dan hasil OCR adalah "analisis dengan metode" (kata 'data' hilang), maka nilai WER-nya adalah 0,25. Kedua metrik ini memberikan gambaran komprehensif tentang ketepatan reproduksi teks, dengan nilai optimal untuk keduanya adalah 0, yang menandakan hasil OCR yang sempurna.

$$WER = \frac{S_w + D_w + I_w}{N_w}$$

Rumus untuk WER sama dengan rumus untuk CER, tetapi WER beroperasi pada tingkat kata. Rumus ini menunjukkan jumlah substitusi, penghapusan, atau penyisipan kata yang diperlukan untuk mengubah satu kalimat menjadi kalimat lain. WER secara umum berkorelasi baik dengan CER (asalkan tingkat kesalahannya tidak terlalu tinggi), meskipun nilai WER absolut diperkirakan lebih tinggi daripada nilai CER.

Selain metrik berbasis kesalahan, penelitian ini juga menggunakan BLEU (Bilingual Evaluation Understudy) Score untuk memberikan perspektif kualitas yang berbeda. Berasal dari dunia terjemahan mesin, BLEU tidak menghitung kesalahan, melainkan mengukur kesamaan dan "kelancaran" kalimat dengan membandingkan urutan kata-kata (n-gram) antara hasil OCR dan teks referensi. Skor yang tinggi (mendekati 1,0) menandakan bahwa hasil OCR tidak hanya mengandung kata-kata yang benar, tetapi juga tersusun dalam kalimat yang alami dan struktural mirip dengan aslinya. Terakhir, untuk mengevaluasi aspek praktis, digunakan metrik Waktu Pemrosesan, yang diukur dalam detik per halaman. Metrik ini sangat penting untuk analisis trade-off karena ia mengukur efisiensi komputasi dan kelayakan sebuah metode untuk diterapkan pada skala besar. Dengan menggunakan kombinasi keempat metrik ini, penelitian dapat melakukan evaluasi yang holistik terhadap setiap pendekatan yang diuji.

$$Bleu\ (N) = Brevity\ Penalty \cdot Geometric\ Average\ Precision\ Scores\ (N)$$

BLEU *score* dapat dihitung untuk nilai N yang berbeda-beda. Biasanya, dapat menggunakan N = 4.

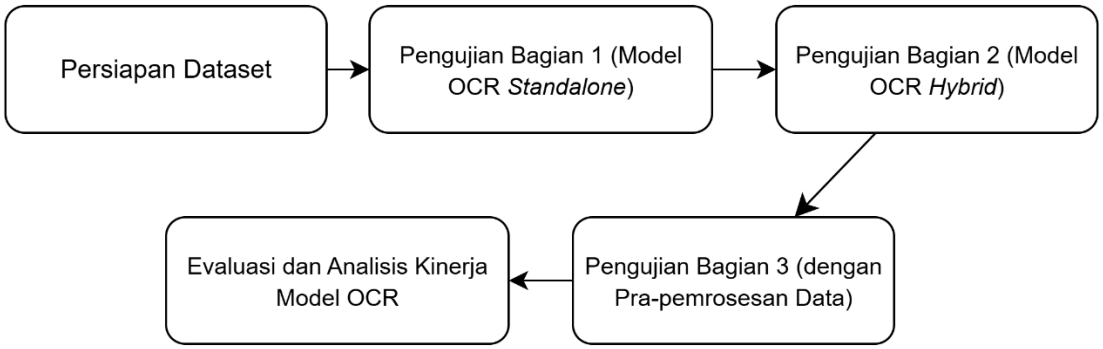
- BLEU-1 menggunakan skor Presisi unigram
- BLEU-2 menggunakan rata-rata geometri presisi unigram dan bigram
- BLEU-3 menggunakan rata-rata geometri presisi unigram, bigram, dan trigram
- dan sebagainya.

Halaman ini sengaja dikosongkan.

BAB 3 METODOLOGI

3.1 Metode yang digunakan

Pada tugas akhir ini, peralatan yang akan digunakan untuk melakukan pengembangan fitur *image to text* terdiri dari beberapa model OCR yaitu, Paddle OCR, DocTR, Tesseract OCR, dan Easy OCR yang memiliki perkembangan teknologi OCR terbaru dan telah dilatih dengan jutaan data yang membuatkan lebih unggul dan canggih dari jenis yang lainnya. Rancangan tahapan pelaksanaan dalam pengembangan model kecerdasan buatan ini adalah sebagai berikut.



Gambar 3.1 Diagram Alir Deteksi dan Pengenalan Karakter Dokumen *Scan*

Bab ini menguraikan secara rinci metodologi penelitian yang digunakan untuk menjawab rumusan masalah. Sebuah alur yang telah dirancang untuk memastikan setiap pendekatan diuji dengan baik, sehingga mendapatkan hasil yang akurat. Kerangka kerja penelitian ini diilustrasikan pada Gambar 3.1. Seperti yang ditunjukkan pada diagram alir, proses penelitian dimulai dengan tahap Persiapan Dataset, di mana seluruh data untuk pengujian disiapkan, termasuk data yang telah ditambahkan dengan *noise*. Tahap selanjutnya adalah inti dari eksperimen yang terbagi menjadi tiga bagian pengujian utama yaitu, Pengujian Bagian 1 untuk mengevaluasi kinerja dasar Model OCR *Standalone*, Pengujian Bagian 2 untuk menganalisis pendekatan Model OCR *Hybrid*, dan Pengujian Bagian 3 untuk menguji pengaruh dari penambahan Pra-pemrosesan Data. Seluruh hasil dari ketiga bagian pengujian tersebut kemudian diproses pada tahap akhir, yaitu Evaluasi dan Analisis Kinerja Model OCR, di mana performa setiap pendekatan diukur dan dibandingkan untuk menarik kesimpulan.

Penting untuk dicatat sebelum menguraikan prosedur dari setiap bagian pengujian, bahwa seluruh model OCR yang digunakan dalam penelitian ini merupakan perangkat lunak *open-source* yang dievaluasi dalam kondisi *pre-trained* atau siap pakai. Penelitian ini tidak melibatkan proses *training* model dari awal, *fine-tuning* terhadap dataset spesifik, maupun *hyperparameter tuning*. Fokus utama dari metodologi ini adalah murni pada evaluasi dan perbandingan kinerja dari model-model yang sudah ada, baik saat bekerja secara individual (*standalone*) maupun saat dikombinasikan dalam arsitektur OCR *Hybrid*.

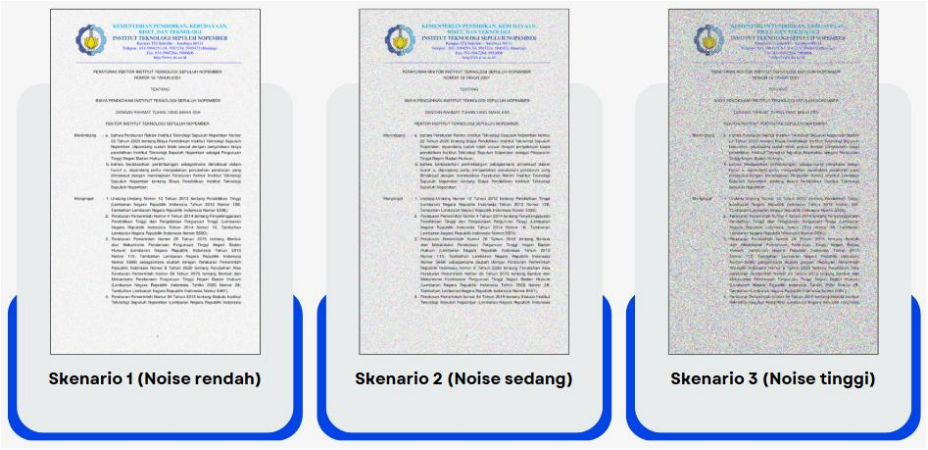
3.1.1 Persiapan Dataset

Langkah paling awal dan paling penting dalam penelitian ini adalah menyiapkan data yang akan dipakai untuk semua percobaan. Data ini harus sesuai dengan topik penelitian, yaitu dokumen akademik. Untuk itu, penulis mengumpulkan sebanyak 22 dokumen dalam format PDF sebagai sumber data utama. Sebagian besar, yaitu 20 dokumen, berasal dari lingkungan ITS seperti contoh Laporan Tugas Akhir dan materi perkuliahan, sementara 2 dokumen sisanya

adalah paper ilmiah untuk membuat jenis dokumennya lebih beragam. Syarat utama saat memilih semua dokumen ini adalah memastikan dokumen tersebut dibuat secara digital (born-digital), bukan hasil pindaian dari kertas. Hal ini sangat penting agar kita bisa mendapatkan teks aslinya, atau ground truth, yang dijamin benar seratus persen langsung dari filenya.

Setelah semua dokumen PDF terkumpul, dokumen-dokumen tersebut kemudian diolah untuk dijadikan data uji. Dari 22 dokumen itu, didapatkan total 153 halaman yang setiap halamannya diubah menjadi sebuah file gambar berformat PNG. Agar kualitasnya tetap bagus dan jelas, setiap gambar diatur resolusinya menjadi 300 DPI. Proses pengubahan ini dijalankan secara otomatis menggunakan program komputer (skrip Python) dengan bantuan library seperti pdf2image. Sambil mengubah halaman menjadi gambar, teks asli dari setiap halaman juga diambil dan disimpan ke dalam file teks terpisah. Kumpulan dari 153 pasang gambar dan teks asli inilah yang disebut sebagai Dataset Original (bersih). Dataset inilah yang akan menjadi patokan untuk mengukur kinerja awal model sebelum diberi gangguan apa pun.

Setelah Dataset *Original (bersih)* terbentuk, langkah metodologis selanjutnya adalah menerapkan augmentasi data untuk mensimulasikan berbagai kondisi dokumen dunia nyata. Tujuan utama dari tahap ini adalah untuk secara artifisial menciptakan degradasi dan *noise* yang umum ditemukan pada dokumen pindaian, sehingga ketahanan (*robustness*) dari setiap model OCR dapat diuji secara ketat, tidak hanya pada kondisi gambar yang sempurna. Proses augmentasi ini tidak dilakukan secara seragam, melainkan dirancang untuk menciptakan tiga tingkat kesulitan. Hal ini menghasilkan total empat kategori *dataset* yang akan digunakan secara konsisten dalam setiap pengujian, yaitu, Original (bersih)**,** Skenario 1 (*noise* rendah)**,** Skenario 2 (*noise* sedang)**,** dan Skenario 3 (*noise* tinggi)**.** Untuk memberikan gambaran visual yang jelas mengenai perbedaan antara setiap skenario, contoh dari setiap kategori data disajikan pada Gambar 3.2.

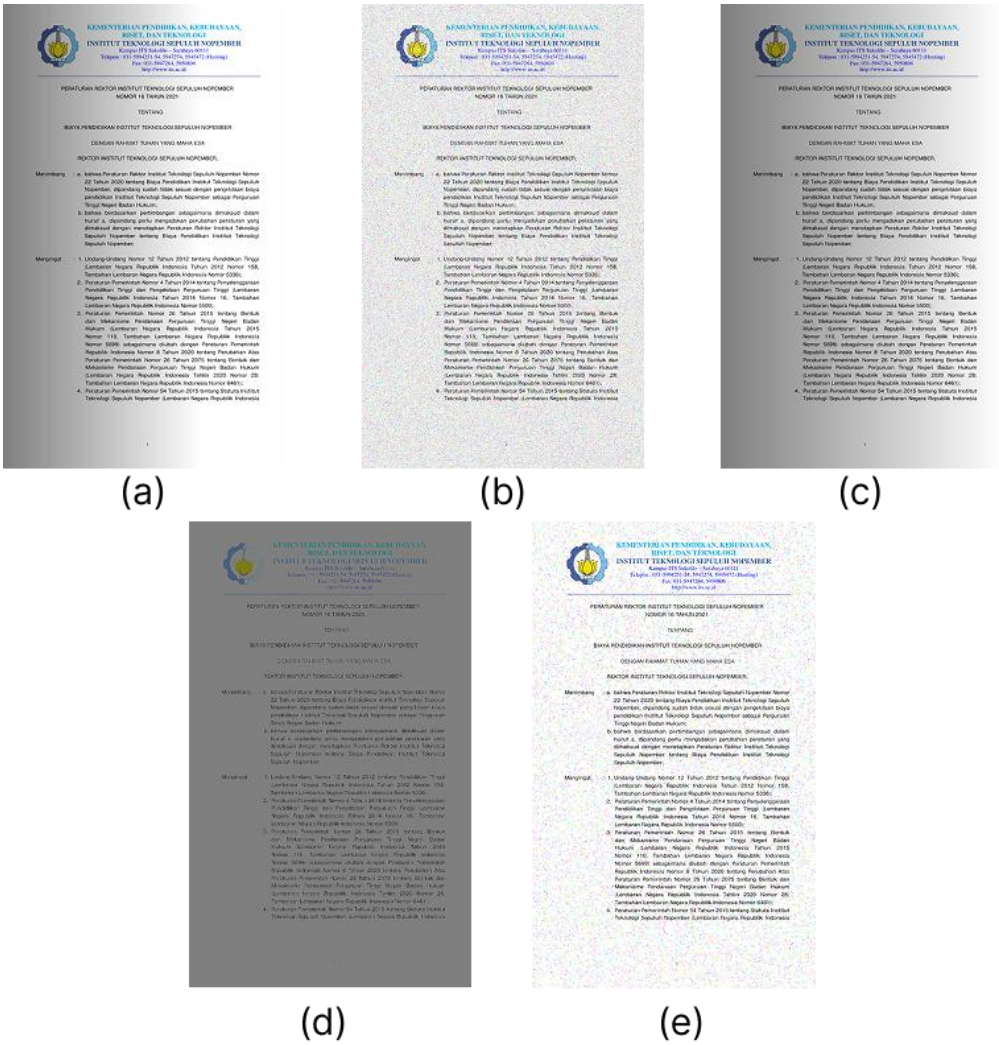


Gambar 3.2 Perbandingan Hasil Gambar Setelah Augmentasi

Augmentasi pada gambar di atas diimplementasikan menggunakan *library* OpenCV. Untuk meniru berbagai kondisi ini, kami sengaja membuat versi tergradasi dari gambar-gambar bersih yang sudah ada. Kami menggunakan program komputer untuk menambahkan empat jenis gangguan atau *noise* yang berbeda, di mana setiap gangguan meniru masalah yang sering muncul di dunia nyata.

- *Salt & Pepper Noise* : Ini menambahkan titik-titik hitam dan putih acak, mirip seperti debu di kaca *scanner* atau noda-noda kecil di kertas tua.
- *Gaussian Noise* : Ini menambahkan bintik-bintik halus di seluruh gambar, seperti gangguan berupa debu yang bisa muncul dari alat *scanner* yang kualitasnya kurang bagus.
- *Poisson Noise* : Gangguan ini biasanya muncul saat *scanner* kesulitan menangkap gambar di tempat gelap. Hasilnya adalah gambar yang berbintik, terutama di area yang seharusnya terang.
- Pencahayaan Tidak Merata : Ini meniru bayangan pekat yang sering muncul di dekat jilidan buku tebal, yang membuat tulisan di area itu jadi sangat sulit dibaca.

Keempat jenis gangguan ini kemudian digabungkan dengan tingkat kerusakan yang berbeda-beda untuk menciptakan tiga *dataset* pengujian akhir yaitu, Skenario *Noise* Rendah, Skenario *Noise* Sedang, dan Skenario *Noise* Tinggi. Dengan cara ini, kami bisa mengukur dengan adil seberapa jauh kemampuan setiap model OCR menurun saat dihadapkan pada gambar yang semakin rusak, sehingga kita bisa tahu seberapa tangguh model tersebut sebenarnya.



Gambar 3.3 Jenis-Jenis *Noise* yang Diterapkan pada Generasi Data Tergradasi

Gambar 3.3 di atas menampilkan contoh visual dari lima jenis degradasi buatan yang diterapkan dalam penelitian ini guna menguji ketahanan model OCR.

- (a) dan (c) Pencahayaan Tidak Merata : Simulasi bayangan pekat yang sering muncul di dekat jilidan buku tebal, yang menyebabkan penurunan kontras teks.
- (b) Gaussian *Noise* : Simulasi gangguan yang dihasilkan oleh sensor pemindai (*scanner*) berkualitas rendah, terlihat seperti bintik-bintik halus yang merata.
- (d) Poisson *Noise* : Simulasi hasil pemindaian dalam kondisi kekurangan cahaya, yang menghasilkan bintik-bintik (*graininess*) pada gambar.
- (e) Salt & Pepper *Noise* : Simulasi kerusakan fisik minor seperti debu pada kaca pemindai atau noda pada kertas, terlihat seperti titik-titik hitam dan putih yang tajam.

3.1.2 Pengujian Bagian 1 (Model OCR *Standalone*)

Tahap pengujian pertama ini merupakan landasan dari keseluruhan analisis dalam penelitian ini. Fokus utamanya adalah untuk mengevaluasi dan memahami secara mendalam kinerja dasar dari keempat model OCR yang dipilih (DocTR, Tesseract OCR, Paddle OCR, dan Easy OCR) saat bekerja secara individual atau *standalone*. Tujuan dari tahap ini adalah untuk membangun sebuah acuan atau *baseline* untuk menjadi dasar untuk melakukan peningkatan pada skenario konfigurasi lain. Pada pengujian bagian ini, ada tiga hal yang akan dinilai yaitu, yang pertama adalah akurasi, kemampuan model untuk mengenali teks secara tepat dalam kondisi ideal. Kedua, efisiensi, diukur dari kecepatan pemrosesan per halaman. Ketiga, ketahanan (*robustness*), yaitu kemampuan model untuk mempertahankan kinerjanya saat kualitas gambar menurun akibat *noise*. Pemahaman yang kuat terhadap ketiga aspek ini berfungsi sebagai tolok ukur untuk menilai pendekatan yang lainnya yaitu, OCR *Hybrid* dan dan Pra-pemrosesan.

Untuk memastikan hasil yang dapat dibandingkan, setiap model diuji kemampuannya terhadap data yang telah disiapkan, mulai dari Dataset Original (Bersih) yang merepresentasikan kondisi ideal, hingga tiga dataset *noise* (Rendah, Sedang, dan Tinggi) yang merepresentasikan berbagai tingkat degradasi dokumen. Luaran dari tahap pengujian *standalone* ini adalah kumpulan data mentah, yaitu serangkaian file teks yang berisi hasil prediksi dari setiap model untuk setiap gambar di bawah setiap kondisi. Kumpulan data ini merupakan bahan utama yang akan diolah pada tahap evaluasi. Pada tahap selanjutnya yang dijelaskan di sub-bab 3.1.5, seluruh *output* teks ini akan diukur performanya dengan cara membandingkannya dengan teks *ground truth* menggunakan berbagai metrik evaluasi seperti *Word Error Rate* (WER) dan *Character Error Rate* (CER). Analisis mendalam terhadap hasil pengukuran inilah yang nantinya akan disajikan pada Bab 4 sebagai bahasan terkait kinerja dasar dari masing-masing model OCR.

3.1.3 Pengujian Bagian 2 (Model OCR *Hybrid*)

Tahap pengujian kedua dalam penelitian ini berfokus pada pendekatan Model OCR *Hybrid*. Berbeda dengan pengujian *standalone* yang mengevaluasi setiap model sebagai satu kesatuan utuh, pendekatan ini didasarkan pada hipotesis bahwa performa OCR dapat dioptimalkan dengan menggabungkan komponen terbaik dari model-model yang berbeda. Dugaan awalnya adalah bahwa sebuah model mungkin memiliki komponen detektor yang bekerja dengan baik dalam menemukan teks pada gambar yang terdegradasi, sementara model lain memiliki komponen *recognizer* yang lebih unggul dalam mengenali karakter. Tujuan dari tahap ini adalah untuk menyelidiki secara sistematis terkait kecocokan antara detektor dari satu model

dengan *recognizer* dari model lain yang mampu menghasilkan kinerja lebih tinggi daripada model aslinya.

Alur kerja pada pengujian OCR *Hybrid* ini secara mendasar membagi model OCR menjadi dua komponen utama dalam OCR yaitu, model deteksi dan model pengenalan. Untuk alur pengujiannya, yang pertama, sebuah gambar dari dataset dioperkan ke komponen detektor dari model yang telah dipilih (misalnya, Detektor dari DocTR). Detektor kemudian menganalisis seluruh gambar untuk memetakkan semua area yang mengandung teks dan menghasilkan serangkaian koordinat *bounding box* sebagai *output*-nya. Selanjutnya, koordinat-koordinat ini digunakan untuk memotong gambar asli menjadi beberapa gambar yang lebih kecil, di mana setiap gambar kecil hanya berisi pecahan teks yang telah diidentifikasi oleh detektor. Kumpulan gambar-gambar kecil inilah yang kemudian diteruskan sebagai masukan ke komponen *recognizer* dari model kedua yang berbeda (misalnya, *recognizer* dari Tesseract) untuk diterjemahkan menjadi teks akh

Untuk memastikan investigasi yang komprehensif, pengujian dilakukan dengan metode kombinasi silang (*cross-combination*). Detektor dari setiap model diuji secara berpasangan dengan *recognizer* dari semua model lainnya, sehingga menciptakan berbagai konfigurasi *hybrid* yang unik. Sama seperti pada pengujian *standalone*, setiap konfigurasi *hybrid* ini diuji terhadap keseluruhan kualitas data, yaitu keempat dataset (Bersih, *Noise* Rendah, Sedang, dan Tinggi). Hal ini dilakukan untuk menilai tidak hanya akurasi terbaik dari setiap kombinasi, tetapi juga untuk memahami ketahanannya terhadap degradasi. Luaran dari tahap ini adalah kumpulan data mentah berupa teks *output* dari setiap konfigurasi OCR *hybrid*, yang selanjutnya akan diukur dan dianalisis pada tahap evaluasi akh

3.1.4 Pengujian Bagian 3 (dengan Pra-pemrosesan Data)

Tahap pengujian ketiga dan terakhir ini dirancang sebagai investigasi lebih lanjut terhadap pendekatan OCR *hybrid*. Berdasarkan hasil dari pengujian sebelumnya, ditemukan bahwa banyak kombinasi detektor dan *recognizer* yang tidak cocok dan justru menghasilkan kinerja yang lebih buruk daripada model *standalone*. Tahap ini menguji hipotesis bahwa kinerja dari beberapa kombinasi *hybrid* yang tidak cocok tersebut dapat diperbaiki melalui *pipeline* pra-pemrosesan data. Ketidakcocokan tersebut mungkin bukan disebabkan oleh ketidakmampuan *recognizer* untuk membaca teks, melainkan karena kualitas gambar hasil potongan (*bounding box*) dari detektor yang tidak standar, misalnya ukurannya bervariasi, terlalu sempit, atau memiliki informasi visual yang tidak relevan seperti warna.

Alur kerja pada pengujian ini sebagian besar identik dengan pengujian OCR *Hybrid* pada subbab sebelumnya, namun dengan penambahan satu langkah penting. Prosesnya dimulai dengan gambar dari dataset yang diumpankan ke komponen detektor, yang kemudian menghasilkan koordinat *bounding box*. Setelah gambar asli dipotong berdasarkan koordinat tersebut, gambar-gambar kecil yang dihasilkan tidak langsung diserahkan ke *recognizer*. Sebaliknya, setiap gambar kecil tersebut pertama-tama dilewatkan melalui sebuah *pipeline* pra-pemrosesan untuk dibersihkan dan distandardisasi. Baru setelah melalui *pipeline* inilah gambar yang telah diproses diserahkan ke komponen *Recognizer* untuk diterjemahkan menjadi teks akh

Penambahan pra-pemrosesan yang diterapkan dalam penelitian ini terdiri dari tiga langkah utama yang berurutan. Pertama adalah Penyeragaman Ukuran (*resizing*), di mana tinggi dari setiap gambar potongan disesuaikan menjadi 64 piksel dengan tetap menjaga rasio aspeknya, bertujuan agar *recognizer* menerima masukan dengan dimensi yang konsisten. Langkah kedua

adalah penambahan *padding*, di mana sebuah bingkai atau batas kecil ditambahkan di sekeliling gambar untuk memberikan ruang tambahan dan memastikan tidak ada bagian karakter di tepian yang terpotong. Langkah terakhir adalah konversi ke *grayscale*, yaitu mengubah gambar dari berwarna menjadi hitam-putih untuk membantu model fokus pada bentuk serta kontras karakter.

Seluruh prosedur ini diterapkan pada konfigurasi-konfigurasi *hybrid* dan diuji kembali terhadap keempat dataset (Bersih, *Noise* Rendah, Sedang, dan Tinggi). *Output* dari tahap ini adalah kumpulan teks hasil OCR dari konfigurasi *hybrid* yang telah ditingkatkan dengan pra-pemrosesan. Hasil ini kemudian akan dibandingkan secara langsung dengan kinerja model *standalone* dan model *hybrid* standar untuk menentukan secara pasti dalam kondisi apa saja *pipeline* pra-pemrosesan ini merupakan strategi optimasi yang efektif.

3.1.5 Evaluasi dan Analisis Kinerja Model OCR

Tahap terakhir dalam metodologi penelitian ini adalah evaluasi dan analisis kinerja. Tahap ini bertujuan untuk mengukur secara kuantitatif hasil dari ketiga bagian pengujian yang telah dijalankan (*Standalone*, *Hybrid*, dan *Hybrid* dengan Pra-pemrosesan). Untuk memastikan perbandingan yang terukur, serangkaian metrik evaluasi yang sama diterapkan pada seluruh teks *output* yang telah dikumpulkan. Pengukuran ini tidak hanya berfokus pada akurasi, tetapi juga pada aspek kualitas dan efisiensi.

Untuk mengukur tingkat akurasi, dua metrik utama digunakan yaitu, *Word Error Rate* (WER) dan *Character Error Rate* (CER). WER adalah metrik yang menghitung tingkat kesalahan pada level kata, dengan mengkalkulasi jumlah kata yang perlu diganti (*substitution*), dihapus (*deletion*), atau disisipkan (*insertion*) agar teks keluaran sesuai dengan teks ground truth. Metrik ini sangat relevan untuk menilai keterbacaan hasil dari sudut pandang manusia, di mana nilai yang semakin rendah menunjukkan akurasi yang semakin tinggi. Sementara itu, CER mengukur kesalahan pada level karakter individual. Metrik ini memberikan gambaran yang lebih granular mengenai presisi model dan sangat berguna untuk mengidentifikasi jenis-jenis kesalahan spesifik, seperti kebingungan antara karakter yang mirip secara visual. Sama seperti WER, nilai CER yang semakin rendah menandakan performa yang lebih baik.

Selain metrik berbasis kesalahan, dua metrik tambahan digunakan untuk menilai kualitas dan efisiensi. BLEU *score* (*Bilingual Evaluation Understudy*) adalah metrik yang diadopsi dari bidang terjemahan mesin untuk mengukur kesamaan dan kelancaran struktur kalimat. Metrik ini membandingkan urutan kata (*n-grams*) antara teks keluaran dengan teks *ground truth*, di mana skor yang tinggi (mendekati 1.0) menandakan hasil yang tidak hanya benar secara kata, tetapi juga alami secara susunan kalimat. Terakhir, waktu pemrosesan diukur untuk mengevaluasi efisiensi komputasi dari setiap pendekatan. Metrik ini mencatat rata-rata waktu dalam detik yang dibutuhkan untuk memproses satu halaman dokumen, sebuah faktor penting untuk menentukan kelayakan penerapan sebuah metode dalam skala besar.

Setelah semua data dari metrik-metrik tersebut terkumpul, proses dilanjutkan dengan analisis komparatif. Pada tahap ini, hasil dari pendekatan *hybrid* dan *hybrid* dengan Pra-pemrosesan akan dibandingkan secara langsung dengan kinerja dasar dari model-model *Standalone*. Analisis ini bertujuan untuk mengidentifikasi pola, menemukan kombinasi yang paling efektif, serta memahami kondisi-kondisi di mana sebuah pendekatan optimasi berhasil atau gagal. Hasil dari evaluasi dan analisis inilah yang akan menjadi dasar dari seluruh pembahasan dan penarikan kesimpulan yang disajikan pada Bab 4 dan Bab 5.

3.2 Peralatan Pendukung

Dalam pengerjaan Tugas Akhir ini, perangkat keras yang digunakan untuk mengerjakan adalah laptop dengan spesifikasi sebagai berikut.

Tabel 3.1 Informasi Peralatan Pendukung

Brand	Lenovo Legion 5i
Processor	Intel Core i7-13650HX
Operating System	Windows 11
System Specification	Windows 11 Home Single Language Version 23H2
System Type	64-bit operating system, x64-based processor
GPU	NVIDIA RTX 4060 8 GB
RAM	32,00 GB (31,7 GB usable)
Storage	474 GB SSD

Selain perangkat keras, terdapat beberapa perangkat lunak pendukung seperti berikut.

Tabel 3.2 Informasi Perangkat Lunak Pendukung

Nama Perangkat Lunak	Keterangan
Git / Github	Git adalah sistem kontrol yang digunakan untuk mengelola perubahan dalam kode, sedangkan GitHub adalah platform hosting yang memanfaatkan Git untuk menyediakan repositori terpusat dan berbagai alat kolaborasi untuk pengembang. Git dan GitHub bersama-sama membantu melacak perubahan, dan mengelola kode sumber proyek-proyek perangkat lunak.
Hugging Face	Hugging Face adalah platform untuk berbagi dan mengelola model kecerdasan yang beragam sama halnya dengan cara kerja Github. Dalam konteks pengerjaan Tugas Akhir yang berfokus pada pengenalan karakter dalam dokumen scan menggunakan algoritma *deep learning* dan integrasi chatbot untuk generasi intent, Hugging Face dapat digunakan sebagai sumber daya yang berharga. Salah satu kegunaannya adalah sebagai sumber beragam model bahasa yang sudah dilatih sebelumnya. Platform ini menyediakan akses ke berbagai model NLP, termasuk model Transformer seperti BERT,

	GPT (Generative Pre-trained Transformer), dan lainnya.
Google Colab	Google Colab adalah platform cloud yang menyediakan lingkungan pengembangan berbasis web, dirancang untuk melakukan pemrograman dengan menggunakan Python. Dalam konteks pengerjaan Tugas Akhir, Google Colab memberikan akses kepada pengguna untuk menjalankan kode, membuat dan membagikan notebook, serta melakukan komputasi di cloud menggunakan sumber daya Google, termasuk penggunaan GPU dan TPU.
ChatGPT	ChatGPT adalah sebuah perangkat lunak yang menggunakan kecerdasan buatan untuk memfasilitasi komunikasi dan pengolahan bahasa alami. Pertama, sebagai alat untuk mendukung proses riset dan studi literatur dengan memberikan informasi atau penjelasan tentang topik yang sedang diteliti. Kedua, sebagai alat untuk membantu menyusun dan merancang metodologi, terutama dalam tahap wawancara kebutuhan atau analisis data, dengan memberikan saran atau contoh pendekatan yang relevan. Ketiga, ChatGPT juga bisa digunakan sebagai alat untuk mendiskusikan ide-ide dan mendapatkan masukan, memungkinkan peneliti untuk menjelaskan dan merumuskan pertanyaan atau ide secara verbal untuk mendapat respons atau pandangan tambahan.
Visual Studio Code	Visual Studio Code (VS Code) adalah sebuah lingkungan pengembangan terintegrasi yang ringan, modular, dan sangat fleksibel yang sering digunakan dalam pengerjaan proyek seperti Tugas Akh Dikenal karena antarmuka yang intuitif, dukungan luas terhadap bahasa pemrograman, serta beragam ekstensi yang memperluas fungsionalitasnya, VS Code menjadi pilihan populer bagi pengembang perangkat lunak, termasuk untuk mengembangkan solusi algoritma dalam Tugas Akh VS Code memungkinkan pengguna untuk melakukan pengembangan pada lingkungan lokal komputer.

3.3 Implementasi Sistem

Setelah semua rencana penelitian disusun, langkah selanjutnya adalah penjelasan teknis tentang bagaimana semua rencana itu dijalankan menggunakan program komputer. Bagian ini akan menjelaskan langkah-langkah kerja teknisnya secara berurutan, dari awal hingga akhir, agar mudah diikuti dan bisa diuji ulang oleh orang lain. Untuk mempermudah penjelasan tentang logika program, beberapa contoh kode sederhana (*pseudocode*) akan disertakan. Rencana implementasi ini dimulai dari persiapan lingkungan, dilanjutkan dengan cara menjalankan setiap skenario pengujian, hingga proses evaluasi akhirnya.

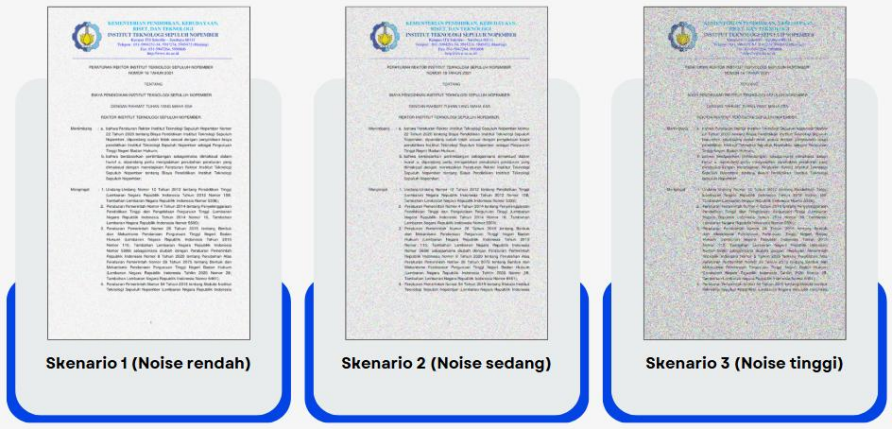
3.3.1 Persiapan Lingkungan, Pemuatan Dataset, dan Inisialisasi Model

Langkah paling awal dalam implementasi program adalah menyiapkan lingkungan kerja dan semua data yang dibutuhkan. Hal ini penting untuk memastikan semua data dan model siap dipakai oleh program secara otomatis dan konsisten. Persiapan ini dimulai dengan memuat semua dataset untuk setiap skenario, yaitu *Original (bersih)*, *Skenario 1*, *Skenario 2*, dan *Skenario 3*. Sebuah fungsi khusus dirancang untuk menangani tugas ini, seperti yang dijelaskan dalam Kode Sumber 3.1. Fungsi loadDataset (baris 1) ini menerima lokasi folder sebagai *input*, kemudian mencari semua file gambar dan file teks *ground truth* di dalamnya (baris 2-3). Selanjutnya, fungsi ini akan memasangkan setiap gambar dengan teks asli yang berpasangan (baris 5) sebelum akhirnya mengembalikan satu pasang data yang utuh dan siap digunakan (baris 7).

```
function:  Pemuatan Dataset
   input:  Direktori dari dataset dokumen scan
  output:  Dataset yang telah dimuat di variable lingkungan

1  Function loadDataset(folderPath):
2    image_files = find_all_images_in(folderPath)
3    text_files = find_all_texts_in(folderPath)
4
5    dataset = pair_images_with_texts(image_files, text_files -
     Pasangkan pasangan image dengan ground truth nya
6
7    RETURN dataset
```

Kode Sumber 3.1 Insialisasi Proyek dan Persiapan Dokumen



Gambar 3.4 Jenis *Noise* pada Dataset Dokumen Tergradasi

Setelah semua dataset berhasil dimuat, langkah berikutnya adalah menyiapkan setiap model dan konfigurasi OCR yang akan dievaluasi.

```
function:  Inisialisasi model OCR standalone dan hybrid
   input:  Model OCR pre-trained
  output:  Model OCR standalone dan hybrid yang telah terkonfigurasi dan
           siap digunakan

        1  Function initializeModels():
        2    tesseractocr = TesseractModel()
        3    doctr = DocTRModel()
        4    paddleocr = PaddleOCRModel()
        5    easyocr = EasyOCRModel()
        6
        7    Standalone_models = { - melakukan grouping untuk
             aksesibilitas
        8      "Tesseract": tesseract,
        9      "DocTR": doctr,
       10      "PaddleOCR": paddle,
       11      "EasyOCR": easy
       12    }
       13
       14    hybrid_models = { - insialisasi model hybrid
       15      "Tess_rec_DocTR_det": new
               HybridModel(recognizer=tesseract, detector=doctr),
       16      "Tess_rec_Easy_det": new HybridModel(recognizer=tesseract,
               detector=easy),
       17      // dan seterusnya untuk semua konfigurasi OCR hybrid
       18    }

       19    RETURN standalone_models, hybrid_models
```

Kode Sumber 3.2 Inisialisasi dan Konfigurasi Model OCR

Proses ini ditangani oleh fungsi initializeModels (baris 1) yang bertujuan untuk membuat objek di dalam program untuk setiap kandidat pengujian. Di dalam fungsi ini, setiap model *standalone* seperti Tesseract, DocTR, PaddleOCR, dan EasyOCR dibuat sebagai objek individual terlebih dahulu (baris 2-5). Objek-objek ini kemudian dikelompokkan ke dalam satu grup bernama *standalone*_models untuk kemudahan akses nanti (baris 7-12). Proses serupa juga dilakukan untuk setiap konfigurasi OCR *Hybrid*, di mana setiap kombinasi *recognizer* dan detektor didefinisikan sebagai sebuah objek *Hybrid*Model yang baru (baris 14-17). Pada akhirnya, fungsi ini mengembalikan dua konfigurasi model yang telah siap untuk dievaluasi pada tahap selanjutnya (baris 19). Dengan selesainya persiapan ini, sistem telah siap untuk memasuki tahap eksekusi pengujian berikutnya.

3.3.2 Alur Kerja Evaluasi

Selanjutnya, pada bagian ini akan diuraikan fungsi utama yang menjadi bagian utama dalam alur pengujian ini. Fungsi pertama adalah fungsi untuk menjalankan skenario yang menggunakan pra-pemrosesan pada input yang diharapkan dapat mengubah gambar dengan sedemikian rupa sehingga menghasilkan kinerja model OCR yang lebih baik. Skenario dengan pra-pemrosesan ini adalah implementasi yang telah dijelaskan pada subbab 3.1.4. Fungsi ini dirancang untuk mengambil gambar hasil potongan dari detektor (*bounding box*) dan mengubahnya menjadi masukan yang standar bagi model *recognizer*. Fungsi yang diilustrasikan pada Kode Sumber 3.3 di bawah ini bertanggung jawab untuk menerapkan

serangkaian langkah perbaikan pada setiap gambar potongan (*bounding box*) sebelum gambar tersebut dibaca. Tujuannya adalah untuk menormalisasi masukan bagi *recognizer*.

```
function:  Penambahan pra-pemrosesan pada gambar yang dideteksi pada
           level bounding box
   input:  Gambar potongan kata yang dideteksi pada level bounding box
  output:  Hasil gambar yang sudah dinormalisasi dengan beberapa aspek
           pemrosesan

        1  Function preprocessImage(inputImage):
        2    // Menyeragamkan ukuran tinggi gambar
        3    preprocessedImage = resize_image(inputImage, height = 64)
        4
        5    // Menambahkan bingkai (padding) di sekeliling gambar
        6    preprocessedImage = add_padding(preprocessedImage,
             border_size = 5)
        7
        8    // Mengubah gambar menjadi
             hitam putih (grayscale)
        9    preprocessedImage = convert_to_grayscale(preprocessedImage)
       10
       11    RETURN preprocessedImage
```

Kode Sumber 3.3 Fungsi Penambahan Pra-pemrosesan

Berdasarkan Kode Sumber 3.3 di atas, langkah pertama adalah penyeragaman ukuran (baris 3). Fungsi resize_image dipanggil untuk mengubah inputImage sehingga tingginya menjadi 64 piksel. Hasil dari proses ini kemudian disimpan ke dalam variabel preprocessedImage. Langkah ini sangat penting karena kebanyakan model *recognizer* dilatih pada gambar dengan dimensi tertentu, sehingga masukan yang ukurannya konsisten akan meningkatkan kemampuannya untuk mengenali karakter dengan benar. Selanjutnya dilakukan penambahan padding (baris 6). Variabel preprocessedImage yang kini berisi gambar hasil *resize*, diteruskan ke fungsi add_padding untuk diberi bingkai selebar 5 piksel di sekelilingnya. Hasilnya kemudian diperbarui kembali ke variabel preprocessedImage yang sama. Tujuannya adalah untuk memberikan ruang di sekitar teks dan memastikan tidak ada bagian karakter di tepian yang terpotong oleh *bounding box* yang terlalu ketat. Langkah terakhir adalah konversi ke *grayscale* (baris 9). Gambar yang sudah diberi *padding* dari variabel preprocessedImage diproses oleh fungsi convert_to_grayscale untuk menghilangkan informasi warna. Ini membantu model untuk fokus hanya pada fitur bentuk dan kontras yang diperlukan untuk pengenalan teks. Terakhir, pada baris 11, variabel preprocessedImage yang telah melalui semua tahap pemrosesan dikembalikan oleh fungsi, siap untuk diproses oleh komponen *recognizer*.

Fungsi selanjutnya adalah fungsi runEvaluation yang merupakan kerangka utama dari keseluruhan proses pengujian. Fungsi ini memiliki satu tugas penting yaitu, menjalankan evaluasi terhadap satu konfigurasi model pada satu *dataset*. Penggunaan fungsi ini untuk semua pengujian memastikan bahwa setiap pendekatan diukur dengan cara yang sama, sehingga perbandingan hasil pada akhirnya menjadi hasil yang valid. Tujuan utama dari fungsi ini adalah untuk mengumpulkan data dari setiap gambar yang diproses, yang kemudian diubah menjadi sebuah tabel data (*DataFrame*) yang terstruktur yang memuat hasil dari evaluasi pada masing-masing pengujian.

```
function:  Evaluasi model OCR terhadap dataset dokumen scan
   input:  Model OCR yang dievaluasi dan dataset yang telah dimuat
  output:  Hasil uji coba berupa nilai metrik evaluasi OCR

 1  Function runEvaluation(model_to_test, dataset):
 2    // Inisialisasi list untuk menampung hasil evaluasi
 3    result_list =[]
 4
 5    // iterasi ke seluruh isi dari dataset
 6    for each (image, ground_truth_text) in dataset:
 7      // mengakses nama file dari gambar yang dievaluasi
 8      image_filename = get_filename_from(image)
 9
10      // menghitung waktu pemrosesan pada tiap gambar
11      start_time = getCurrentTime()
12
13      // memulai prediksi pada gambar yang dievaluasi
14      predicted_text = model_to_test.predict(image)
15
16      // mencatat waktu setelah prediksi selesai
17      time_taken = getCurrentTime() - start_time
18
19      // menghitung metrik evaluasi
20      wer = calculateWER(ground_truth_text, predicted_text)
21      cer = calculateCER(ground_truth_text, predicted_text)
22      bleu = calculateBLEU(ground_truth_text, predicted_text)
23
24      result_list.add({
25        "filename": image_filename,
26        "ground_truth": ground_truth_text,
27        "predicted_text": predicted_text,
28        "wer": wer,
29        "cer": cer,
30        "time": time_taken
31      })
32
33      END FOR
34
35    // Mengubah daftar hasil menjadi sebuah DataFrame
36    results_dataframe = create_dataframe_from(results_list)
37
38    RETURN results_dataframe
```

Kode Sumber 3.4 Keseluruhan Alur Kerja Evaluasi

Proses di dalam fungsi ini diawali dengan inisialisasi sebuah *list* kosong bernama result_list pada baris 3, yang akan berfungsi untuk menampung semua hasil evaluasi. Inti dari fungsi ini adalah sebuah *loop* yang dimulai pada baris 6, yang akan berjalan untuk setiap pasang gambar dan teks *ground truth* di dalam dataset. Di dalam setiap iterasi, pertama-tama akan diambil dan disimpan untuk nama file dari gambar yang sedang dievaluasi (baris 8). Selanjutnya, untuk mengukur durasi proses evaluasi, program mencatat waktu mulai (start_time) pada baris 11, menjalankan proses prediksi OCR pada baris 14 untuk menghasilkan teks prediksi (predicted_text), dan segera setelah itu menghitung total waktu pemrosesan (time_taken) pada baris 17.

Setelah mendapatkan teks prediksi, program langsung melakukan perhitungan metrik kinerja untuk gambar tersebut, meliputi WER, CER, dan BLEU (baris 20-22). Semua informasi ini termasuk nama file, teks asli, teks prediksi, serta skor metriknya kemudian dikumpulkan dan

disimpan sebagai satu kesatuan ke dalam result_list (baris 24-31). Setelah iterasi selesai untuk semua gambar dalam dataset (END FOR pada baris 33), daftar yang berisi hasil-hasil ini diubah menjadi sebuah tabel data (DataFrame) pada baris 36. DataFrame inilah yang menjadi keluaran akhir dari fungsi (baris 38), siap untuk disimpan dan dianalisis lebih lanjut.

3.3.3 Menjalankan Semua Pengujian

Setelah semua fungsi persiapan selesai dibuat, langkah terakhir adalah membuat sebuah skrip utama yang akan menjalankan semua pengujian secara otomatis. Skrip ini bisa dianggap sebagai elemen yang mengatur jalannya seluruh eksperimen dari awal hingga akhir. Tujuannya adalah untuk memastikan semua model diuji pada semua skenario dengan cara yang sama. Cara kerja skrip ini pada dasarnya menggunakan perulangan di dalam perulangan (*nested loops*) untuk menjalankan setiap kombinasi pengujian yang sudah direncanakan.

```
function:  Menjalankan semua skenario percobaan terhadap semua skenario
           dataset yang telah dibuat
   input:  Dataset dan Model OCR
  output:  Hasil uji coba berupa nilai metrik evaluasi OCR pada tiap
           skenario yang diujikan

 1  function main():
 2    // memuat dataset yang telah dibuat sebelumnya
 3    datasets = load_dataset()
 4
 5    // inisialisasi model
 6    standalone_models, hybrid_models = initializeModels()
 7
 8    // list untuk menampung hasil akhir
 9    results = []
11
12    // Pengujian bagian 1 (model Standalone)
13    for each name, model in standalone_models:
14      result_df = runEvaluation(model, datasets)
15      results.add({
16        "name": name,
17        "type": "Standalone",
18        "data": results_df
20      })
21
22    END FOR
23
24    // Pengujian bagian 2 (model Hybrid)
25    for each name, model in hybrid_models:
26      result_df = runEvaluation(model, datasets)
27      results.add({
28        "name": name,
29        "type": "Hybrid",
30        "data": results_df
31      })
32
33    END FOR
34
35    // Pengujian bagian 3 (model Hybrid dengan pra-pemrosesan)
36    for each name, model in hybrid_models:
37      preprocessed_model = configure_for_preprocessing(model)
38      result_df = runEvaluation(preprocessed_model, datasets)
39      results.add({
40        "name": name,
41        "type": "Hybrid + Preprocessing",
```

```
42          "data": results_df
43        })
44
45    END FOR
46
47    // menyimpan semua hasil evaluasi dalam sebuah file CSV
48    results.to_csv(results)
```

Kode Sumber 3.5 Fungsi Utama Pengujian Model OCR

Sub-bab terakhir ini menjelaskan tentang fungsi utama yang berfungsi sebagai pemersatu dari keseluruhan proses pengujian. Seperti yang dapat dilihat pada Kode Sumber 3.5, fungsi ini berperan untuk memanggil semua fungsi yang telah dipersiapkan sebelumnya guna menjalankan ketiga bagian pengujian dan menyimpan hasil akhirnya. Proses diawali dengan memuat dataset (baris 3) dan melakukan inisialisasi semua model OCR (baris 7). Selanjutnya, menyiapkan sebuah daftar kosong (results) untuk menampung semua hasil akhir (baris 11). Inti dari skrip utama ini adalah tiga putaran (loop) pengujian yang terpisah, yang masing-masing dirancang untuk satu bagian eksperimen. Putaran pertama (baris 13-22) mengeksekusi Pengujian Bagian 1, di mana program melakukan iterasi untuk setiap model *standalone*, memanggil fungsi runEvaluation untuk mendapatkan DataFrame hasilnya, lalu menyimpannya ke dalam *list* results. Putaran kedua (baris 24-33) menjalankan Pengujian Bagian 2 dengan proses yang sama seperti bagian sebelumnya namun kali ini untuk setiap model *hybrid*, di mana hasilnya disimpan juga ke dalam *list* results.

Putaran ketiga (baris 35-45) adalah untuk Pengujian Bagian 3. Di sini, terdapat satu langkah tambahan yang penting. Sebelum memanggil runEvaluation, setiap model *hybrid* pertama-tama dikonfigurasi untuk mengaktifkan pra-pemrosesan melalui fungsi configure_for_preprocessing (baris 37). Hasil dari model yang telah ditingkatkan inilah yang kemudian dievaluasi dan disimpan. Setelah ketiga putaran pengujian selesai, *list* results yang kini berisi semua data kinerja dari seluruh eksperimen, disimpan menjadi sebuah file CSV tunggal pada baris 48. File inilah yang menjadi data untuk semua analisis yang disajikan pada Bab 4.

BAB 4 Hasil dan Pembahasan

Bab ini menyajikan dan menganalisis hasil dari serangkaian eksperimen yang dilakukan untuk meningkatkan akurasi OCR pada dokumen akademik yang diberi *noise*. Bagian pertama, Hasil Penelitian, memaparkan data pengujian secara kuantitatif, mulai dari kinerja dasar model, dampak *noise*, hingga hasil uji dari penambahan *pipeline* pra-pemrosesan yang diusulkan. Bagian kedua, pembahasan, akan menjelaskan temuan-temuan tersebut lebih lanjut.

4.1 Hasil Penelitian

Bagian ini menyajikan seluruh data yang diperoleh dari serangkaian uji coba yang telah dijalankan. Data disajikan dalam bentuk tabel untuk memudahkan pembacaan dan perbandingan. Penyajian hasil akan mengikuti alur kerja penelitian yang telah ditetapkan sebelumnya, dimulai dari hasil kinerja dasar model *standalone*, dilanjutkan dengan dampak degradasi pada skenario *noise*, hasil investigasi pendekatan OCR *hybrid*, hingga validasi akhir dari *pipeline* pra-pemrosesan. Penting untuk dicatat bahwa bagian ini hanya berfokus pada pemaparan data. Interpretasi dan pembahasan mendalam terhadap hasil-hasil ini akan diuraikan secara terpisah pada bab selanjutnya.

4.1.1 Lingkungan Implementasi

Lingkungan implementasi merupakan lingkungan dimana sistem akan dibangun. Lingkungan implementasi dibagi menjadi lingkungan implementasi perangkat keras dan lingkungan implementasi perangkat lunak.

Tabel 4.1 Spesifikasi Lingkungan Eksperimen

Kategori	Komponen	Spesifikasi / Versi
Perangkat Keras	GPU	NVIDIA GeForce RTX 4060
	CPU	Intel Core i7-11800H
	RAM	32 GB DDR5
Perangkat Lunak	Bahasa Pemrograman	Python 3.9
	Framework	PyTorch 1.12
	Model & Library	DocTR 0.5.1, EasyOCR 1.6.2, PaddleOCR 2.6, Tesseract 5.2, OpenCV 4.6

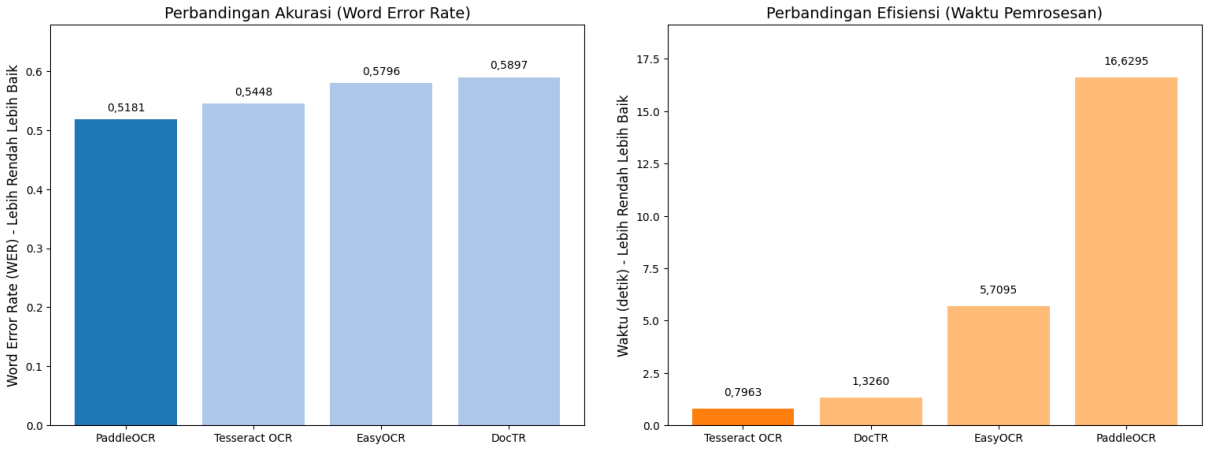
4.1.2 Kinerja Model Baseline OCR pada Data Bersih

Tahap pengujian pertama dalam penelitian ini adalah mengukur kinerja dasar (*baseline*) dari setiap model OCR *standalone* pada dataset yang bersih, yaitu Skenario Original. Evaluasi ini penting guna menjadikan acuan untuk semua pengujian pada skenario dengan *noise*. Data kinerja yang telah diuji coba mencakup metrik akurasi dan efisiensi, disajikan secara lengkap pada Tabel 4.2.

Tabel 4.2 Kinerja Baseline Model OCR pada Skenario Original

Model OCR	CER	WER	BLEU Score	Waktu (detik)
DocTR	0,3627	0,5897	0,7466	1,3260
EasyOCR	0,3317	0,5796	0,4085	5,7095
PaddleOCR	0,3624	0,5181	0,7328	16,6295
Tesseract OCR	0,3452	0,5448	0,4875	0,7963

Berdasarkan data pada Tabel 4.2, tercatat bahwa Paddle OCR menghasilkan akurasi tertinggi dengan *Word Error Rate* (WER) terendah yaitu 0,5181. Untuk akurasi pada tingkat karakter, Easy OCR menunjukkan *Character Error Rate* (CER) terendah dengan nilai 0,3317. Dari segi kelancaran kalimat yang diukur dengan BLEU Score, DocTR menunjukkan hasil tertinggi dengan skor 0,7466. Terakhir, dari segi efisiensi, Tesseract OCR menegaskan posisinya sebagai model dengan waktu pemrosesan tercepat, yaitu hanya 0,7963 detik per halaman. Hasil ini menetapkan keberagaman kinerja untuk setiap model pada kondisi ideal dan akan menjadi acuan untuk evaluasi pada skenario *noise*.



Gambar 4.1 Kinerja Baseline Model OCR pada Skenario Original

4.1.3 Kinerja Model OCR pada Data dengan *Noise*

Setelah kinerja *baseline* ditetapkan, pengujian dilanjutkan pada dataset yang telah diberi *noise* untuk mengukur ketahanan dari setiap model. Bagian ini menyajikan data kinerja setiap model *standalone* saat diuji pada tiga skenario dengan tingkat *noise* yang berbeda, yaitu Skenario 1 (*noise* rendah), Skenario 2 (*noise* sedang), dan Skenario 3 (*noise* tinggi). Hasil lengkap dari pengujian ini disajikan pada Tabel 4.3.

Tabel 4.3 Kinerja Model OCR pada Skenario *Noise*

Model OCR	Skenario	CER	WER	BLEU Score	Waktu (detik)
DocTR	Skenario 1	0,3667	0,6188	0,6915	0,7354
	Skenario 2	0,3751	0,6978	0,4874	0,7501
	Skenario 3	0,4899	0,7950	0,3625	0,7349
Easy OCR	Skenario 1	0,3414	0,5977	0,3801	3,1160
	Skenario 2	0,4396	0,7383	0,2574	3,3373
	Skenario 3	0,5989	0,8045	0,2155	2,8164
Paddle OCR	Skenario 1	0,3512	0,5132	0,6724	7,7155
	Skenario 2	0,3584	0,5909	0,4846	7,6180
	Skenario 3	0,5279	0,7179	0,3617	8,1892
Tesseract OCR	Skenario 1	0,3823	0,6654	0,4188	0,8382
	Skenario 2	0,6263	0,9197	0,1570	0,8936
	Skenario 3	0,7447	0,9978	0,1037	0,9048

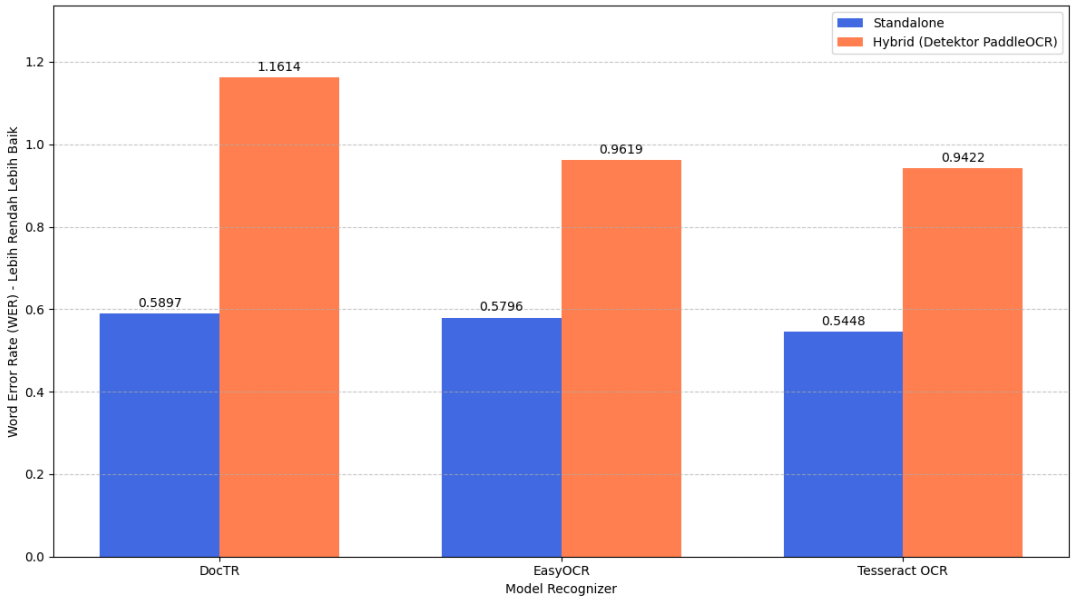
Data pada tabel di atas menunjukkan sebuah pola penurunan kinerja model OCR seiring bertambahkan intensitas *noise*. Untuk semua model, nilai *error rate* (baik CER maupun WER) secara umum menunjukkan tren peningkatan seiring dengan bertambahnya intensitas *noise* dari Skenario 1 ke Skenario 3. Paddle OCR tercatat sebagai model dengan akurasi tertinggi secara konsisten di ketiga skenario *noise*, dengan nilai WER 0,5132 (rendah), 0,5909 (sedang), dan 0,7179 (tinggi). Model DocTR menempati posisi kedua dalam hal akurasi pada kondisi *noise* ini. Sebaliknya, Tesseract OCR menunjukkan degradasi kinerja yang paling tajam, terutama pada Skenario 2 dan 3, di mana nilai WER-nya mendekati 1,0 yang menandakan tingkat kesalahan yang sangat tinggi. Dari segi efisiensi, Tesseract OCR tetap menjadi model dengan waktu pemrosesan tercepat di semua skenario *noise*.

4.1.4 Hasil Evaluasi OCR *Hybrid* (Detektor Paddle OCR)

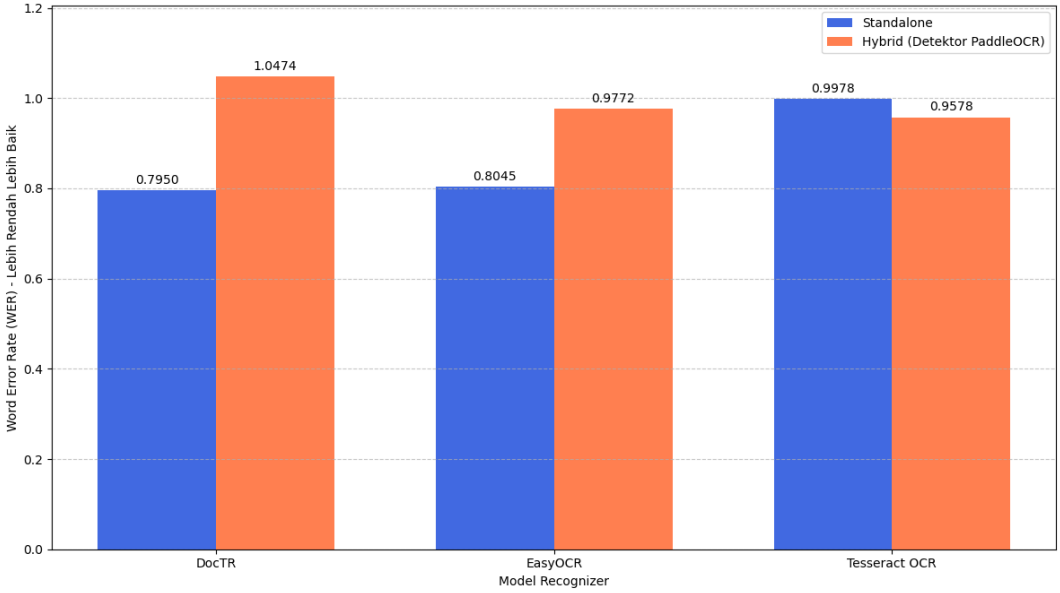
Setelah performa model-model *standalone* ditetapkan, tahap selanjutnya adalah menguji kelayakan pendekatan OCR *Hybrid*. Bagian ini menyajikan hasil dari konfigurasi OCR *Hybrid* di mana detektor dari Paddle OCR digunakan sebagai komponen deteksi teks, yang kemudian dikombinasikan dengan modul *recognizer* dari model DocTR, Easy OCR, dan Tesseract OCR. Tabel 4.4 menyajikan perbandingan kinerja secara langsung antara model *standalone* dengan konfigurasi OCR *Hybrid* pasangannya.

Tabel 4.4 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* (Detektor Paddle OCR)

***Recognizer* Model**	**Skenario**	**WER (*Standalone*)**	**WER (*Hybrid*)**
DocTR	Original	0,5897	1,1614
	Skenario 1	0,6188	1,1372
	Skenario 2	0,6978	1,0820
	Skenario 3	0,7950	1,0474
Easy OCR	Original	0,5796	0,9619
	Skenario 1	0,5977	0,9581
	Skenario 2	0,7383	0,9739
	Skenario 3	0,8045	0,9772
Tesseract OCR	Original	0,5448	0,9422
	Skenario 1	0,6654	0,9463
	Skenario 2	0,9197	0,9513
	Skenario 3	0,9978	0,9578



Gambar 4.3 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario Original (Detektor Paddle OCR)



Gambar 4.4 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario *Noise* Tinggi (Detektor Paddle OCR)

Pengujian pada bagian ini bertujuan untuk menguji penggabungan detektor teks dari PaddleOCR dengan *recognizer* dari model lain agar dapat menciptakan sistem *hybrid* yang lebih unggul. Namun, visualisasi pada Gambar 4.3 dan Gambar 4.4 menunjukkan bahwa hal tersebut tidak terbukti, bahkan pada kondisi data yang paling ideal (bersih) sekalipun. Seperti yang terlihat jelas pada gambar di atas, untuk setiap model *recognizer* yang diuji, batang oranye yang merepresentasikan kinerja OCR *Hybrid* secara konsisten jauh lebih tinggi daripada batang biru yang merepresentasikan kinerja OCR *Standalone*. Karena sumbu vertikal mengukur *Word Error Rate* (WER) di mana nilai yang semakin rendah semakin baik, hal ini menunjukkan penurunan performa yang signifikan.

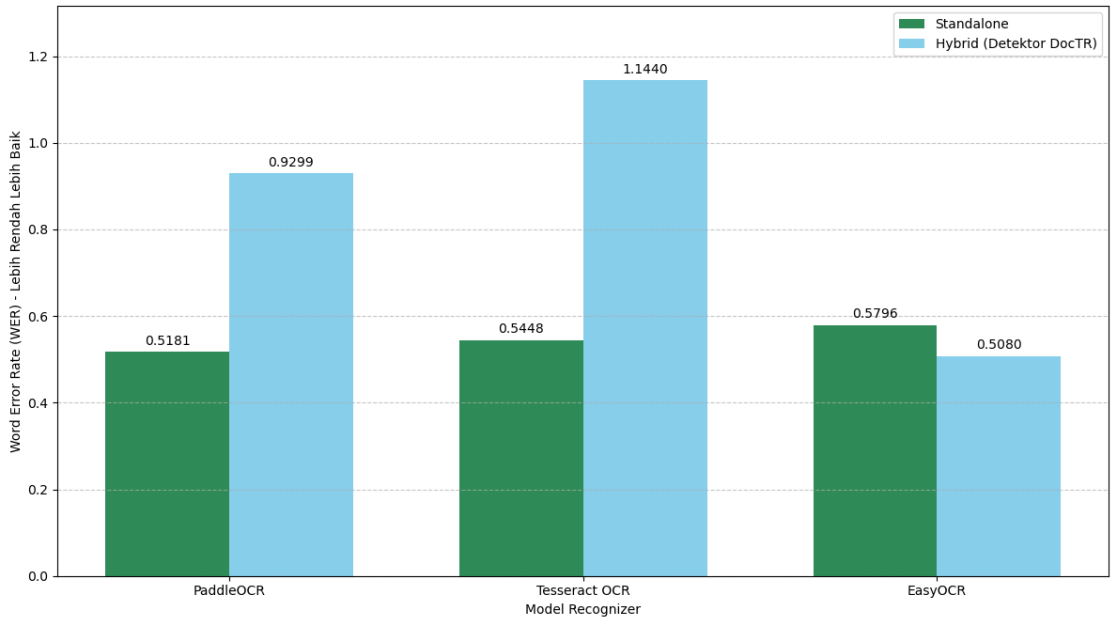
Untuk *Recognizer* DocTR, kinerja *standalone* (batang biru) untuk metrik evaluasi WER adalah 0,5897. Namun, saat menggunakan mode *hybrid* (batang oranye), tingkat kesalahannya melonjak drastis menjadi 1,1614, yang mana adalah hampir dua kali lipat lebih buruk. Pola kegagalan yang sama terulang pada skenario *Recognizer* Easy OCR. Batang biru menunjukkan WER sebesar 0,5796, yang kemudian meningkat tajam pada mode *hybrid* menjadi 0,9619. Berlaku pula hal yang sama pada skenario *Recognizer* Tesseract OCR, yang kinerja awalnya (WER sebesar 0,5448) juga memburuk secara saat dipertemukan dengan skenario OCR *hybrid*, dengan WER mencapai 0,9422. Alih-alih terjadi kecocokan, yang terjadi justru adalah penurunan performa secara drastis. Hal ini berarti terdapat ketidakcocokan antara komponen detektor Paddle OCR dengan *recognizer* dari model lainnya (DocTR, Easy OCR, dan Tesseract OCR).

4.1.5 Hasil Evaluasi OCR *Hybrid* (Detektor DocTR)

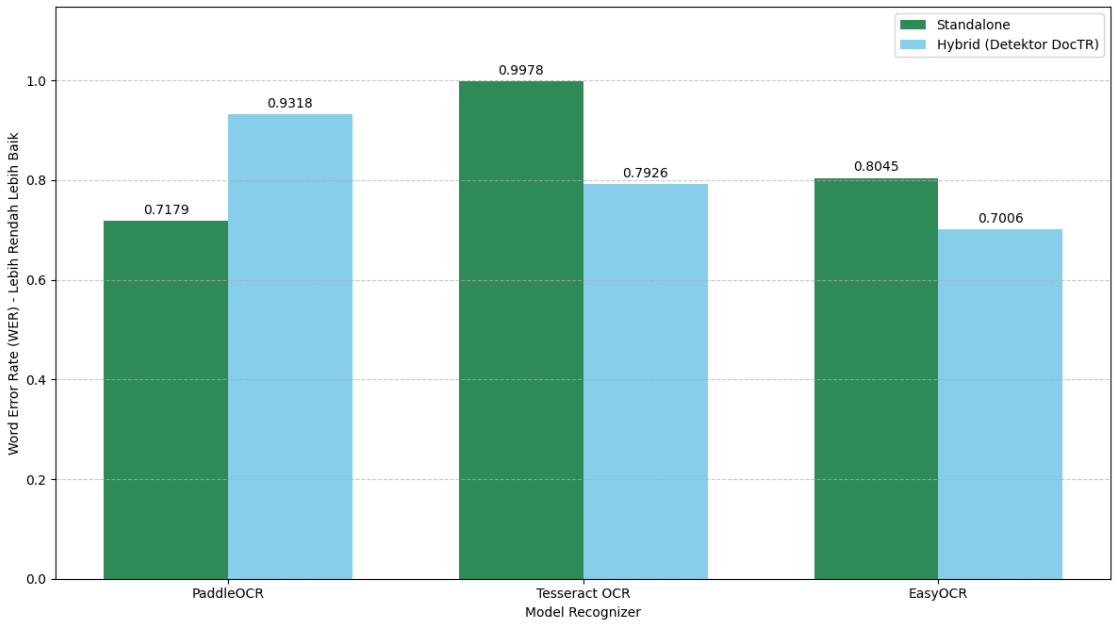
Untuk pemaparan berikutnya akan dilanjutkan dengan menguji Detektor DocTR yang dinilai memiliki performa yang baik pada uji coba *Standalone*. Bagian ini menyajikan hasil dari konfigurasi *Hybrid* di mana detektor DocTR dipasangkan dengan *recognizer* dari Paddle OCR, Tesseract OCR, dan Easy OCR. Tabel 4.5 menyajikan perbandingan kinerja WER secara langsung.

Tabel 4.5 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* (Detektor DocTR)

***Recognizer* Model**	**Skenario**	**WER (*Standalone*)**	**WER (*Hybrid*)**
Paddle OCR	Original	0,5181	0,9299
	Skenario 1	0,5132	0,9167
	Skenario 2	0,5909	0,9223
	Skenario 3	0,7179	0,9318
Tesseract OCR	Original	0,5448	1,144
	Skenario 1	0,6654	1,119
	Skenario 2	0,9197	0,7854
	Skenario 3	0,9978	0,7926
Easy OCR	Original	0,5796	0,5080
	Skenario 1	0,5977	0,5129
	Skenario 2	0,7383	0,6410
	Skenario 3	0,8045	0,7006



Gambar 4.5 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario Original (Detektor DocTR)



Gambar 4.6 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario *Noise* Tinggi (Detektor DocTR)

Visualisasi pada Gambar 4.5 menampilkan hasil yang cukup menarik yang diambil pada Skenario Original. Di satu sisi, kita melihat kegagalan saat detektor DocTR dipasangkan dengan *recognizer* Paddle OCR dan Tesseract OCR. Untuk Paddle OCR, tingkat kesalahan (WER) melonjak dari 0,5181 menjadi 0,9299. Penurunan performa yang drastic juga terjadi pada Tesseract OCR di mana tingkat kesalahannya meningkat lebih dari dua kali lipat, dari 0,5448 menjadi 1,1440. Ini menunjukkan bahwa bahkan detektor yang sangat kuat dan modern pun bisa menjadi tidak cocok dan justru merusak kinerjanya jika dipasangkan dengan *recognizer* yang tidak dirancang untuknya.

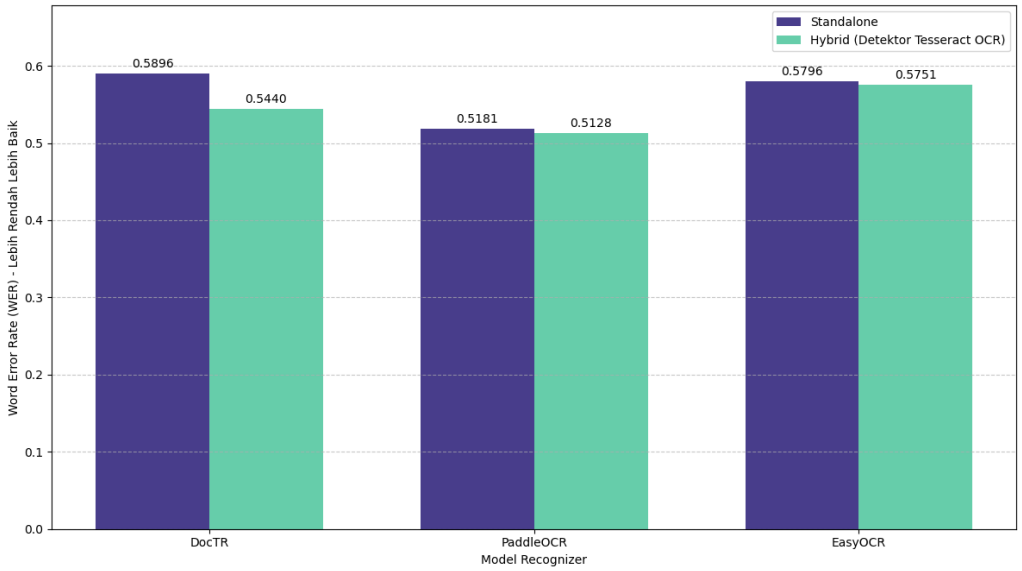
Namun, terdapat hal menarik yang dapat kita amati dan telusuri. Di sisi lain, visualisasi ini juga menunjukkan keberhasilan pertama dari pendekatan OCR *hybrid* dalam pengujian ini. Saat dipasangkan dengan *recognizer* Easy OCR, detektor DocTR berhasil menurunkan tingkat *error*. Seperti yang terlihat jelas pada diagram, batang biru muda (*Hybrid*) untuk Easy OCR secara nyata lebih pendek daripada batang hijau (*Standalone*), dengan nilai WER yang berhasil ditekan dari 0,5796 menjadi 0,5080. Lebih lanjut lagi, data pada Tabel 4.5 menunjukkan fenomena menarik yang tidak terlihat pada visualisasi Skenario Original. Pada Gambar 4.6 berkaitan dengan Skenario 3 (*noise* tinggi), kinerja *standalone* Tesseract OCR hampir gagal total dengan WER 0,9978. Namun, detektor DocTR mampu meningkatkan kinerjanya, menekan WER secara signifikan menjadi 0,7926. Hasil dari pengujian detektor DocTR ini sangat penting. Ini membuktikan bahwa sebuah detektor yang kuat bukanlah solusi yang tepat untuk setiap *use case*, namun dapat secara efektif meningkatkan *recognizer* lain yang lebih rentan terhadap *noise*, meskipun bisa juga tidak cocok dengan *recognizer* lainnya.

4.1.6 Hasil Evaluasi OCR *Hybrid* (Detektor Tesseract OCR)

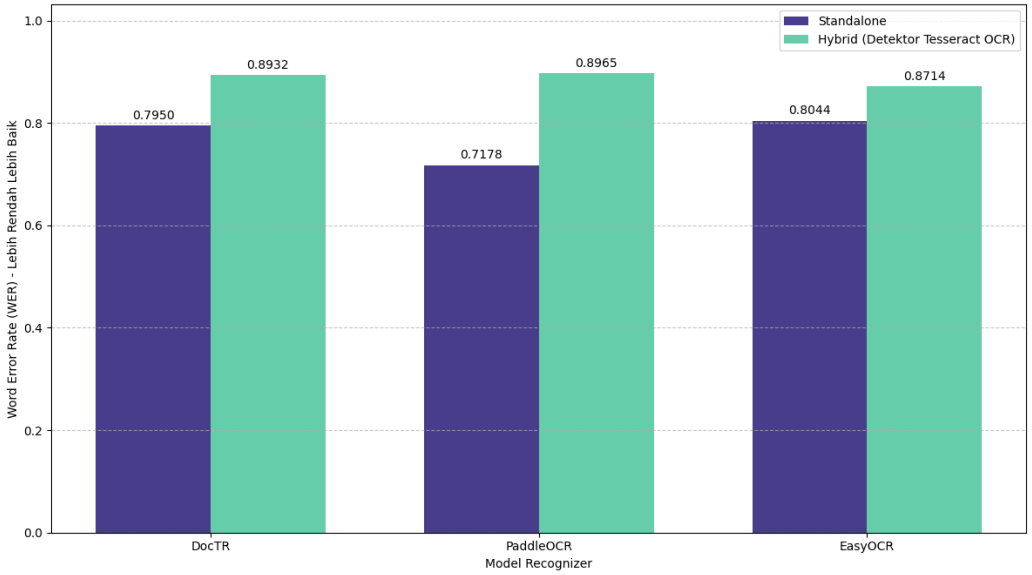
Selanjutnya dalam pemaparan hasil evaluasi model OCR, pengujian kali ini dilakukan untuk mengevaluasi kinerja konfigurasi yang menggunakan Detektor dari Tesseract OCR. Tabel 4.6 menyajikan perbandingan kinerja *Word Error Rate* (WER) antara model *standalone* dengan konfigurasi OCR *Hybrid* yang menggunakan detektor Tesseract OCR.

Tabel 4.6 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* (Detektor Tesseract OCR)

***Recognizer* Model**	**Skenario**	**WER (*Standalone*)**	**WER (*Hybrid*)**
DocTR	Original	0,5896	0,5440
	Skenario 1	0,6187	0,6037
	Skenario 2	0,6978	0,8576
	Skenario 3	0,7950	0,8932
Paddle OCR	Original	0,5181	0,5128
	Skenario 1	0,5132	0,5995
	Skenario 2	0,5908	0,8481
	Skenario 3	0,7178	0,8965
Easy OCR	Original	0,5796	0,5751
	Skenario 1	0,5976	0,6066
	Skenario 2	0,7382	0,8300
	Skenario 3	0,8044	0,8714



Gambar 4.7 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario Original (Detektor Tesseract OCR)



Gambar 4.8 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario *Noise* Tinggi (Detektor Tesseract OCR)

Hasil yang disajikan pada Tabel 4.6 menunjukkan sebuah pola yang sangat jelas yaitu, detektor Tesseract hanya memberikan peningkatan pada Skenario Original dan gagal pada saat dihadapkan pada skenario *noise*. Visualisasi untuk Skenario Original pada Gambar 4.7 menampilkan sedikit keberhasilan dari pendekatan ini. Untuk ketiga *recognizer* (DocTR, Paddle OCR, dan Easy OCR), batang yang merepresentasikan kinerja *hybrid* (hijau) terlihat sedikit lebih pendek daripada batang *standalone* (biru). Sebagai contoh, pada *recognizer* DocTR, tingkat kesalahan (WER) turun dari 0,5896 menjadi 0,5440. Hal ini menunjukkan bahwa pada gambar berkualitas tinggi, detektor Tesseract cukup mampu memberikan informasi tentang teks yang dapat dimengerti oleh *recognizer* lain.

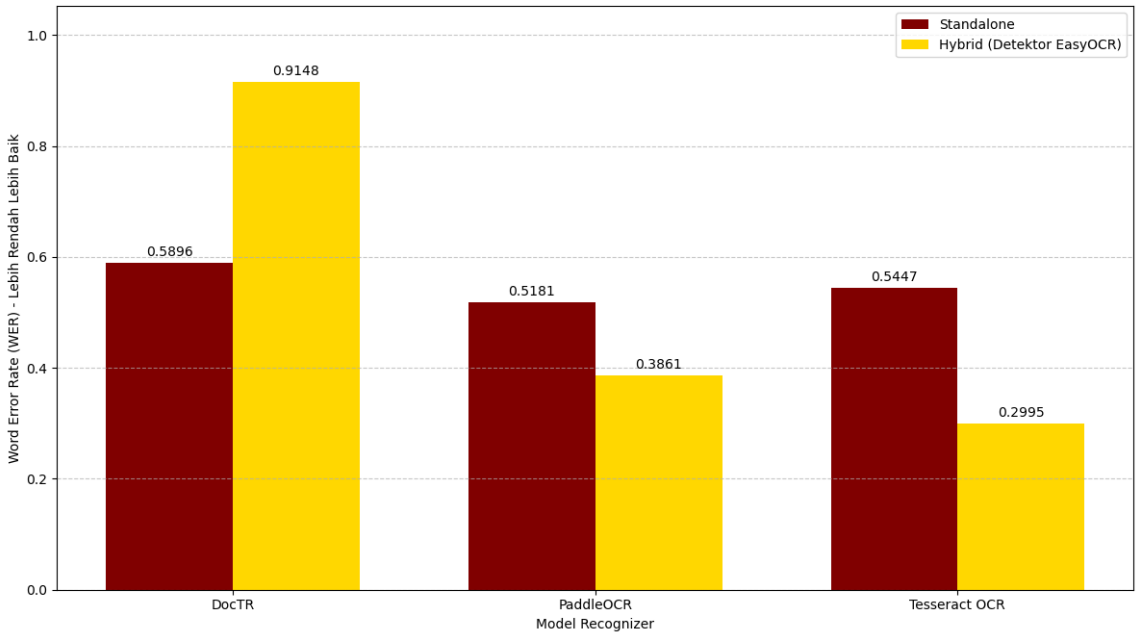
Berkaitan dengan skenario paling buruk dengan *noise* paling berat, batang *hybrid* kini jauh lebih tinggi daripada batang *standalone*, menandakan penurunan kinerja yang drastis. Sebagai contoh, seperti yang bisa kita lihat pada Gambar 4.8, kinerja *hybrid* (batang biru muda) untuk DocTR (WER 0,8932) menjadi jauh lebih buruk daripada kinerja *standalone* (batang biru tua) dengan nilai WER sebesar 0,7950. Hal yang sama terjadi pada Paddle OCR, di mana WER *hybrid*-nya (0,8965) lebih tinggi dari *standalone* (0,7178). Didapati dari hasil pengujian ini bahwa detektor Tesseract, meskipun cepat, tidak memiliki ketahanan terhadap *noise*. Kelemahannya dalam mendeteksi teks pada gambar yang diberi *noise* menyebabkan kesalahan yang kemudian memperburuk kinerja *recognizer* OCR yang dipasangkan dengannya. Ini membuktikan bahwa detektor Tesseract hanya merupakan pilihan yang layak untuk sistem yang memproses dokumen berkualitas tinggi dan tidak dapat diandalkan untuk skenario yang penuh dengan *noise* seperti yang ada pada dunia nyata.

4.1.7 Hasil Evaluasi OCR *Hybrid* (Detektor Easy OCR)

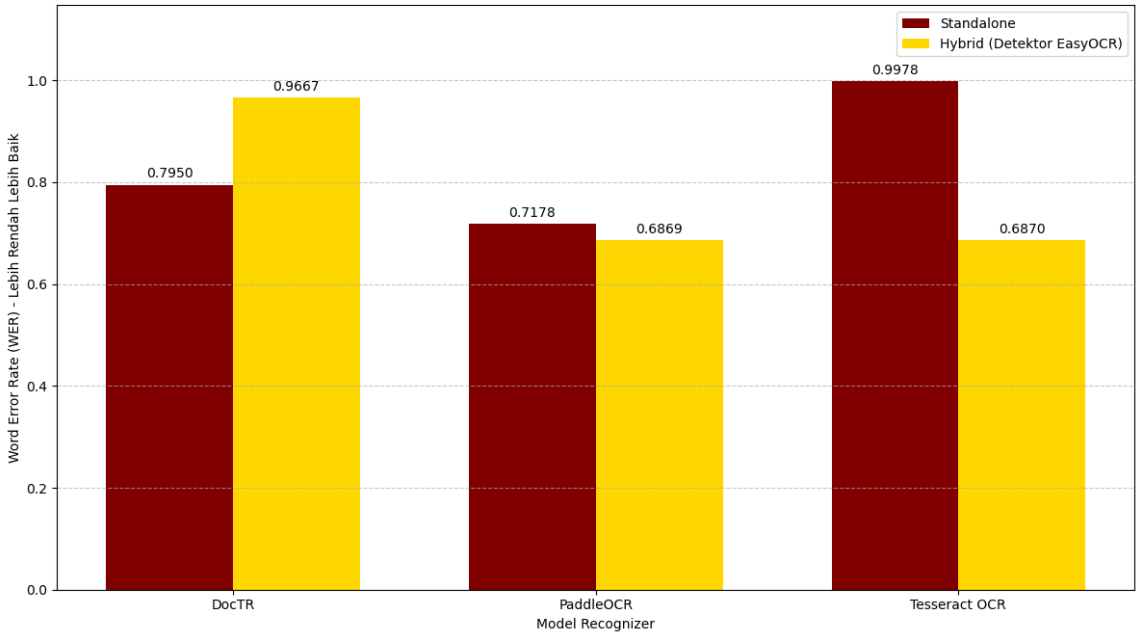
Sebagai pengujian terakhir dari scenario OCR *Hybrid*, selanjutnya akan dilakukan evaluasi kinerja konfigurasi yang menggunakan Detektor dari Easy OCR. Tabel 4.7 menyajikan perbandingan kinerja WER antara model *standalone* dengan konfigurasi OCR *hybrid* yang menggunakan detektor Easy OCR.

Tabel 4.7 Perbandingan Kinerja *Standalone* vs. OCR *Hybrid* (Detektor Easy OCR)

***Recognizer* Model**	**Skenario**	**WER (*Standalone*)**	**WER (*Hybrid*)**
DocTR	Original	0,5896	0,9148
	Skenario 1	0,6187	0,9275
	Skenario 2	0,6978	0,9447
	Skenario 3	0,7950	0,9667
Paddle OCR	Original	0,5181	0,3861
	Skenario 1	0,5132	0,4277
	Skenario 2	0,5908	0,5664
	Skenario 3	0,7178	0,6869
Tesseract OCR	Original	0,5447	0,2995
	Skenario 1	0,6653	0,3242
	Skenario 2	0,9197	0,5032
	Skenario 3	0,9978	0,6870



Gambar 4.9 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario Original (Detektor Easy OCR)



Gambar 4.10 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* pada Skenario *Noise* Tinggi (Detektor Easy OCR)

Pengujian terakhir pada scenario OCR *Hybrid* ini akan menguji konfigurasi yang menggunakan detektor dari Easy OCR. Hasil dari pengujian ini, yang divisualisasikan pada Gambar 4.9, menunjukkan temuan yang paling signifikan dari seluruh pengujian *hybrid*. Visualisasi pada Gambar 4.9 menampilkan hasil yang sangat beragam pada Skenario Original. Di satu sisi, kita melihat kegagalan saat detektor Easy OCR dipasangkan dengan *recognizer* DocTR. Batang kuning (*Hybrid*) menjulang tinggi di atas batang merah (*Standalone*), yang menunjukkan peningkatan kesalahan (WER) dari 0,5896 menjadi 0,9148

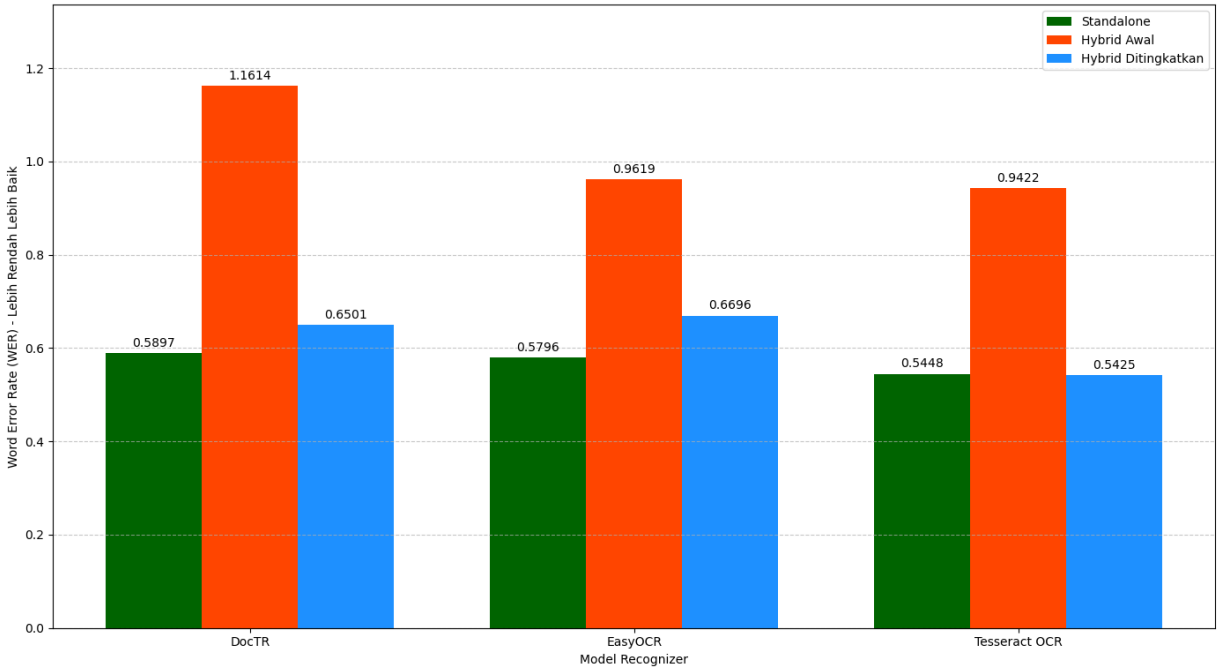
Namun, di sisi lain, visualisasi ini juga menunjukkan sebuah temuan yang menarik pada kombinasinya dengan dua *recognizer* lainnya. Saat dipasangkan dengan Paddle OCR, batang kuning terlihat lebih pendek, menandakan perbaikan kinerja di mana WER berhasil diturunkan dari 0,5181 menjadi 0,3861. Peningkatan yang paling menonjol terjadi pada Tesseract OCR. Seperti yang terlihat jelas, batang kuning untuk Tesseract secara signifikan lebih pendek, menunjukkan bahwa detektor Easy OCR berhasil mengurangi tingkat kesalahan Tesseract sebanyak hampir setengahnya, dari 0,5447 menjadi hanya 0,2995. Ini membuktikan bahwa detektor yang paling modern dan canggih belum tentu merupakan pasangan yang terbaik. Justru, detektor sederhana seperti Easy OCR ternyata memiliki tingkat kecocokan yang sangat tinggi dengan *recognizer* tertentu (terutama Tesseract OCR), menghasilkan peningkatan kinerja yang paling menonjol di antara semua konfigurasi OCR *Hybrid* yang diuji.

4.1.8 Hasil Uji Coba OCR *Hybrid* yang Ditingkatkan dengan Pra-pemrosesan (Detektor Paddle OCR)

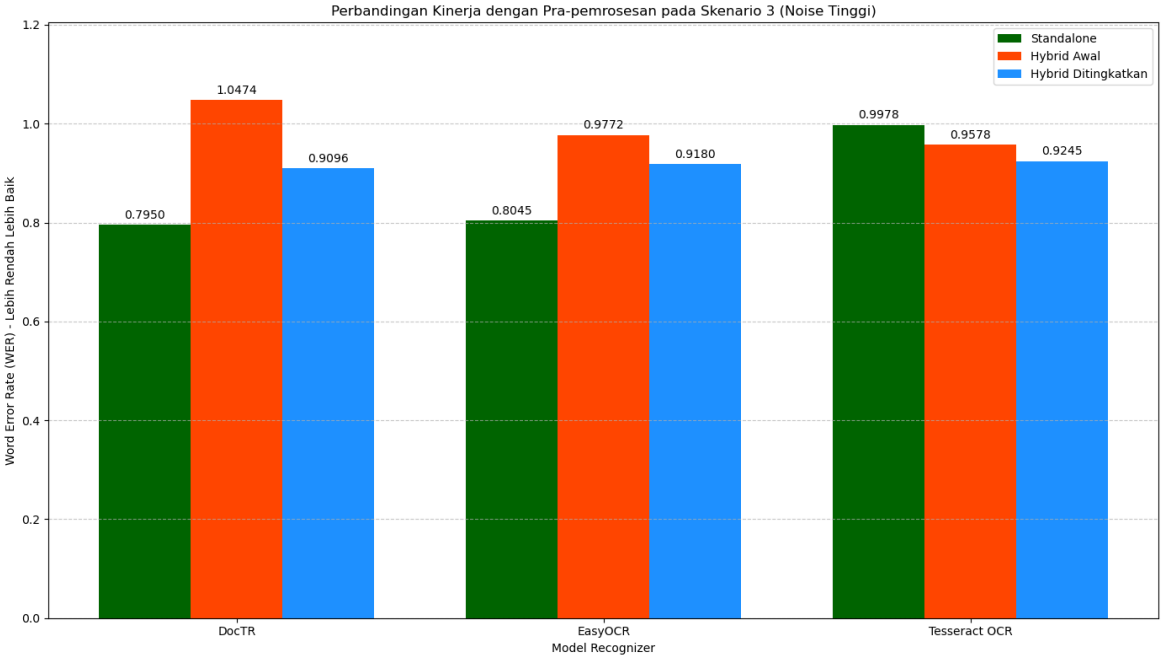
Sebagai tahap akhir dari serangkaian skenario pengujian model OCR, sebuah eksperimen dilakukan untuk menguji agar kinerja OCR *Hybrid* dapat ditingkatkan melalui pra-pemrosesan sederhana. Tabel 4.8 menyajikan perbandingan kinerja WER pada model OCR *Standalone*, sebelum (*Hybrid* Awal) dan sesudah (*Hybrid* Ditingkatkan) penerapan pra-pemrosesan.

Tabel 4.8 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* Awal vs. OCR *Hybrid* Ditingkatkan (Detektor Paddle OCR)

Model *Recognizer*	**Skenario**	**WER (*Standalone*)**	**WER (*Hybrid* Awal)**	**WER (*Hybrid* Ditingkatkan)**
DocTR	Original	0,5897	1,1614	0,6501
	Skenario 1	0,6188	1,1372	0,6733
	Skenario 2	0,6978	1,0820	0,8420
	Skenario 3	0,7950	1,0474	0,9096
Easy OCR	Original	0,5796	0,9619	0,6696
	Skenario 1	0,5977	0,9581	0,6908
	Skenario 2	0,7383	0,9739	0,8444
	Skenario 3	0,8045	0,9772	0,9180
Tesseract OCR	Original	0,5448	0,9422	0,5425
	Skenario 1	0,6654	0,9463	0,5928
	Skenario 2	0,9197	0,9513	0,8179
	Skenario 3	0,9978	0,9578	0,9245



Gambar 4.11 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario Original (Detektor Paddle OCR)



Gambar 4.12 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario *Noise* Tinggi (Detektor Paddle OCR)

Pada tahap ini, sebuah langkah pra-pemrosesan sederhana diterapkan pada *bounding box* yang dihasilkan oleh detektor sebelum diteruskan ke *recognizer*. Hasil dari upaya perbaikan ini disajikan pada Tabel 4.8 dan divisualisasikan pada Gambar 4.11 dan 4.12. Analisis visual pada Gambar 4.11 untuk Skenario Original menunjukkan bahwa langkah pra-pemrosesan ini memberikan perbaikan yang sangat signifikan. Untuk semua model *recognizer*, batang biru

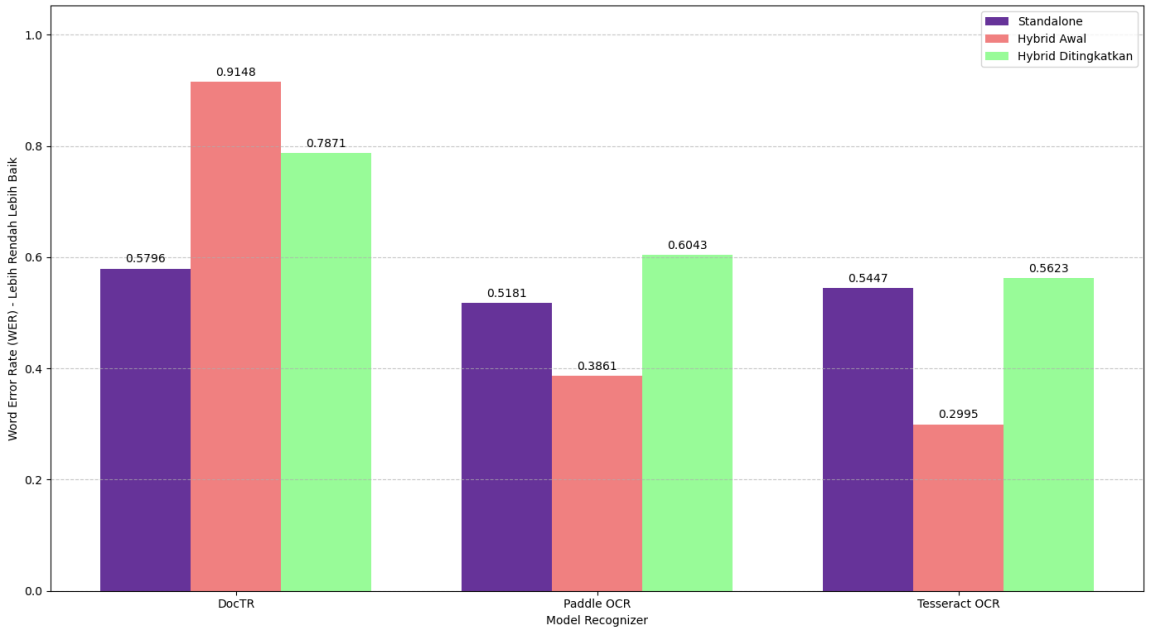
(*Hybrid Ditingkatkan*) lebih pendek daripada batang oranye (*Hybrid Awal*). Pada *recognizer* DocTR, tingkat kesalahan (WER) berhasil diturunkan dari 1,1614 menjadi 0,6501.

Selanjutnya, kinerja *Hybrid Ditingkatkan* (batang biru) dibandingkan dengan kinerja *Standalone* (batang hijau). Untuk *recognizer* DocTR dan Easy OCR, performa *Hybrid Ditingkatkan* masih belum mampu melampaui performa *standalone*-nya. Temuan yang paling penting muncul pada *recognizer* Tesseract OCR. Dalam kasus ini, pra-pemrosesan tidak hanya memperbaiki kinerja *hybrid*, tetapi juga membuatnya lebih unggul daripada kinerja *standalone*-nya. Pada data asli, WER *Hybrid Ditingkatkan* (0,5425) berhasil sedikit mengalahkan WER *Standalone* (0,5448). Keunggulan ini juga terlihat pada skenario *noise*, seperti yang ditunjukkan pada Gambar 4.12 untuk Skenario 3. Pra-pemrosesan sederhana terbukti merupakan teknik yang sangat ampuh untuk memperbaiki sistem OCR *Hybrid* yang tidak berfungsi seoptimal model OCR *Standalone*nya. Pengujian ini menunjukkan bahwa dengan perbaikan yang tepat, sebuah sistem *hybrid* dapat secara efektif melampaui kinerja model *standalone* tertentu, dalam hal ini adalah Tesseract OCR.

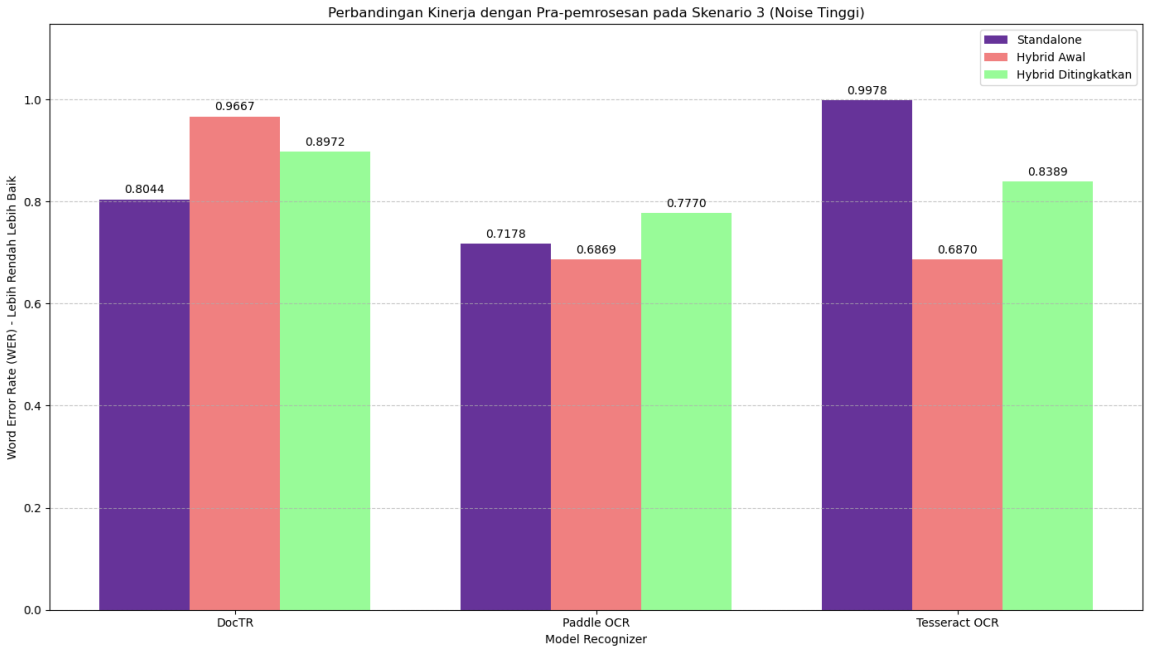
4.1.9 Hasil Uji Coba OCR *Hybrid* yang Ditingkatkan dengan Pra-pemrosesan (Detektor Easy OCR)

Tabel 4.9 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* Awal vs. Ditingkatkan (Detektor Easy OCR)

Model *Recognizer*	**Skenario**	**WER (*Standalone*)**	**WER (*Hybrid* Awal)**	**WER (*Hybrid* Ditingkatkan)**
DocTR	Original	0,5796	0,9148	0,7871
	Skenario 1	0,5976	0,9275	0,8071
	Skenario 2	0,7382	0,9447	0,9429
	Skenario 3	0,8044	0,9667	0,8972
Paddle OCR	Original	0,5181	0,3861	0,6043
	Skenario 1	0,5132	0,4277	0,6857
	Skenario 2	0,5908	0,5664	0,7934
	Skenario 3	0,7178	0,6869	0,7770
Tesseract OCR	Original	0,5447	0,2995	0,5623
	Skenario 1	0,6653	0,3242	0,5947
	Skenario 2	0,9197	0,5032	0,7998
	Skenario 3	0,9978	0,6870	0,8389



Gambar 4.13 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario Original (Detektor Easy OCR)



Gambar 4.14 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario *Noise* Tinggi (Detektor Easy OCR)

Pada tahap pengujian terakhir ini, kita menginvestigasi sebuah skenario yang menarik. Mengingat kombinasi *hybrid* yang menggunakan detektor EasyOCR sebelumnya terbukti sangat sukses, langkah pra-pemrosesan yang sama diterapkan untuk melihat apakah kinerjanya bisa ditingkatkan lebih jauh lagi. Hasil dari eksperimen ini, yang disajikan pada Tabel 4.9 dan divisualisasikan pada Gambar 4.13 dan 4.14, menunjukkan sebuah temuan yang sepenuhnya kontra-intuitif dan sangat penting. Visualisasi pada Gambar 4.13 untuk Skenario Original secara jelas menunjukkan bahwa langkah pra-pemrosesan yang sebelumnya berhasil memperbaiki kombinasi lain, justru merusak kombinasi yang sudah berhasil ini.

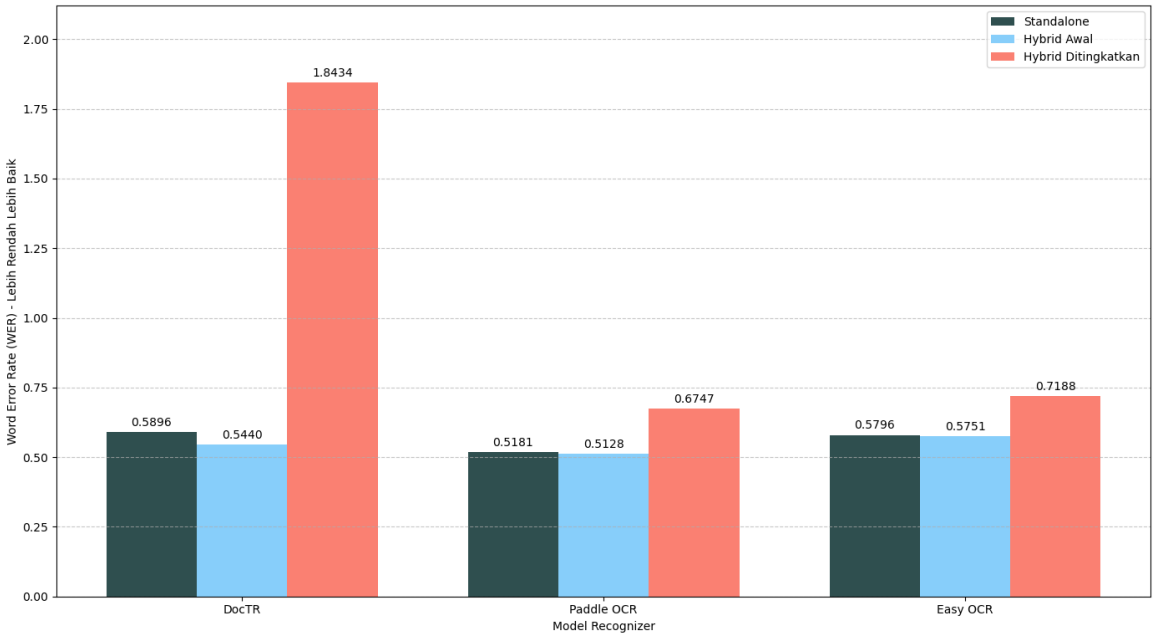
Untuk *recognizer* DocTR, polanya mirip seperti sebelumnya. *Hybrid Awal* (batang merah muda) lebih buruk dari *Standalone* (batang ungu), dan pra-pemrosesan memberikan sedikit perbaikan (batang hijau muda), meskipun masih belum bisa mengalahkan kinerja *standalone*. Terdapat temuan yang cukup unik yang terjadi pada Paddle OCR dan Tesseract OCR. Pada Skenario Original, kombinasi *Hybrid Awal* (batang merah muda) adalah yang paling unggul di mana untuk Tesseract, WER-nya adalah 0,2995, jauh lebih baik dari *standalone* (WER sebesar 0,5447). Namun, setelah pra-pemrosesan diterapkan, kinerjanya justru jauh lebih memburuk, dengan WER meningkat kembali menjadi 0,5623 (batang hijau muda). Hal yang sama terjadi pada Paddle OCR, di mana kinerja unggul dari *Hybrid Awal* (WER sebesar 0,3861) merosot setelah pra-pemrosesan (WER menjadi 0,6043). Ini membuktikan bahwa pra-pemrosesan bukanlah solusi yang bisa diterapkan di semua kondisi. Keberhasilan dari kombinasi detektor EasyOCR dengan *recognizer* Tesseract atau Paddle OCR tampaknya bergantung pada sebuah kecocokan yang sangat spesifik.

4.1.10 Hasil Uji Coba OCR *Hybrid* yang Ditingkatkan dengan Pra-pemrosesan (Detektor Tesseract OCR)

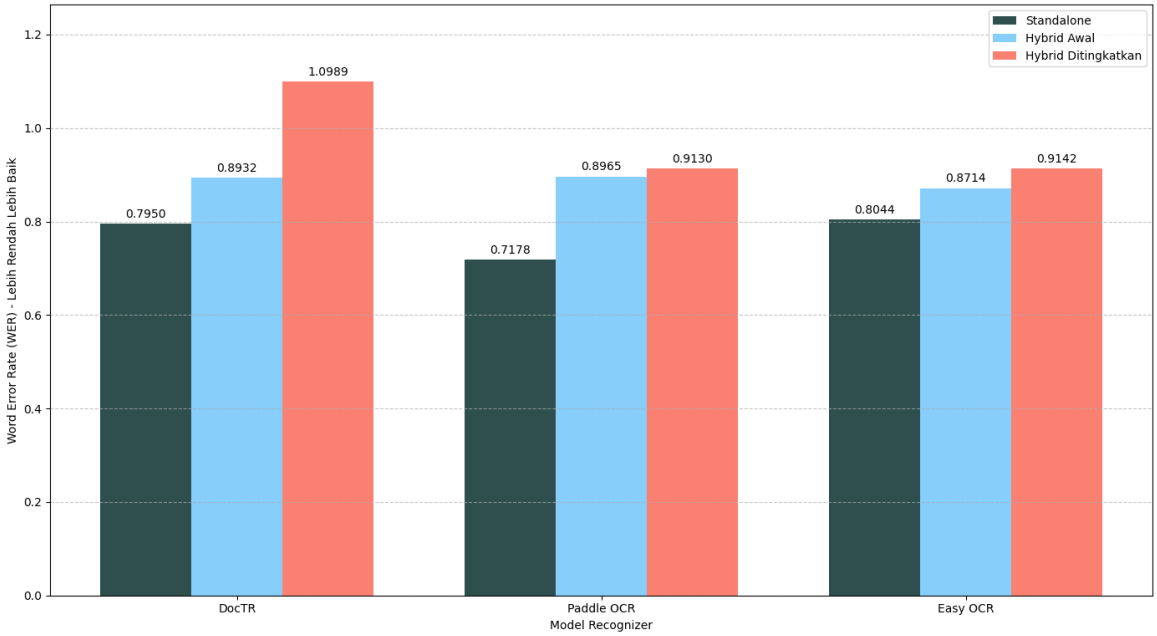
Selanjutnya, mari kita bahas terkait hasil dari skenario yang menggunakan Detektor dari Tesseract OCR.

Tabel 4.10 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* Awal vs. Ditingkatkan (Detektor Tesseract OCR)

Model *Recognizer*	**Skenario**	**WER (*Standalone*)**	**WER (*Hybrid* Awal)**	**WER (*Hybrid* Ditingkatkan)**
DocTR	Original	0,5896	0,5440	1,8434
	Skenario 1	0,6187	0,6037	1,6348
	Skenario 2	0,6978	0,8576	1,2607
	Skenario 3	0,7950	0,8932	1,0989
Paddle OCR	Original	0,5181	0,5128	0,6747
	Skenario 1	0,5132	0,5995	0,7136
	Skenario 2	0,5908	0,8481	0,8939
	Skenario 3	0,7178	0,8965	0,9130
Easy OCR	Original	0,5796	0,5751	0,7188
	Skenario 1	0,5976	0,6065	0,7568
	Skenario 2	0,7382	0,8299	0,9004
	Skenario 3	0,8044	0,8714	0,9142



Gambar 4.15 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario Original (Detektor Tesseract OCR)



Gambar 4.16 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario *Noise* Tinggi (Detektor Tesseract OCR)

Pada bagian ini, investigasi dilanjutkan dengan menerapkan langkah pra-pemrosesan pada kombinasi *hybrid* yang menggunakan detektor dari Tesseract OCR. Sebelumnya, kombinasi ini menunjukkan sedikit keberhasilan pada data bersih. Eksperimen ini bertujuan untuk melihat apakah pra-pemrosesan dapat meningkatkan keberhasilan tersebut. Hasil yang disajikan pada Tabel 4.10 dan divisualisasikan pada Gambar 4.15 dan 4.16 menunjukkan temuan yang menarik dan memperkuat kesimpulan hasil evaluasi sebelumnya. Visualisasi pada Gambar 4.15 untuk Skenario Original secara menampilkan kegagalan luar biasa ini.

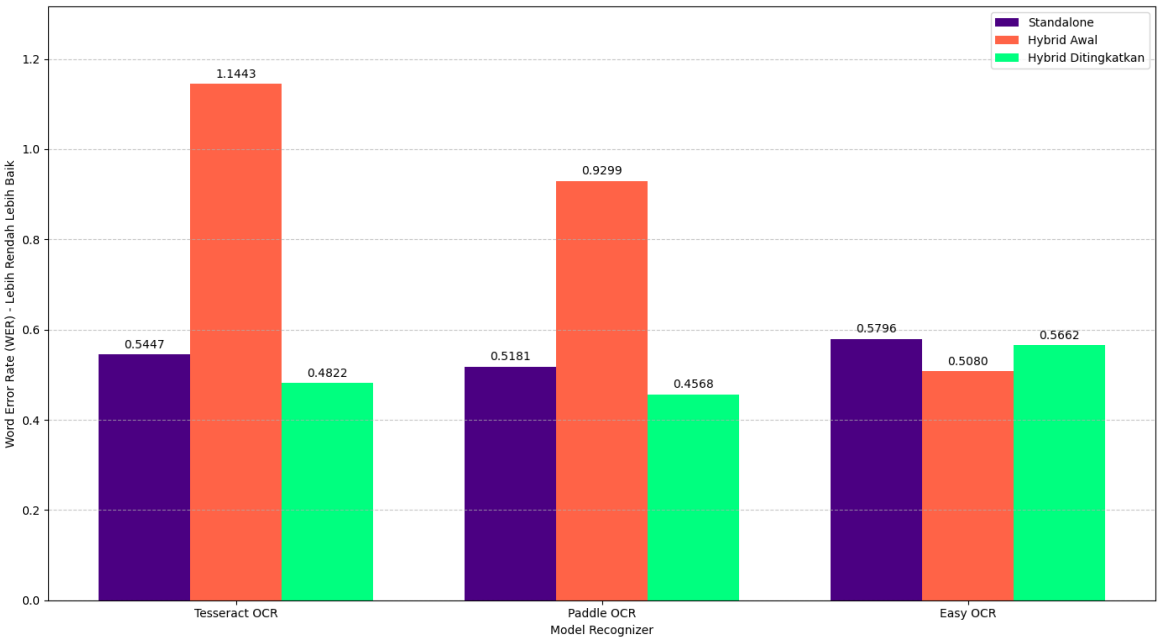
Untuk *recognizer* DocTR, kombinasi *Hybrid Awal* (batang biru muda) dengan WER 0.5440 sedikit lebih unggul dari *Standalone* (batang hitam) dengan WER 0.5896. Namun, setelah pra-pemrosesan diterapkan, kinerjanya merosot, dengan tingkat kesalahan (batang merah muda) meningkat lebih dari tiga kali lipat menjadi 1.8434. Pola yang sama, meskipun tidak seburuk DocTR, juga terjadi pada Paddle OCR dan Easy OCR. Kinerja *Hybrid Awal* yang sedikit lebih baik atau sebanding dengan *standalone* menjadi jauh lebih buruk setelah melalui pra-pemrosesan. Untuk Easy OCR, misalnya, WER meningkat dari 0.5751 (*Hybrid Awal*) menjadi 0.7188 (*Hybrid Ditingkatkan*). Keberhasilan awal dari kombinasi detektor Tesseract bergantung pada sebuah kecocokan.

4.1.11 Hasil Uji Coba OCR *Hybrid* yang Ditingkatkan dengan Pra-pemrosesan (Detektor DocTR)

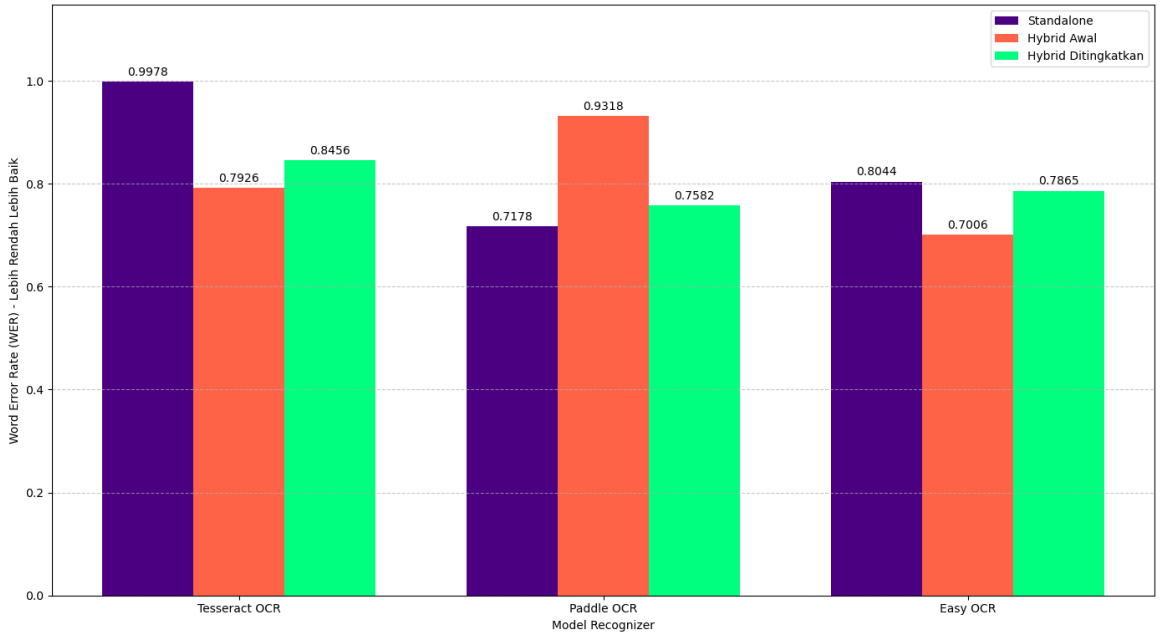
Selanjutnya, mari kita bahas terkait hasil dari skenario yang menggunakan Detektor dari DocTR.

Tabel 4.11 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* Awal vs. Ditingkatkan (Detektor DocTR)

Model *Recognizer*	**Skenario**	**WER (*Standalone*)**	**WER (*Hybrid* Awal)**	**WER (*Hybrid* Ditingkatkan)**
Tesseract OCR	Original	0,5447	1,1443	0,4822
	Skenario 1	0,6653	1,1194	0,4677
	Skenario 2	0,9197	0,7854	0,6893
	Skenario 3	0,9978	0,7926	0,8456
Paddle OCR	Original	0,5181	0,9299	0,4568
	Skenario 1	0,5132	0,9167	0,5982
	Skenario 2	0,5908	0,9223	0,6892
	Skenario 3	0,7178	0,9318	0,7582
Easy OCR	Original	0,5796	0,5080	0,5662
	Skenario 1	0,5976	0,5129	0,5876
	Skenario 2	0,7382	0,6410	0,7212
	Skenario 3	0,8044	0,7006	0,7865



Gambar 4.17 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario Original (Detektor DocTR)



Gambar 4.18 Perbandingan Kinerja OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* Ditingkatkan pada Skenario *Noise* Tinggi (Detektor DocTR)

Sebagai penutup dari seluruh evaluasi OCR *Hybrid*, bagian ini menerapkan langkah pra-pemrosesan pada kombinasi yang menggunakan detektor modern dari DocTR. Sebelumnya, kombinasi ini menunjukkan hasil yang menarik di mana kombinasi ini mengalami kegagalan saat dikombinasikan dengan *recognizer* kuat/modern, namun berhasil dengan baik bersinergi dengan *recognizer* yang lebih lemah/sederhana. Visualisasi pada Gambar 4.17 untuk Skenario Original secara efektif menampilkan sebuah terobosan yang menarik untuk dibahas. Untuk *recognizer* Tesseract OCR dan Paddle OCR, pra-pemrosesan berhasil mengubah kegagalan

menjadi keberhasilan. Dapat diamati pada Gambar 4.17 bahwa kinerja *Hybrid Awal* untuk Tesseract (batang merah) sangat buruk dengan WER 1.1443.

Namun, setelah pra-pemrosesan, kinerjanya (batang hijau) tidak hanya membaik tetapi juga menjadi yang terbaik, dengan WER turun drastis menjadi 0.4822, bahkan mengalahkan kinerja *standalone*-nya (batang ungu) yang memiliki WER 0.5447. Pola yang sama terjadi pada Paddle OCR, di mana WER turun dari 0.9299 menjadi 0.4568, menjadikannya kombinasi OCR *Hybrid* dengan akurasi tertinggi secara keseluruhan. Di sisi lain, sebuah keanehan kembali terjadi pada *recognizer* Easy OCR. Kombinasi *Hybrid Awal* yang sebelumnya sudah berhasil (WER 0.5080) justru kinerjanya sedikit memburuk setelah pra-pemrosesan (WER menjadi 0.5662). Dari pengujian yang telah dilaksanakan, pra-pemrosesan terbukti mampu membukakan potensi dari detektor modern seperti DocTR, memungkinkannya untuk bekerja dengan baik dengan *recognizer* lain dan menghasilkan akurasi yang menyaingi model OCR *standalone*. Namun, sekali lagi akan ditegaskan bahwa tidak ada solusi yang sempurna yang mampu menyediakan satu solusi untuk semua jenis skenario yang ada.

4.2 Pembahasan

Pada bagian ini, data hasil penelitian yang telah disajikan di Bab 4.1 akan dianalisis lebih dalam. Tujuan dari pembahasan ini adalah untuk menjawab pertanyaan di balik angka-angka yang muncul, menghubungkan temuan dengan dasar teori, dan membangun sebuah argumen yang utuh mengenai strategi optimasi OCR yang paling efektif.

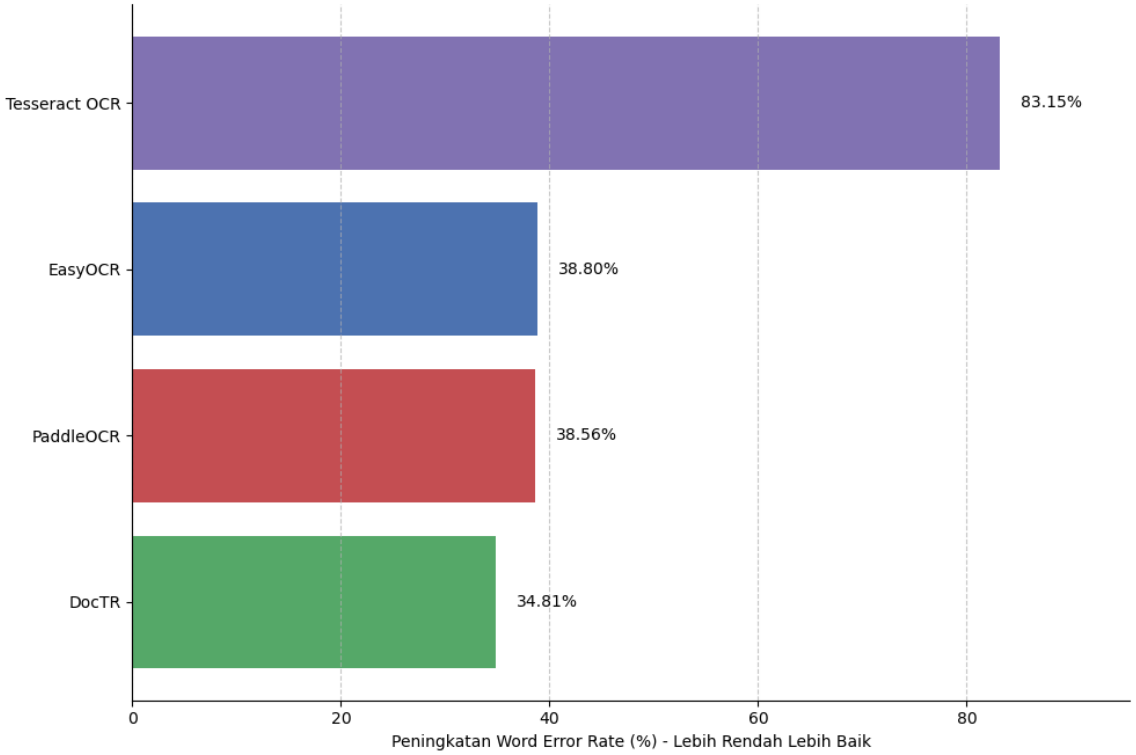
4.2.1 Analisis Kinerja dan Karakteristik Model *Standalone*

Gambar 4.19 Perbandingan WER dan Waktu Pemrosesan Model *Standalone* pada Skenario Bersih

Gambar 4.19 di atas membandingkan rata-rata waktu yang dibutuhkan untuk memproses tiap gambar dan *Word Error Rate* (WER) pada tiap gambar. Tujuannya adalah untuk melihat kemampuan setiap model *standalone* saat diuji pada dokumen yang bersih tanpa gangguan. Dari gambar ini, kita bisa melihat adanya dua temuan utama yang menyoroti temuan tentang akurasi dan kecepatan. Gambar di sebelah kiri menunjukkan tingkat kesalahan setiap model OCR *Standalone* dalam membaca kata (WER). Semakin rendah nilainya berarti semakin bagus atau akurat kemampuan tiap model dalam membaca kata. Dari gambar tersebut, terlihat jelas bahwa Paddle OCR adalah yang paling unggul dengan batangnya yang paling pendek, yang

berarti tingkat kesalahannya paling rendah (0.5181) Sehingga jika kita hanya membutuh hasil yang paling akurat, Paddle OCR adalah jawabannya.

Namun, terdapat perspektif lain yang sangat berbeda jika kita melihat gambar di sebelah kanan yang menunjukkan kecepatan pemrosesan setiap model. Di kasus ini, Tesseract OCR terbukti sebagai yang paling cepat. Model ini hanya butuh waktu kurang dari satu detik (0,7963 detik) untuk memproses satu gambar. Sebaliknya, Paddle OCR, yang paling akurat dari segi WER tadi, justru menjadi yang paling lambat. Model ini membutuhkan waktu hingga 16,6 detik, atau lebih dari 20 kali lebih lama dibandingkan Tesseract. Ketika kita melihat kedua gambar ini bersamaan, didapati bahwa model yang paling akurat ternyata yang paling lambat, dan yang paling cepat akurasinya tidak sebagus yang lain. Tidak ada satu pun model yang unggul di kedua metrik. Temuan ini menandakan bahwa bahkan pada kondisi dokumen yang bersih tanpa gangguan, terdapat *trade-off* dari kinerja model OCR yang telah dievaluasi. Pilihan sulit inilah yang menjadi alasan kuat untuk kita perlu mencari cara lain untuk mendapatkan hasil terbaik, dan menjadi dasar untuk pengujian selanjutnya pada skenario yang diimbuhi dengan *noise* pada beberapa tingkat yang berbeda.



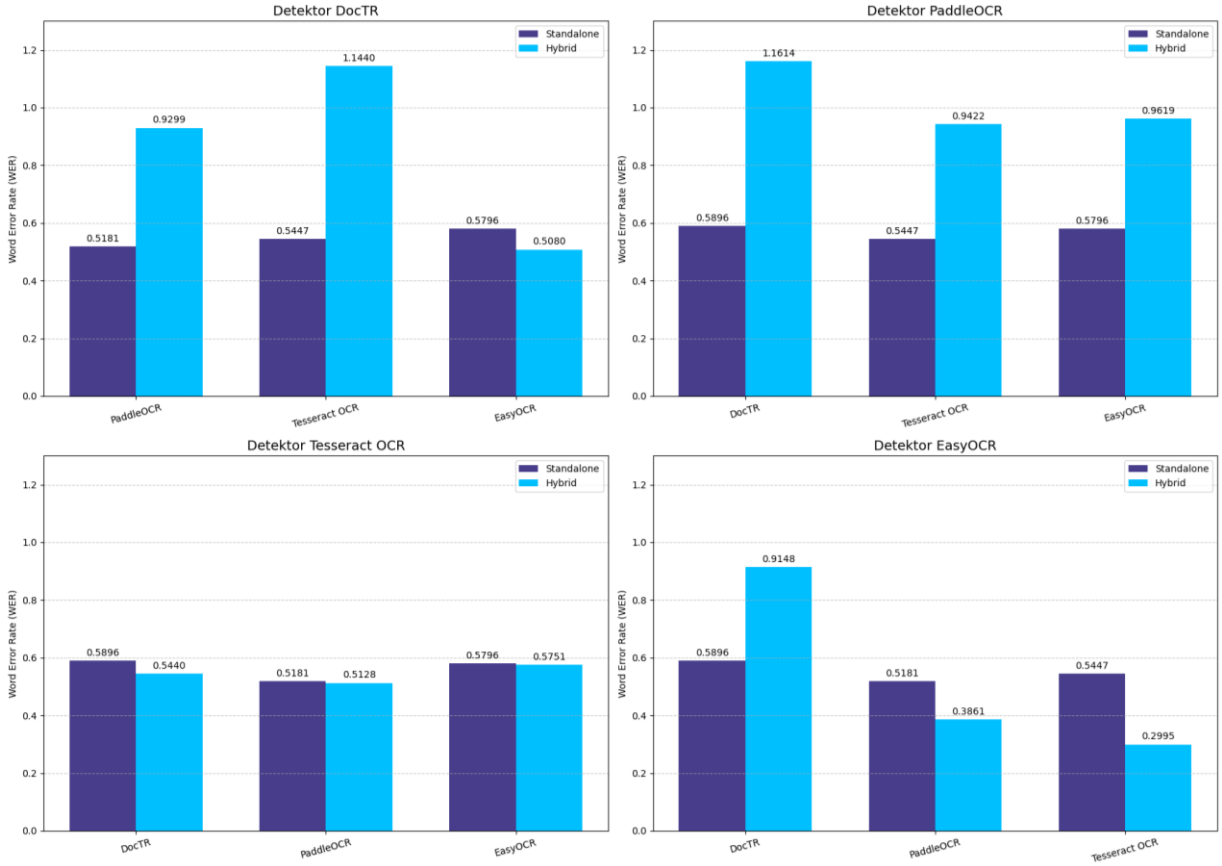
Gambar 4.20 Peningkatan *Word Error Rate* (WER) Model OCR *Standalone* dari Skenario Original ke Skenario *Noise* Tinggi

Selanjutnya kita akan membahas tentang peningkatan *Word Error Rate* (WER) dari Skenario original ke Skenario *Noise* Tinggi yang dapat kita lihat pada Gambar 4.20. Visualisasi ini secara langsung menjawab tentang penurunan performa dari tiap model OCR *Standalone* yang dihadapkan pada dokumen yang diberi *noise* pada tingkat tertentu. Diagram di atas mengukur tingkat penurunan performa dalam bentuk persentase kenaikan *Word Error Rate* (WER). Batang yang lebih panjang berarti model tersebut lebih sensitif atau rapuh terhadap *noise*.

Dari gambar tersebut, terlihat sangat jelas bahwa Tesseract OCR memiliki batang yang paling panjang, dengan tingkat kesalahan yang melonjak hingga 83.15%. Ini adalah bukti bahwa Tesseract OCR adalah model yang paling sensitif terhadap *noise*. Meskipun sangat cepat pada data bersih, kemampuannya menurun drastis begitu kualitas gambar memburuk. Hal ini menjadikannya pilihan yang sangat berisiko untuk digunakan pada dokumen hasil *scan* yang tidak sempurna. Di sisi lain, DocTR menunjukkan batang yang paling pendek, dengan kenaikan tingkat kesalahan hanya sebesar 34.81%. Ini membuktikan bahwa DocTR adalah model yang paling tahan uji atau paling kuat (*robust*) di antara keempat model yang diuji. Kemampuannya untuk mempertahankan tingkat akurasi yang stabil meskipun kualitas gambar menurun drastis menjadikannya pilihan yang paling dapat diandalkan untuk kondisi dokumen di dunia nyata yang penuh dengan gangguan.

Di satu sisi, Model Easy OCR dan Paddle OCR berada di posisi Tengah dari segi penurunan performa. Tingkat kesalahan keduanya meningkat sekitar 38%, yang menunjukkan bahwa mereka memiliki ketahanan yang menengah. Mereka tidak separah Tesseract, namun juga tidak sebagus DocTR. Jika Gambar 4.19 menunjukkan adanya *trade-off* antara akurasi dan kecepatan, Gambar 4.20 ini memperkenalkan faktor yang tidak kalah penting, yaitu ketahanan (*robustness*). Kinerja model OCR *Standalone* tidak hanya diukur dari segi akurasi dan kecepatan, tetapi juga dari segi ketahanan terhadap *noise*. Model yang paling akurat di awal (Paddle OCR) bukanlah yang paling tahan banting, model yang paling tahan banting (DocTR) bukanlah yang tercepat, dan model yang tercepat (Tesseract OCR) adalah yang paling rapuh.

4.2.2 Analisis Kinerja OCR *Hybrid*

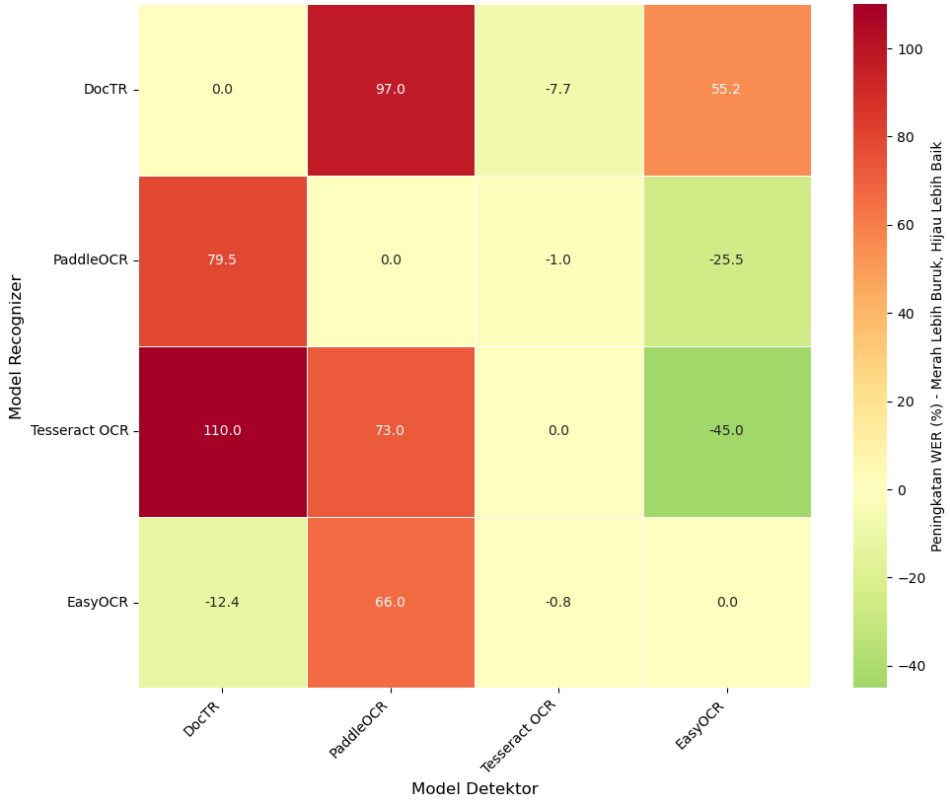


Gambar 4.21 Perbandingan Kinerja Model OCR *Standalone* vs. OCR *Hybrid* pada Skenario Original

Gambar 4.21 di atas menyajikan rangkuman lengkap dari seluruh pengujian OCR *Hybrid* pada kondisi dokumen yang bersih (Skenario Original). Visualisasi ini fokus pada satu model detektor dan menunjukkan bagaimana kinerjanya saat dipasangkan dengan *recognizer* dari model lain. Batang berwarna ungu mewakili kinerja asli (*standalone*), sementara batang biru muda mewakili kinerja setelah digabung (*hybrid*). Tujuan kita adalah untuk melihat batang biru bisa lebih pendek dari batang ungu, yang berarti kinerjanya membaik. Mari kita mulai dari visualisasi di bagian kiri atas (Detektor DocTR) yang menunjukkan hasil yang beragam. Saat detektor dari DocTR dipasangkan dengan *recognizer* Paddle OCR dan Tesseract OCR, kinerjanya memburuk. Batang biru untuk keduanya jauh lebih tinggi daripada batang ungu, dengan tingkat kesalahan (WER) Tesseract melonjak dari 0.5447 menjadi 1.1440. Namun, sebuah keberhasilan kecil terlihat saat dipasangkan dengan Easy OCR, di mana tingkat kesalahannya sedikit menurun dari 0.5796 menjadi 0.5080.

Di sisi lain, pada bagian kanan atas dari visualisasi di atas (Detektor Paddle OCR), menunjukkan penurunan kinerja yang drastis sehingga dapat dianggap gagal total. Dapat dikatakan gagal karena pada saat detektor dari Paddle OCR digunakan, semua kombinasi OCR *Hybrid* menghasilkan kinerja yang jauh lebih buruk. Batang biru untuk semua *recognizer* (DocTR, Tesseract, dan Easy OCR) menjulang tinggi, menunjukkan bahwa detektor ini sangat tidak cocok jika dipisahkan dari *recognizer* aslinya. Selanjutnya, pada bagian kiri bawah pada visualisasi di atas (Detektor Tesseract OCR), terdapat sedikit peningkatan yang dapat dianggap sebagai peningkatan yang konsisten meski tidak signifikan. Untuk semua *recognizer* (DocTR, Paddle OCR, dan Easy OCR), penggunaan detektor Tesseract OCR berhasil menurunkan tingkat kesalahan, meskipun sangat tipis. Batang biru selalu sedikit lebih pendek dari batang ungu. Ini menunjukkan bahwa detektor Tesseract OCR, meskipun sederhana, cukup cocok oleh *recognizer* lain pada kondisi data bersih.

Di satu sisi, pada bagian kanan bawah visualisasi di atas (Detektor Easy OCR), terdapat hasil yang paling menonjol. Di satu sisi, terdapat kegagalan saat dipasangkan dengan *recognizer* DocTR (WER naik dari 0.5896 menjadi 0.9148). Namun, di sisi lain, terjadi peningkatan yang signifikan saat dipasangkan dengan *recognizer* Paddle OCR dan Tesseract OCR. Kombinasi dengan Tesseract OCR adalah yang paling menarik di antara kombinasi yang lainnya, di mana tingkat kesalahannya berhasil berkurang hampir setengahnya, dari 0.5447 menjadi hanya 0.2995. Secara keseluruhan, visualisasi ini membantah perihal penggabungan model yang akan selalu memberikan hasil lebih baik yang pada kenyataan pendekatan ini terbukti sangat tidak bisa diprediksi. Kegagalan pada detektor Paddle OCR menunjukkan pentingnya kecocokan dalam sebuah model terhadap pasangannya. Namun, keberhasilan pada detektor EasyOCR membuktikan adanya kecocokan yang tidak terduga, di mana detektor yang lebih sederhana justru bisa menjadi pasangan terbaik untuk *recognizer* tertentu.



Gambar 4.22 Heatmap Kinerja OCR *Hybrid* pada Skenario Original

Setelah melihat kinerja dari setiap kombinasi pada Gambar 4.21, visualisasi pada Gambar 4.22 di atas akan membantu kita dalam merangkum peningkatan atau penurunan kinerja pada masing-masing kombinasi OCR *Hybrid* pada Skenario Original. *Heatmap* ini tidak menunjukkan tingkat kesalahan secara langsung, melainkan seberapa besar kinerja sebuah kombinasi memburuk atau membaik dibandingkan saat model tersebut bekerja sendiri (*standalone*).

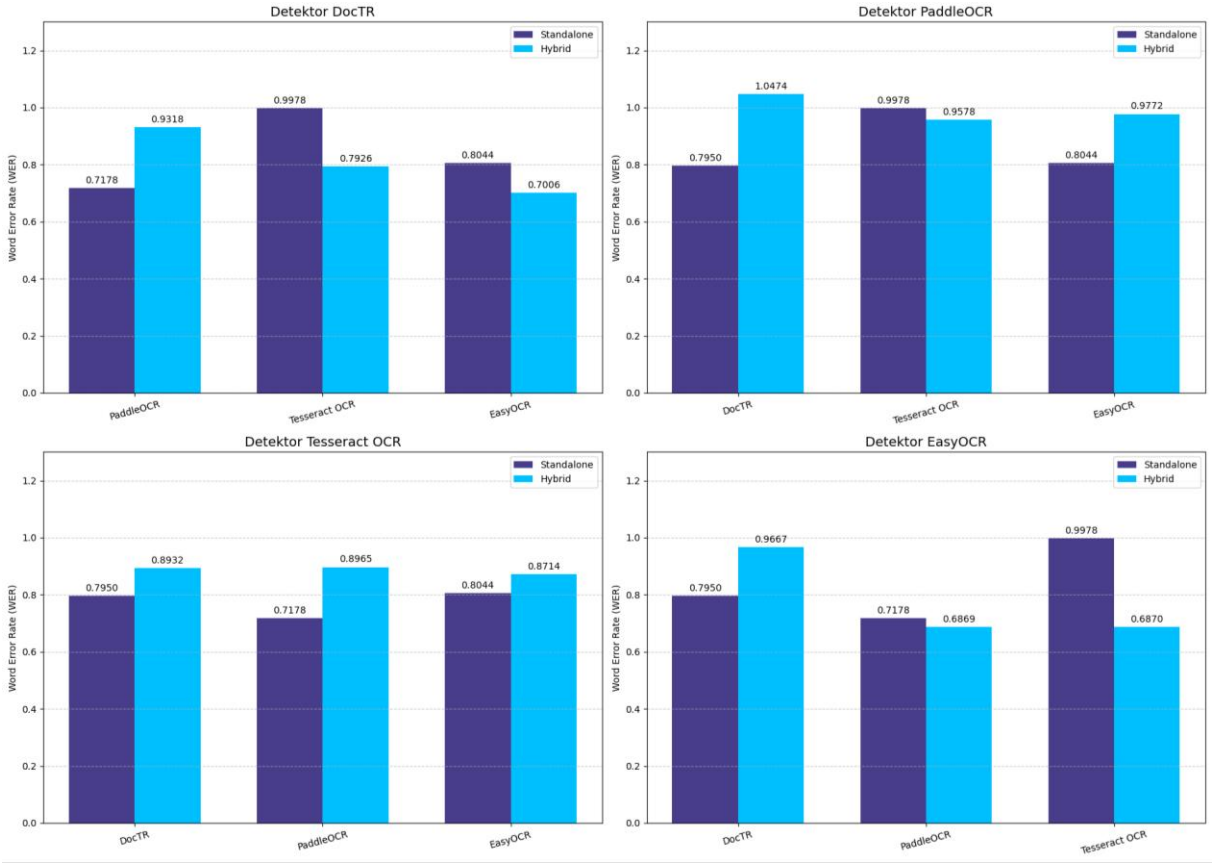
Infografis dari visualisasi di atas dapat dipahami sebagai berikut.

- Warna Merah : Menandakan kinerja memburuk. Semakin gelap merahnya, semakin parah penurunannya.
- Warna Hijau : Menandakan kinerja membaik. Semakin gelap hijaunya, semakin bagus peningkatannya.
- Angka di dalam kotak : Menunjukkan persentase perubahan tersebut.

Hal pertama yang paling mencolok adalah dominasi warna merah di sebagian besar gambar visualisasi. Ini adalah bukti yang paling kuat bahwa pendekatan OCR *Hybrid* secara menyeluruh mengalami kegagalan dalam meningkatkan performa terutama dalam segi akurasi. Sebagian besar kombinasi justru membuat tingkat *error* menjadi lebih tinggi. Contoh paling menonjol adalah saat Detektor DocTR dipasangkan dengan *Recognizer* Tesseract OCR, di mana tingkat kesalahan melonjak hingga 110%, yang berarti kinerjanya menjadi dua kali lipat lebih buruk. Namun, di tengah visualisasi yang Sebagian besar berwarna merah pertanda kemerosotan kinerja OCR *Hybrid*, kita juga mendapati beberapa kotak berwarna hijau yang menjadi sesuatu menarik untuk dibahas. Kotak paling hijau adalah saat Detektor Easy OCR

dipasangkan dengan *Recognizer* Tesseract OCR. Angka -45.0% di dalamnya berarti kombinasi ini berhasil menurunkan tingkat kesalahan Tesseract OCR hingga hampir setengahnya. Ini adalah kombinasi terbaik yang ditemukan dalam pengujian ini. Peningkatan signifikan juga terlihat saat Detektor Easy OCR dipasangkan dengan *Recognizer* Paddle OCR dengan penurunan *error* mencapai 25.5%.

Dapat kita simpulkan bahwa melalui visualisasi ini, kita mengetahui kombinasi apa saja yang menjadi kombinasi paling menonjol secara keseluruhan. Yang pertama, bagian Detektor DocTR dan Detektor Paddle OCR didominasi warna merah, yang menunjukkan keduanya paling sulit untuk dipasangkan dengan *recognizer* lain. Terdapat juga temuan menarik sekaligus unik yang berada pada bagian Detektor Easy OCR memiliki visualisasi yang paling berwarna-warni (campuran hijau dan merah), menunjukkan bahwa detektor sederhana ini justru paling fleksibel dan mampu menciptakan kecocokan terbaik dengan *recognizer* tertentu. Secara ringkas, visualisasi ini merangkum seluruh hasil dari evaluasi OCR *Hybrid*, sebuah pendekatan yang sebagian besar gagal, namun di dalamnya juga terdapat beberapa kecocokan yang tidak terduga. Ini secara menjawab rumusan masalah kedua bahwa pendekatan *hybrid* bukanlah strategi yang efektif secara umum, tetapi memiliki potensi jika dan hanya jika pasangan yang tepat ditemukan.

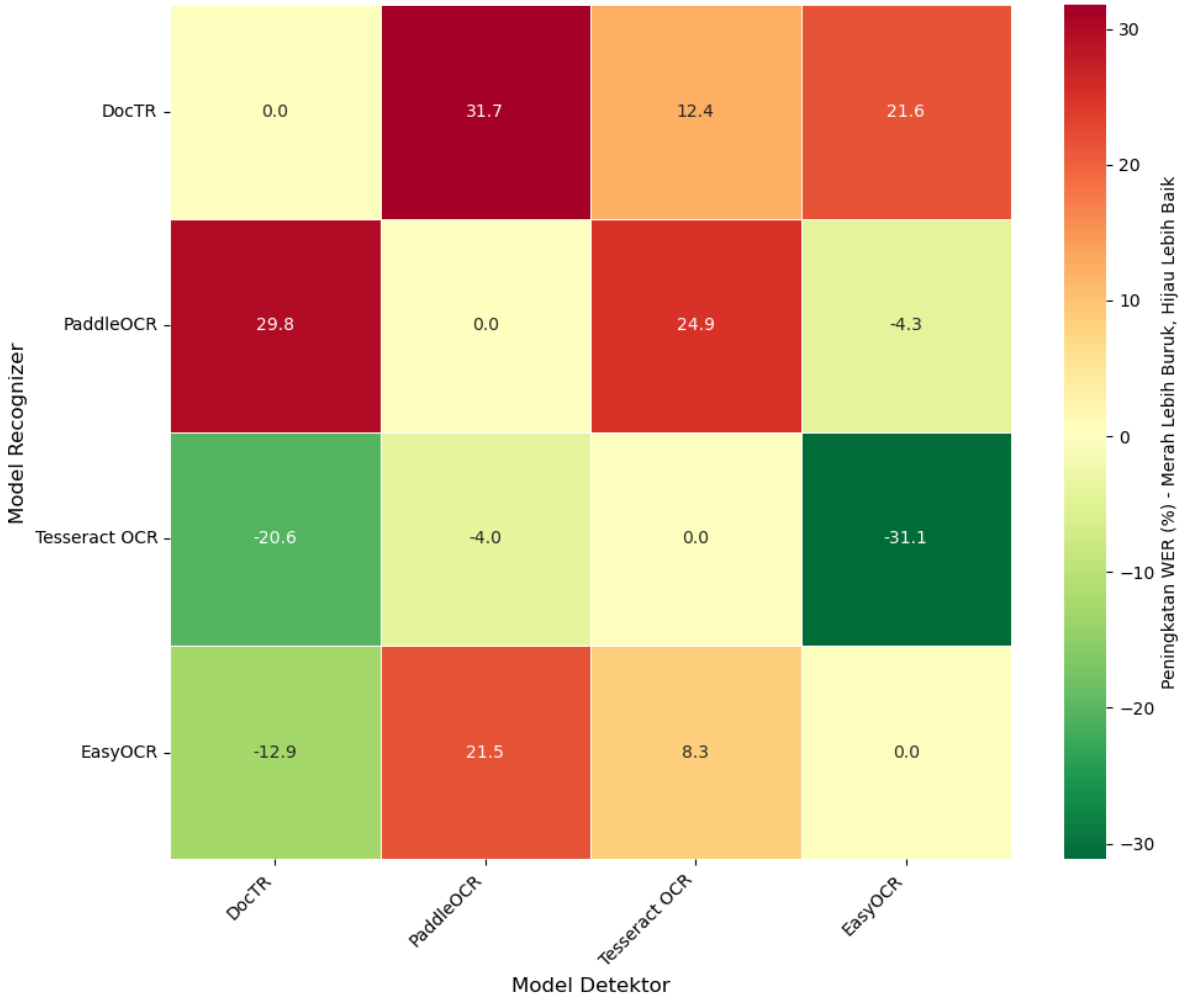


Gambar 4.23 Perbandingan Kinerja Model OCR *Standalone* vs. OCR *Hybrid* pada Skenario *Noise* Tinggi

Gambar 4.23 di atas menyajikan hasil uji ketahanan tiap kombinasi OCR *Hybrid* dengan memberikan stress menggunakan dataset dari Skenario *Noise* Tinggi. Visualisasi ini akan menunjukkan pola keberhasilan dan kegagalan yang kita lihat pada data bersih, tetap bertahan atau justru berubah saat kualitas gambar menurun drastis.

Dimulai dari bagian kiri atas dari visualisasi di atas (Detektor DocTR) di mana terdapat temuan yang menunjukkan peningkatan yang signifikan di mana Detektor DocTR mampu mendorong kinerja *recognizer* yang lebih lemah seperti Tesseract OCR dan Easy OCR. Untuk Tesseract OCR, yang kinerja aslinya (*standalone*) memiliki performa sangat buruk dengan tingkat kesalahan (WER) 0.9978, pada kombinasi OCR *Hybrid* kali ini berhasil menekan tingkat kesalahannya secara drastis menjadi 0.7926. Hal yang sama terjadi pada Easy OCR, di mana kinerjanya membaik dari 0.8044 menjadi 0.7006. Ini membuktikan bahwa detektor yang kuat dapat memberikan input lokasi teks (*bounding box*) yang jauh lebih baik pada gambar yang rusak, sehingga membantu *recognizer* lain untuk bekerja lebih efektif.Selanjutnya, pada bagian kanan atas dan kiri bawah dari visualisasi di atas (Detektor Paddle OCR dan Detektor Tesseract OCR) terdapat pola penurunan yang membuat keduanya mengalami kegagalan. Baik detektor dari Paddle OCR maupun Tesseract OCR tidak mampu menangani *noise* dengan baik. Akibatnya, semua kombinasi OCR *Hybrid* yang menggunakan kedua detektor ini menghasilkan kinerja yang lebih buruk daripada kinerja model *standalone*-nya. Batang biru muda selalu lebih tinggi daripada batang ungu tua. Ini menegaskan bahwa jika detektor sebagai fondasi awal sudah gagal menemukan teks dengan benar, maka *recognizer* secanggih apa pun tidak akan bisa memperbaikinya.

Yang terakhir, pada bagian kanan bawah visualisasi di atas (Detektor Easy OCR), terdapat sebuah temuan yang menarik. Mirip dengan DocTR, detektor sederhana dari Easy OCR juga menunjukkan untuk mendorong kinerja *recognizer* pasangannya. Seperti kombinasi dengan Tesseract OCR, kombinasi ini kembali menjadi yang memiliki hasil yang memuaskan, dengan tingkat *error* turun dari 0.9978 menjadi 0.6870. Selain itu juga terdapat kombinasi dengan Paddle OCR yang juga menunjukkan sedikit peningkatan dengan WER turun dari 0.7178 menjadi 0.6869. Ini menunjukkan bahwa bahkan detektor yang sederhana pun, jika memiliki kecocokan yang baik, bisa lebih efektif daripada detektor canggih dalam kondisi tertentu. Analisis pada kondisi *noise* tinggi ini memperkuat temuan sebelumnya dan memberikan kesimpulan yang lebih spesifik. Kemampuan sebuah detektor untuk mengatasi *noise* adalah faktor penentu dalam keberhasilan sistem OCR *Hybrid*. Detektor yang kuat (seperti DocTR) atau yang "cocok" (seperti Easy OCR) dapat secara signifikan meningkatkan kinerja *recognizer* lain. Sebaliknya, detektor yang rentan terhadap *noise* (seperti Tesseract OCR dan Paddle OCR) akan selalu menghasilkan sistem *hybrid* yang lebih buruk.

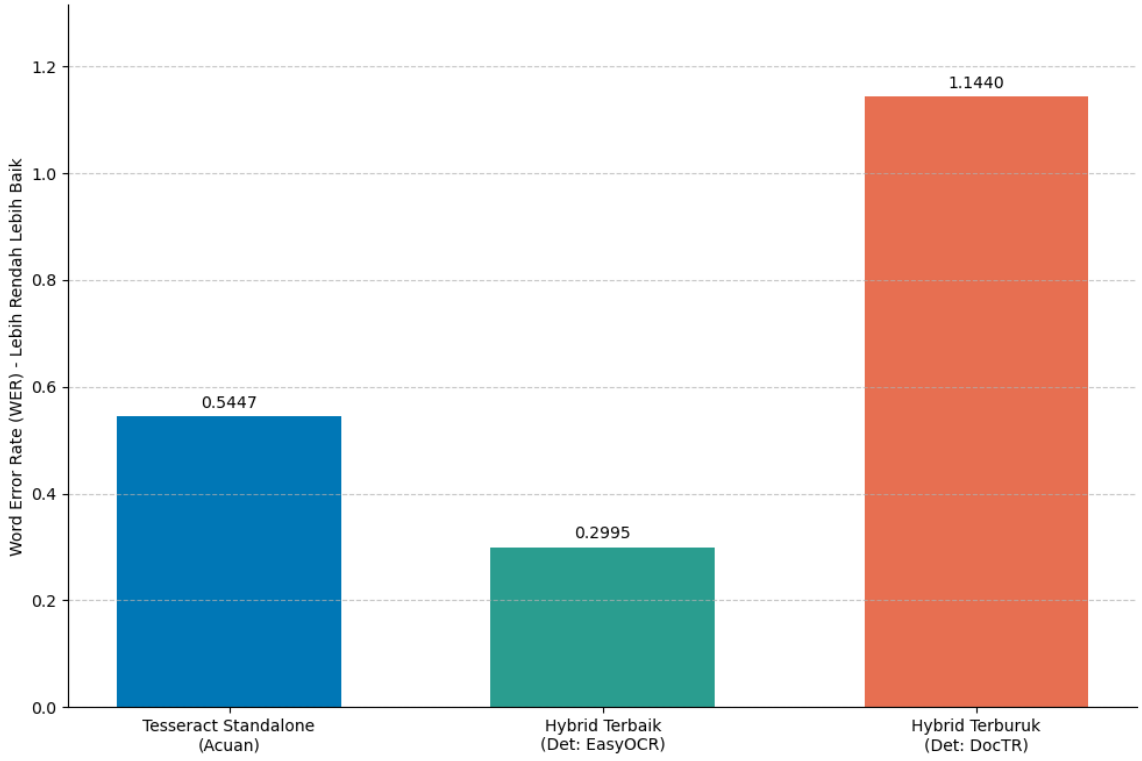


Gambar 4.24 Heatmap Kinerja OCR *Hybrid* pada Skenario *Noise* Tinggi

Gambar 4.24 di atas menyajikan kinerja dari seluruh pengujian OCR *Hybrid* dalam kondisi yang paling menantang. Visualisasi ini menunjukkan besar kinerja setiap kombinasi memburuk (warna merah) atau membaik (warna hijau) saat dihadapkan pada dokumen dengan tingkat *noise* yang tinggi. Dari visualisasi ini, kita dapat melihat temuan yang sangat jelas mengenai faktor apa yang paling menentukan keberhasilan sebuah sistem *hybrid*. Hal pertama yang paling mencolok adalah beberapa sebaran warna hijau pada visualisasi tersebut menandakan bahwa tidak sedikit skenario uji membuahkan hasil yang cukup memberikan peningkatan terhadap performa keseluruhan terutama dalam segi pengurangan tingkat error pada level kata (WER). Mari kita ambil dari hasil yang memiliki warna hijau yang paling mencolok yaitu, kombinasi Detektor Easy OCR dan *recognizer* Tesseract OCR. Hasil yang menunjukkan bahwa kombinasi ini berhasil memangkas lebih dari 30% tingkat kesalahan Tesseract OCR pada versi *standalone*-nya yang memiliki performa jauh lebih buruk dan dapat dianggap gagal dalam melakukan tugasnya. Selain kombinasi ini, juga terdapat kombinasi lain yang tidak kalah bersinar yaitu, kombinasi Detektor DocTR dan *recognizer* Tesseract OCR yang membuahkan hasil sebesar -20,6%. Temuan ini membuktikan fenomena peningkatan di tengah ketidakcocokan OCR *Hybrid* lainnya yang sudah diujikan. Detektor yang mampu mengatasi *noise* dengan baik (seperti DocTR dan Easy OCR) dapat memberikan input lokasi teks yang jauh lebih akurat, sehingga secara efektif dapat mendongkrak kinerja *recognizer* yang rapuh terhadap *noise*.

Di sisi lain, ternyata masih ada beberapa kombinasi gagal yang ditandai dengan warna merah. Visualisasi untuk bagian Detektor Paddle OCR dan Detektor Tesseract OCR didominasi oleh warna merah dan oranye. Detektor Paddle OCR dengan rentan nilai tertingginya berada di kisaran 21,5% hingga 31,7% dan untuk detektor Tesseract OCR dengan rentan nilai tertingginya berada pada kisaran 12,4% hingga 24,9%. Ini menunjukkan bahwa kedua detektor ini tidak memiliki ketahanan yang baik terhadap *noise*. Ketika mereka gagal menemukan teks maupun mengoper informasi yang sesuai kepada *recognizer* pasangannya dengan benar pada gambar yang rusak, kesalahan ini kemudian menyebabkan kinerja *recognizer* pasangannya menjadi lebih buruk. Ini menegaskan bahwa ketahanan detektor adalah dasar yang paling penting di mana jika dasarnya rapuh, keseluruhan sistem akan ikut gagal.

Secara ringkas Detektor DocTR adalah yang paling canggih, sementara Detektor Easy OCR adalah yang paling sederhana. Namun, pada kondisi *noise* tinggi, detektor yang paling sederhana inilah yang justru menciptakan kombinasi terbaik secara keseluruhan. Ini menunjukkan bahwa dalam kondisi yang sulit, kemampuan sebuah *recognizer* untuk memahami output sederhana dari detektor bisa jadi lebih penting daripada kecanggihan detektor itu sendiri. Visualisasi ini menyimpulkan bahwa saat berhadapan dengan dokumen berkualitas buruk, kunci keberhasilan sistem *hybrid* terletak pada kemampuan detektor untuk mengatasi *noise*. Detektor yang kuat dan cocok dapat secara signifikan meningkatkan kinerja, sementara detektor yang rapuh akan selalu menghasilkan kegagalan. Ini memberikan jawaban yang sangat jelas untuk rumusan masalah kedua dalam konteks skenario dunia nyata.



Gambar 4.25 Perbandingan Hasil Evaluasi OCR *Standalone* (Acuan), OCR *Hybrid* Terbaik, dan OCR *Hybrid* Terburuk pada Skenario Original

Gambar 4.25 menyajikan bukti mengenai pentingnya pemilihan pasangan detektor dalam sebuah sistem OCR *hybrid*. Visualisasi ini berfokus pada satu *recognizer*, yaitu Tesseract OCR, dan menunjukkan bagaimana kinerjanya berubah secara drastis saat dipasangkan dengan detektor terbaik dan terburuknya pada Skenario Original. Pada batang paling kiri, berwarna biru tua, menunjukkan kinerja asli dari Tesseract OCR saat bekerja sendiri (*standalone*). Dengan tingkat kesalahan (WER) sebesar 0.5447, ini adalah titik acuan atau *baseline* kita. Kinerja ini cukup baik, namun menjadi standar yang harus kita coba kalahkan dengan pendekatan *hybrid*.

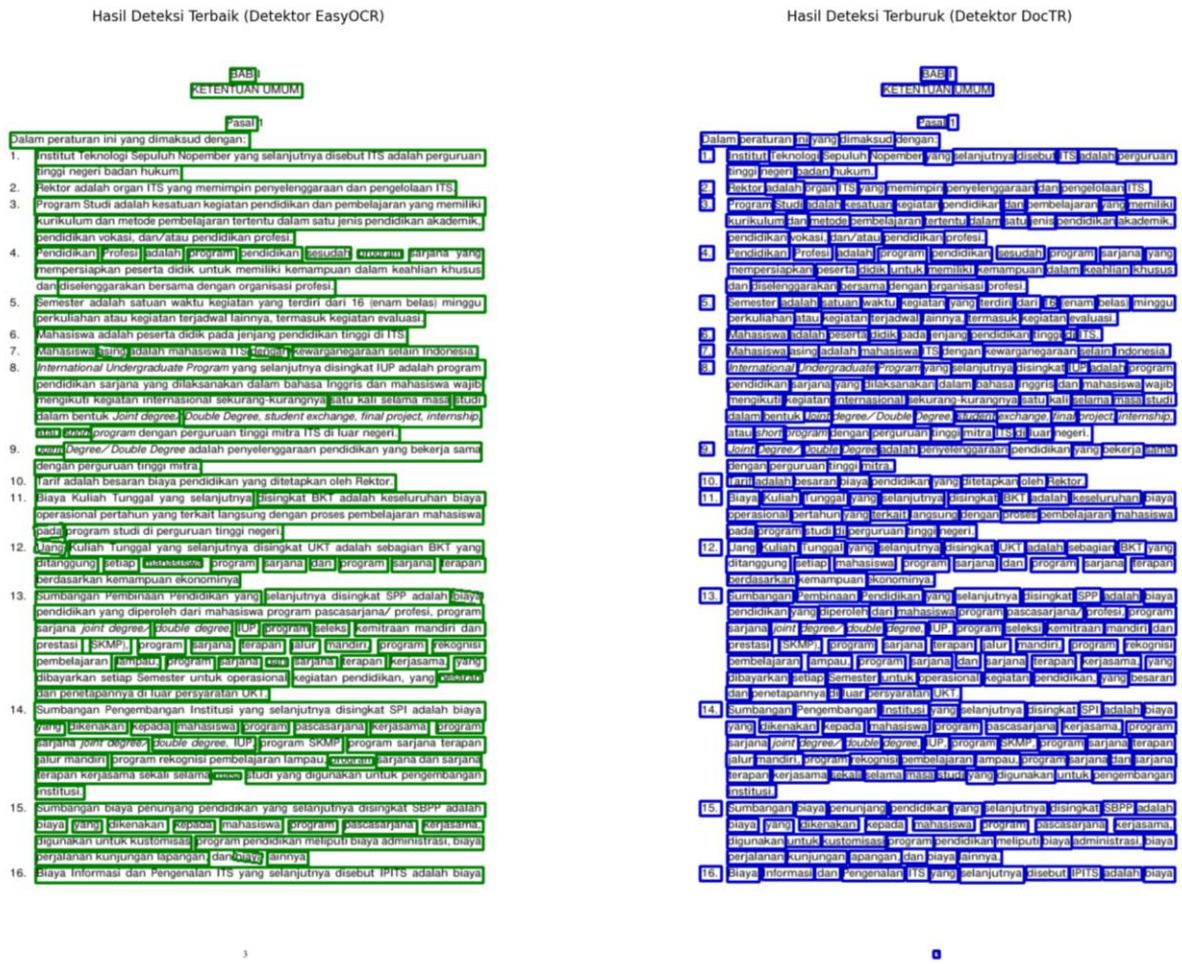
Untuk batang Tengah, berwarna hijau, menunjukkan hasil dari kombinasi *hybrid* terbaik yaitu, Detektor Easy OCR dan *Recognizer* Tesseract OCR. Batang ini jauh lebih pendek, dengan tingkat kesalahan hanya 0.2995. Ini berarti kemampuan Tesseract dalam membaca teks meningkat secara dramatis, dengan tingkat kesalahan yang terpangkas hampir setengahnya. Input lokasi teks yang sesuai dari EasyOCR memungkinkan *recognizer* Tesseract OCR untuk bekerja dengan baik. Selanjutnya, pada batang paling kanan berwarna merah, memuat hasil dari kombinasi OCR *hybrid* terburuk yaitu, Detektor DocTR dan *Recognizer* Tesseract OCR. Batang ini menjulang sangat tinggi, dengan tingkat kesalahan mencapai 1.1440, lebih dari dua kali lipat lebih buruk daripada kinerja aslinya. Detektor canggih dan modern seperti DocTR justru tidak cocok *recognizer* Tesseract, yang pada akhirnya merusak kinerjanya.

Tabel 4.12 Perbandingan Hasil Pengenalan Teks OCR *Standalone* vs. OCR *Hybrid* Terbaik vs. OCR *Hybrid* Terburuk pada Skenario Original

Lokasi Teks (Image Path)	**Teks Asli (Ground Truth)**	**Hasil Tesseract *Standalone***	**Hasil *Hybrid* Terbaik (Det : EasyOCR)**	**Hasil *Hybrid* Terburuk (Det : DocTR)**
ocr_img_316PeraturanRek_2.png	1. Undang-Undang Nomor 12 Tahun 2012 tentang Pendidikan Tinggi	1. Undang-Undang Nomer 12 Tahun 2012 tentang Pendidikan Tinggi	1. Undang-Undang Nomor 12 Tahun 2012 tentang Pendidikan Tinggi	1, Undang-Undang Nomor 12 Tahun 2012 tentang Pendidikan Tinggi
ocr_img_316PeraturanRek_3.png	Peraturan Pemerintah Nomor 4 Tahun 2014 tentang Penyelenggaraan	Peraturan Pemerintah Nomor 4 Tahun 2014 tentang Penyelenggaran	Peraturan Pemerintah Nomor 4 Tahun 2014 tentang Penyelenggaraan	Peraturan Pemerintah Nomor 4 Tahun 2014 tentang Penyclenggaraan

Lokasi Teks (Image Path)	**Teks Asli (Ground Truth)**	**Hasil Tesseract *Standalone***	**Hasil *Hybrid* Terbaik (Det : EasyOCR)**	**Hasil *Hybrid* Terburuk (Det : DocTR)**
ocr_img_BakuMutuProgram_4.png	c. bahwa berdasarkan pertimbangan sebagaimana dimaksud dalam	c. bahwa berdasarkan pertimbangan sebagaimana dimaksud dalamn	c. bahwa berdasarkan pertimbangan sebagaimana dimaksud dalam	c. bahwa berdasarkan pertimbangan scbagaimana dimaksud dalam

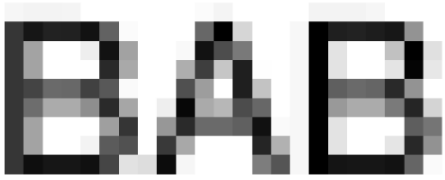
Berdasarkan data dari Tabel 4.12, terdapat beberapa temuan yang menarik untuk diamati. Pada hasil tesseract *Standalone*, kinerja aslinya sudah cukup baik, namun secara konsisten membuat kesalahan-kesalahan kecil yang umum terjadi, seperti salah ketik/*typo* "Penyelenggaran", "dalamn" dan "Nomer" bukan "Nomor". Untuk hasil OCR *Hybrid* Terbaik (Kombinasi Detektor Easy OCR dan *Recognizer* Tesseract OCR) dapat dilihat bahwa kombinasi ini secara konsisten menghasilkan transkripsi yang akurat. Ini adalah bukti nyata adanya kecocokan yang sempurna, di mana input lokasi teks yang sesuai dan bersih dari Easy OCR memungkinkan Tesseract OCR untuk bekerja dengan baik dan mengeliminasi kesalahan-kesalahan. Yang terakhir, pada hasil OCR *Hybrid* terburuk (Kombinasi Detektor DocTR dan *Recognizer* Tesseract OCR), dihasilkan teks yang mengandung kesalahan substitusi karakter (misalnya, "Penyclenggaraan", "scbagaimana") dan kesalahan tanda baca (penggunaan koma). Ini adalah bukti dari ketidakcocokan di mana *recognizer* Tesseract kesulitan mengolah input dari detektor DocTR, sehingga menghasilkan kesalahan yang tidak terjadi pada model *standalone*-nya.



Gambar 4.26 Perbandingan Hasil Deteksi *Bounding box* Detektor Easy OCR (OCR *Hybrid* Terbaik) vs. Detektor DocTR (OCR *Hybrid* Terburuk)

Gambar 4.26 di atas hadir untuk memaparkan karakteristik dari deteksi teks dari kedua detektor yang memiliki hasil kinerja OCR *Hybrid* Terbaik (Detektor Easy OCR) dan detektor yang memiliki hasil evaluasi OCR *Hybrid* Terburuk (Detektor DocTR). Hasil di atas tampak menunjukkan sebuah temuan yang tampaknya agak berkebalikan dari fakta karena deteksi dari DocTR terlihat lebih teliti sementara deteksi dari EasyOCR terlihat lebih kasar. Hal ini menimbulkan pertanyaan penting mengenai hasil evaluasi Detektor Easy OCR yang lebih baik bahkan menjadi yang terbaik dibanding Detektor DocTR yang memiliki hasil evaluasi paling buruk. Untuk mencari kebenaran dan membuktikan bahwa kombinasi *hybrid* tertentu berhasil dan yang lain gagal, kita perlu melihat apa yang sebenarnya diterima dan dilihat oleh *recognizer* pasangannya, dalam kasus ini adalah Tesseract OCR. Gambar 4.27 menyajikan hasil simulasi ini: irisan-irisan gambar dari setiap detektor diserahkan kepada *recognizer* Tesseract untuk dibaca. Hasilnya secara gamblang menunjukkan perbedaan fundamental dalam kualitas input.

Hasil Tesseract: 'BAB'



Hasil Tesseract: 'KETENTUAN UMUM'

Hasil Tesseract: 'Pasal'

Hasil Tesseract: 'Dalam peraturan ini yang dimaksud dengan:'

Hasil Tesseract: 'Dalam peraturan ini yang dimaksud dengan:'

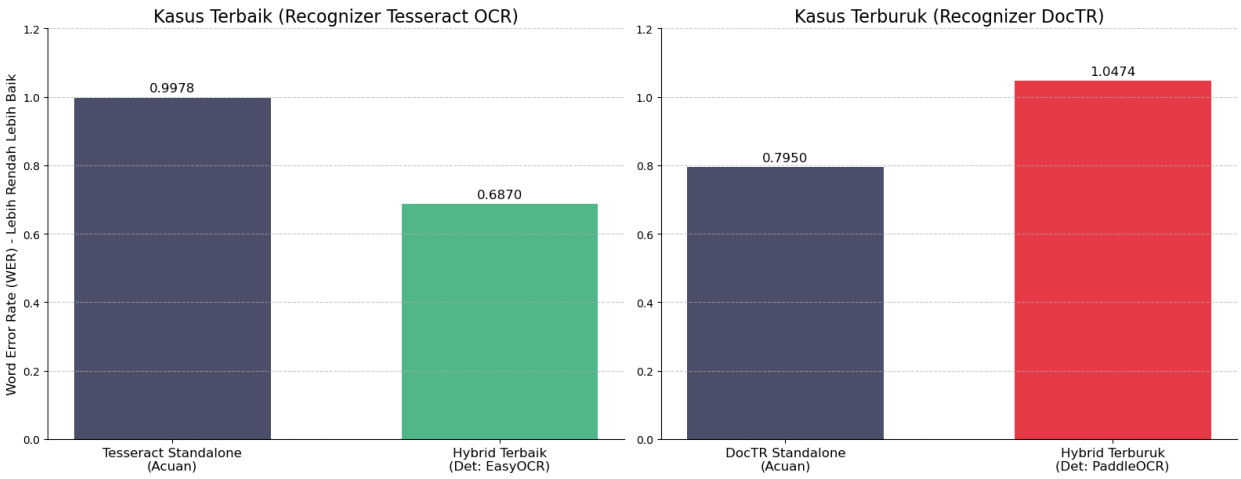
Hasil Tesseract: 'Institut Teknologi Sepuluh Nopember yang selanjutnya disebut ITS adalah perguruan:'

Gambar 4.27 Hasil Deteksi oleh Detektor Easy OCR dan Pengenalan Teks oleh *Recognizer* Tesseract OCR

Visualisasi pada Gambar 4.27 menampilkan irisan-irisan gambar yang dihasilkan oleh Detektor Easy OCR beserta hasil teks yang dibaca oleh *recognizer* Tesseract. Tesseract menerima serangkaian gambar yang panjang dan rapi. Setiap gambar berisi satu baris teks yang utuh dan lengkap. Dengan menerima satu baris penuh alih-alih kata demi kata, *recognizer* Tesseract OCR dapat melihat konteks antar kata, termasuk spasi, yang sangat membantunya dalam proses pengenalan. Tidak ada risiko kata terpotong atau informasi yang hilang di antara kata-kata. Seperti yang terlihat dari teks di atas setiap irisan, Tesseract OCR mampu membaca hampir semua baris dengan presisi. Input yang berkualitas inilah yang menjadi alasan mengapa kombinasi Detektor Easy OCR dan *Recognizer* Tesseract menjadi konfigurasi yang terbaik.

Gambar 4.28 Hasil Deteksi oleh Detektor DocTR dan Pengenalan Teks oleh *Recognizer* Tesseract OCR

Visualisasi pada Gambar 4.28 di atas menampilkan irisan-irisan gambar yang dihasilkan oleh Detektor DocTR beserta hasil teks yang dibaca oleh Tesseract OCR. Tesseract menerima potongan-potongan gambar kecil yang terpisah-pisah. Setiap gambar hanya berisi satu kata atau bahkan satu karakter saja. *Recognizer* Tesseract dipaksa untuk membaca setiap potongan kecil ini secara individual, tanpa konteks dari kata-kata di sekitarnya. Proses ini sangat rentan terhadap kesalahan sehingga menyebabkan hasil bacaan *Recognizer* Tesseract OCR menjadi tidak akurat dan seringkali salah. Perbandingan kedua hasil simulasi ini menyatakan bahwa keberhasilan atau kegagalan sebuah sistem OCR *Hybrid* tidak ditentukan oleh seberapa teliti detektornya, melainkan oleh kualitas dan format input yang diberikannya kepada *recognizer*. Pendekatan deteksi per baris dari Easy OCR, meskipun terlihat lebih kasar, terbukti jauh lebih unggul. Sebaliknya, pendekatan deteksi per kata dari DocTR, meskipun terlihat lebih teliti, justru menciptakan input yang menyebabkan kegagalan sistem.



Gambar 4.29 Perbandingan Hasil Evaluasi OCR *Standalone* (Acuan), OCR *Hybrid* Terbaik, dan OCR *Hybrid* Terburuk pada Skenario Original

Visualisasi pada Gambar 4.29 di atas menyajikan bukti terkait kinerja OCR *Hybrid* saat dihadapkan pada tantangan sesungguhnya yaitu, dokumen dengan tingkat *noise* yang tinggi. Gambar ini menunjukkan peran ketahanan (*robustness*) dari sebuah detektor menjadi penentu keberhasilan atau kegagalan kinerja model. Pada visualisasi bagian kiri ditampilkan kinerja terbaik, di mana kombinasi *hybrid* berhasil meningkatkan kinerja *recognizer* Tesseract OCR. Batang berwarna gelap menunjukkan kinerja asli model OCR Tesseract *Standalone* yang gagal dengan tingkat kesalahan (WER) mencapai 0,9978. Selanjutnya, batang hijau yang jauh lebih pendek menunjukkan hasil dari kombinasi Detektor Easy OCR dan *Recognizer* Tesseract OCR, dengan WER turun drastis menjadi 0,6870. Temuan ini adalah bukti yang kuat bahwa meskipun gambar yang dimasukkan rusak karena *noise*, Detektor Easy OCR terbukti cukup tahan banting. Input yang cukup baik inilah yang memungkinkan *recognizer* Tesseract OCR untuk bekerja jauh lebih efektif dan terhindar dari kegagalan pada kinerjanya.

Di sisi lain, visualisasi bagian kanan menampilkan kasus terburuk, di mana kombinasi *hybrid* justru menurunkan kinerja *recognizer* DocTR yang sebenarnya sudah cukup kuat. Batang berwarna gelap menunjukkan kinerja asli DocTR *Standalone* yang relatif baik pada kondisi *noise*, dengan WER 0,7950. Selanjutnya, batang merah yang lebih tinggi menunjukkan hasil dari kombinasi Detektor Paddle OCR dan *Recognizer* DocTR, di mana WER melonjak menjadi 1,0474, yang berarti kinerjanya jauh lebih memburuk. Ini berarti bahwa Detektor Paddle OCR tidak memiliki ketahanan yang baik terhadap *noise*. Meskipun *recognizer* DocTR kuat, ia tidak dapat memperbaiki kesalahan yang sudah dibuat oleh detektornya. Visualisasi ini menyimpulkan bahwa pada kondisi dokumen yang rusak, ketahanan detektor adalah segalanya. Detektor yang tahan banting (seperti Easy OCR) dapat menjadi penyelamat bagi *recognizer* yang rapuh. Sebaliknya, detektor yang tidak tahan banting (seperti Paddle OCR) justru dapat merusak kinerja *recognizer* yang kuat sekalipun.

Tabel 4.13 Perbandingan Hasil Pengenalan Teks OCR *Standalone* vs. OCR *Hybrid* Terbaik pada Skenario *Noise* Tinggi

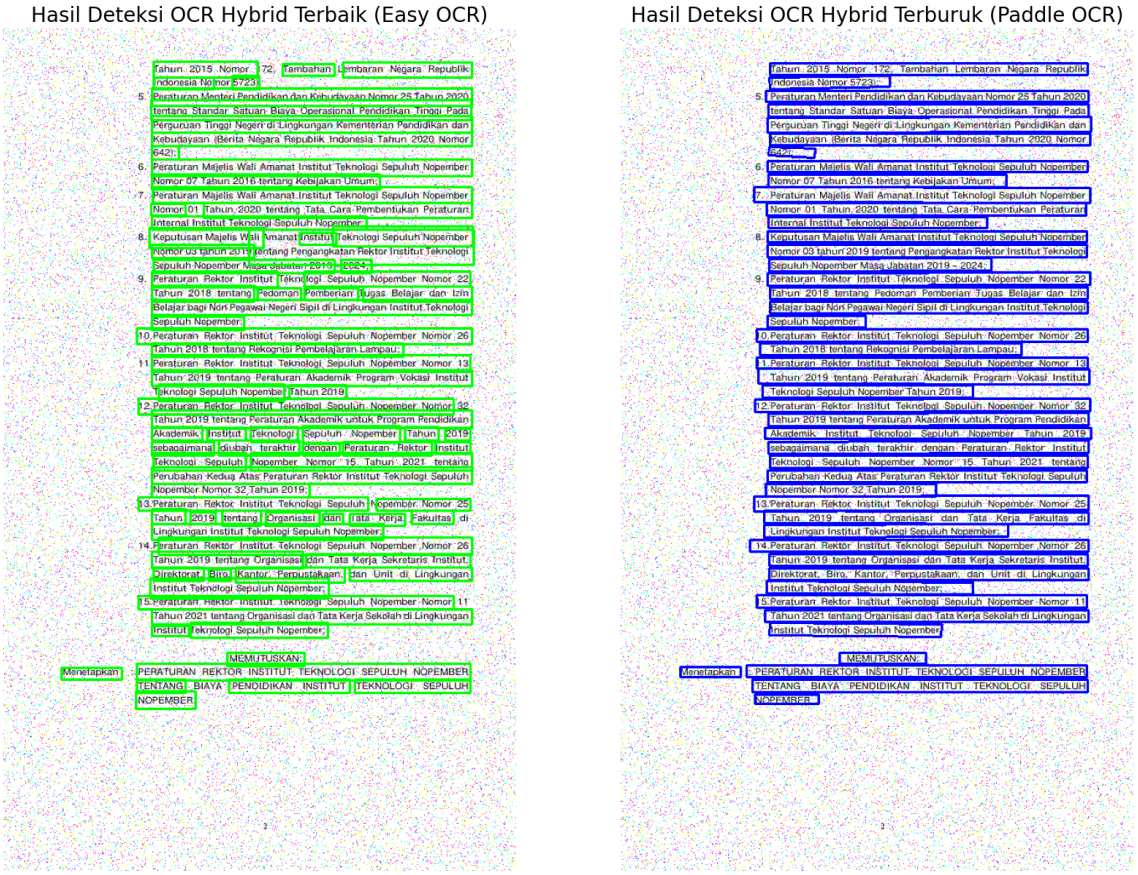
Lokasi Teks (Image Path)	**Teks Asli (Ground Truth)**	**Hasil Tesseract *Standalone* (Gagal)**	**Hasil *Hybrid* Terbaik (Det : Easy OCR)**
ocr_img_ 316PeraturanRek _2.png	9. Peraturan Rektor Institut Teknologi Sepuluh Nopember Nomor 22 Tahun 2018 tentang Pedoman Pemberian Tugas Belajar dan Izin Belajar bagi Pegawai Negeri Sipil di Lingkungan Institut Teknologi Sepuluh Nopember;	9. Peraturan Rektor Institut Teknologi Sepuluh Nopember 22 Tahun 2018 tentang Pemberian Tugas Belajar dan Izin Belajar bagi Pegawai Negeri Sipil di Lingkungan Institut Teknologi Sepuluh Nopember;	9. Peraturan Rektor Institut Teknologi Sepuluh Nopember Nomor 22 Tahun 2018 tentang Pedoman Pemberian Tugas Belajar dan Izin Belajar bagi Pegawai Negeri Sipil di Lingkungan Institut Teknologi Sepuluh Nopember;

Berdasarkan data dari Tabel 4.13 di atas, terdapat beberapa temuan yang bisa kita amati. Yang pertama, pada hasil Tesseract *Standalone*, pada gambar dengan *noise* tinggi, kinerja Tesseract mengalami penurunan. Alhasil, terjadi kehilangan kata-kata kunci penting seperti "Nomor" dan "Pedoman", yang dapat mengubah makna dari kalimat tersebut. Yang kedua, pada hasil OCR *Hybrid*, adanya bantuan input dari Detektor Easy OCR, *recognizer* Tesseract berhasil membaca kalimat dengan akurat, mengembalikan semua kata yang sebelumnya hilang pada model *standalone*-nya.

Tabel 4.14 Perbandingan Hasil Pengenalan Teks OCR *Standalone* vs. OCR *Hybrid* Terburuk pada Skenario *Noise* Tinggi

Lokasi Teks (Image Path)	**Teks Asli (Ground Truth)**	**Hasil DocTR *Standalone* (Ketahanan Tinggi)**	**Hasil *Hybrid* Terburuk (Det : Paddle OCR)**
ocr_img_ 316PeraturanRek _2.png	9. Peraturan Rektor Institut Teknologi Sepuluh Nopember Nomor 22 Tahun 2018 tentang Pedoman Pemberian Tugas Belajar dan Izin Belajar bagi Pegawai Negeri Sipil di Lingkungan Institut Teknologi Sepuluh Nopember;	9. Peraturan Rektor Institut Teknologi Sepuluh Nopember Nomor 22 Tahun 2018 tentang Pedoman Pemberian Tugas Belajar dan Izin Belajar bagi Pegawai Negeri Sipil di Lingkungan Institut Teknologi Sepuluh Nopember;	9. Peraturan Rektor Institut Teknologi Sepuluh Nopember 22 Tahun 2018 tentang Pemberian Tugas Belajar dan Izin Belajar bagi Pegawai Negeri Sipil di Lingkungan Institut Teknologi Sepuluh Nopember;

Berdasarkan data dari Tabel 4.14 di atas, terdapat beberapa temuan yang dapat kita amati bersama. Pertama, pada DocTR *Standalone* di mana Kinerja DocTR asli terbukti sangat kuat dan tahan banting. Meskipun ada *noise*, ia masih mampu membaca kalimat panjang dengan baik. Kedua, pada Hasil OCR *Hybrid* yang ketika dipasangkan dengan Detektor Paddle OCR kinerjanya justru merosot. Hasilnya menjadi tidak lengkap, kehilangan kata "Nomor" dan "Pedoman", sama buruknya dengan Tesseract *Standalone*. Dapat dilihat dengan sangat jelas bahwa kombinasi *hybrid* terbaik (Detektor Easy OCR dan *Recognizer* Tesseract OCR) berhasil mengubah hasil yang tidak lengkap menjadi sempurna sedangkan kombinasi OCR *hybrid* terburuk (Detektor Paddle OCR dan *Recognizer* DocTR) tidak berhasil mendapatkan kinerja yang lebih baik dibandingkan dengan OCR *Standalone*-nya.



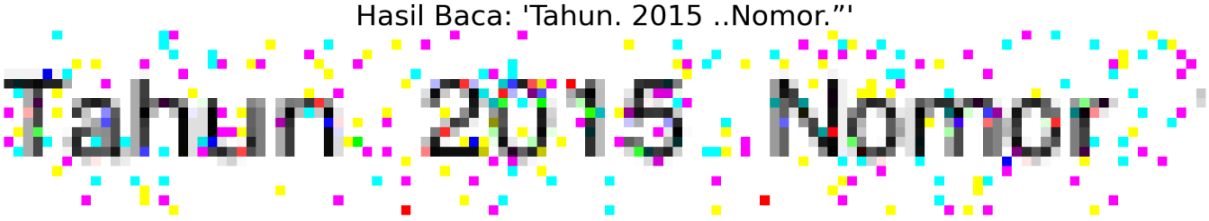
Gambar 4.30 Perbandingan Hasil Deteksi *Bounding box* Detektor Easy OCR (OCR *Hybrid* Terbaik) vs. Detektor Paddle OCR (OCR *Hybrid* Terburuk)

Gambar 4.30 di atas menyuguhkan perbandingan deteksi teks yang digunakan oleh dua detektor yang berbeda pada gambar dengan *noise* tinggi. Visualisasi ini sangat penting untuk memahami tentang kinerja kombinasi OCR *Hybrid* yang bisa sangat berbeda, karena ia menunjukkan secara langsung apa yang dilihat dari masing-masing detektor sebelum informasi diteruskan ke *recognizer*. Pada visualisasi bagian kanan, ditampilkan hasil deteksi dari Detektor Paddle OCR, yang ditandai dengan kotak-kotak berwarna biru. Meskipun gambar aslinya penuh dengan *noise*, kinerja deteksi Paddle OCR menunjukkan beberapa hasil yang baik. Paddle OCR secara konsisten mencoba untuk mendeteksi satu baris teks penuh dalam satu kotak. Meskipun ada beberapa area teks yang terlewat karena tingkat *noise* yang tinggi, area yang berhasil dideteksi dibingkai dengan cukup baik dan bersih. Dapat diakui bahwa kinerja deteksi dari Paddle OCR dapat dinilai cukup baik.

Di sini hal mulai menjadi lebih menarik, pada visualisasi bagian kiri, ditampilkan hasil deteksi dari Detektor Easy OCR, yang ditandai dengan kotak-kotak berwarna hijau. Easy OCR mencoba untuk menjadi sangat teliti dengan mendeteksi kalimat baris ber baris meski tidak dapat dipungkiri bahwa hal tersebut terkadang tidak terpenuhi. Pada gambar dengan *noise* tinggi, kinerja Detektor Easy OCR belum bisa membuahkan hasil yang baik bahkan sedikit lebih buruk dibandingkan Detektor Paddle OCR. Terdapat beberapa kata yang tidak terdeteksi, dan kotak-kotak yang ada seringkali tidak presisi dan rapi. Hal ini merupakan suatu temuan yang menarik karena Detektor Easy OCR merupakan pasangan dari kombinasi OCR *Hybrid* yang memiliki performa terbaik meski performanya tidak bisa menyaingi Detektor Paddle OCR. Perbandingan ini memberikan kesimpulan di mana pada kondisi dokumen dengan *noise* tinggi,

meski pola deteksi dari kedua detektor cenderung sama, namun ada hal yang membuat keduanya masing-masing menjadi pasangan OCR *Hybrid* yang terbaik dan yang satunya menjadi pasangan OCR *Hybrid* yang terburuk. Oleh karena itu, mari kita gali lebih dalam untuk menganalisa penyebab ini dan menemukan titik pokok masalah sebenarnya.

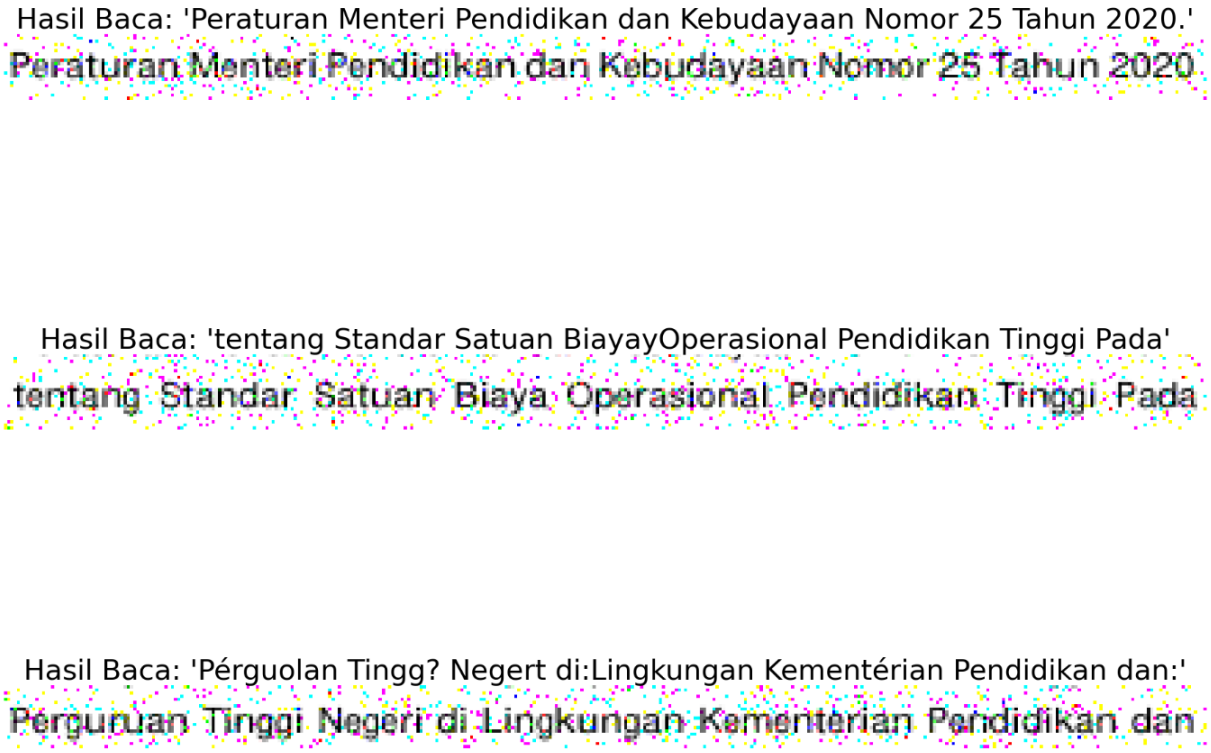
Hasil Baca: 'Tahun. 2015 ..Nomor."'



Hasil Baca: 'Tambahart.'

Hasil Baca: 'embaran Négara- Republik:'

Hasil Baca: 'Indonesia. No'



Gambar 4.31 Hasil Deteksi oleh Detektor Easy OCR dan Pengenalan Teks oleh *Recognizer* Tesseract OCR pada Skenario *Noise* Tinggi

Setelah melihat hasil deteksi pada halaman penuh, bagian ini akan menggali lebih dalam dengan melihat apa yang sebenarnya dilihat dan diterima oleh *recognizer* pada setiap kasus. Gambar-gambar berikut menampilkan hasil irisan-irisan gambar dari setiap detektor diserahkan kepada *recognizer* pasangannya untuk dibaca. Visualisasi di atas menampilkan irisan-irisan gambar yang dihasilkan oleh Detektor Easy OCR, beserta hasil teks yang dibaca oleh *Recognizer* Tesseract untuk setiap irisan. Tesseract menerima serangkaian gambar strip horizontal yang Panjang yang setiap gambar berisi satu baris teks. Seperti yang terlihat dari teks di atas setiap irisan, Tesseract mampu membaca hampir semua baris dengan akurasi yang baik. Berarti secara keseluruhan, meskipun Detektor Easy OCR dapat dikatakan telah sedikit lebih diungguli dari Detektor Paddle OCR, pasangan *recognizer* yang bersanding dengan detector ini berhasil memutarbalikkan keadaan sehingga kecocokan ini membuahkan kinerja yang baik dibandingkan kombinasi OCR *Hybrid* lainnya.

Hasil Baca: 'NOPEMBER:'

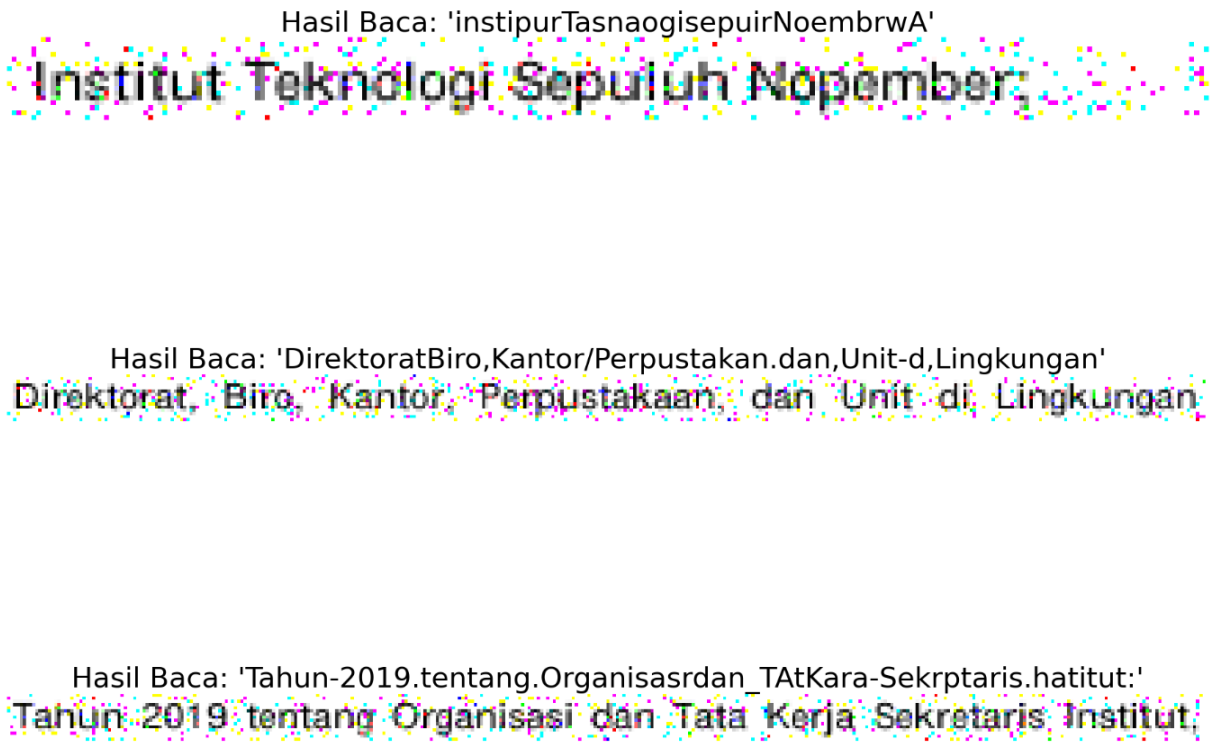
Hasil Baca: 'TENTANG-BIAYAYAPENDIDIKAN.NSTTUT.TEKNOLOGI/,SEPULUH'

Hasil Baca: 'Menetapkan:'

Hasil Baca: 'SPERATURAN'REKTOR-INSTTUFTEKNOLOGFSEPULUH.NOPEMBER'

Hasil Baca: 'aPwaSAaanew'

Hasil Baca: 'Tahun202rtentangorgansaidanTatakafaSakalahdiLingkuingan'



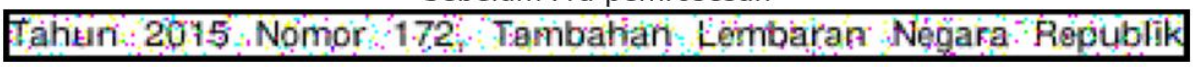
Gambar 4.32 Hasil Deteksi oleh Detektor Paddle OCR dan Pengenalan Teks oleh *Recognizer* DocTR pada Skenario *Noise* Tinggi

Visualisasi di atas menampilkan potongan gambar yang dihasilkan oleh Detektor Paddle OCR beserta hasil teks yang dibaca oleh *Recognizer* DocTR. Dapat kita lihat meskipun Detektor Paddle OCR dinilai lebih presisi dan unggul dibandingkan dengan Detektor Easy OCR, itu bukanlah penentu dari hasil akhir kinerja gabungan ini. Pada skema OCR *Hybrid* keberhasilan ditentukan bukan dari salah satu komponen saja melainkan keduanya harus bisa untuk saling melengkapi dan mengerjakan tugas masing-masing dengan baik. Pada kasus ini, Detektor Paddle OCR memang menyediakan hasil deteksi teks yang baik namun sayangnya tidak ada kecocokan dengan *recognizer* DocTR yang berperan dalam mendeteksi teks. Keberhasilan atau kegagalan sebuah sistem *hybrid* tidak ditentukan oleh seberapa bagus hasil deteksinya saja, melainkan oleh kecocokan yang dimiliki oleh kedua bagian yang berperan itu.

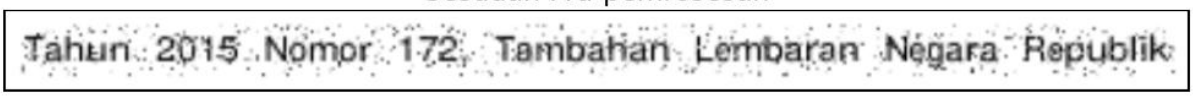
4.2.3 Analisis Kinerja Model OCR *Hybrid* dengan Pra-pemrosesan

Setelah menguji dan mengamati bahwa pendekatan OCR *Hybrid* tidak dapat konsisten dalam mempertahankan peningkatan karena masalah ketidakcocokan antara detektor dan *recognizer*, tahap penelitian selanjutnya ini akan menjelajahi opsi lain untuk meningkatkan kinerja dari model OCR yang sudah diujikan sebelumnya. Khususnya, untuk pengujian pada penelitian kali ini akan mencoba untuk meningkatkan kinerja dari OCR *Hybrid* yang sebelumnya sudah berhasil meningkatkan kinerjanya dalam beberapa kombinasi meski tidak semuanya memiliki hasil yang diinginkan yang berakhir gagal. Sub-bab ini akan meninjau performa dari sebuah *pipeline* pra-pemrosesan yang diterapkan pada *bounding box* yang dihasilkan oleh detektor sebelum diteruskan ke *recognizer*. Pada Gambar 4.33 berikut kita akan melihat data sebelum dan sesudah melalui pra-pemrosesan sekaligus memahami mengapa pra-pemrosesan ini diharapkan dapat meningkatkan kinerja OCR *Hybrid* yang sudah diuji sebelumnya.

Sebelum Pra-pemrosesan

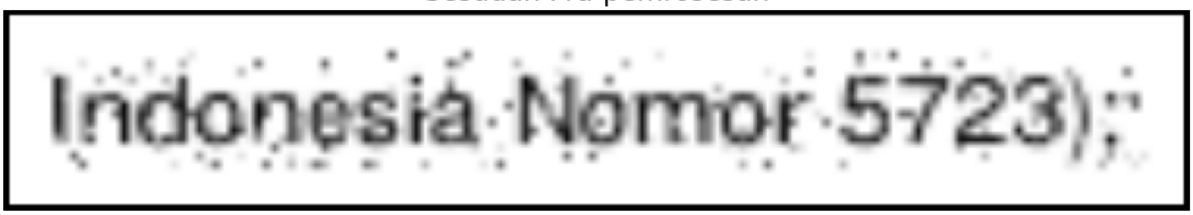


Sesudah Pra-pemrosesan

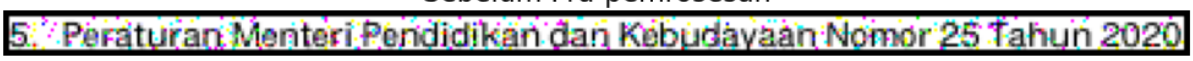


Sebelum Pra-pemrosesan

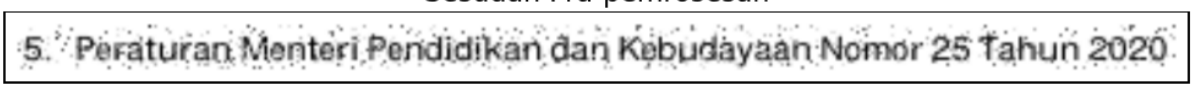
Sesudah Pra-pemrosesan



Sebelum Pra-pemrosesan



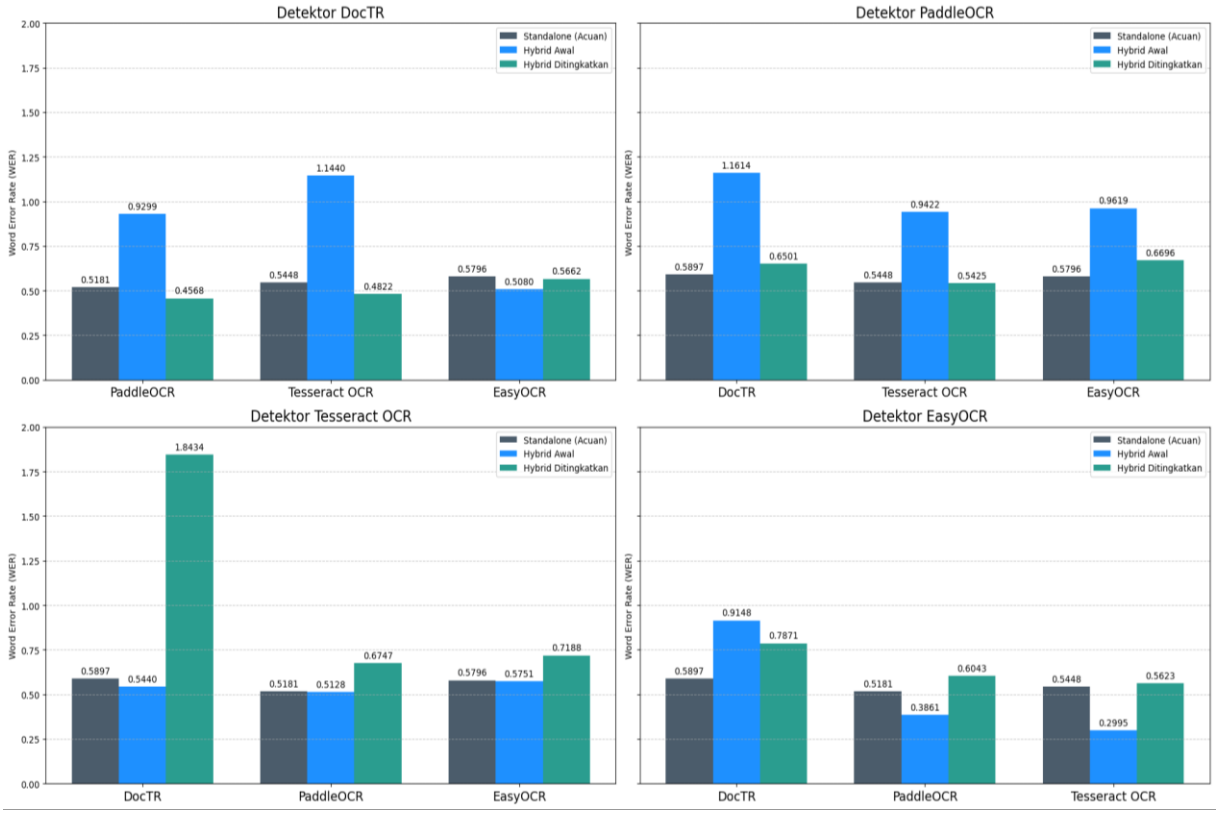
Sesudah Pra-pemrosesan



Gambar 4.33 Perbandingan Gambar Sebelum dan Sesudah Pra-pemrosesan

Terdapat tiga aspek yang didapati dari proses pra-pemrosesan ini yang diharapkan dapat meningkatkan kinerja dari OCR *Hybrid*. Yang pertama, setiap potongan gambar diubah ukurannya agar memiliki tinggi yang seragam (dalam penelitian ini 64 piksel), sambil menjaga rasio aspeknya. Langkah ini sangat penting karena model *recognizer* umumnya dilatih untuk bekerja dengan baik pada teks dengan ukuran input tertentu. Dengan memberikan input yang

konsisten, model diharapkan dapat bekerja lebih akurat juga. Selain penyeragaman ukuran tinggi, yang kedua adalah penambahan *padding* di sekeliling setiap potongan gambar. Tujuannya adalah untuk memberikan ruang di sekitar teks dan memastikan tidak ada bagian dari karakter yang terpotong oleh *bounding box* yang bisa jadi terlalu ketat. Yang terakhir adalah menghilangkan informasi warna pada gambar. Bagi model OCR, bentuk dan kontras karakter adalah yang terpenting, bukan warnanya. Konversi ini membantu menghilangkan gangguan dari informasi warna yang tidak relevan, sehingga model dapat lebih fokus pada tugas pengenalan bentuk. Dengan menerapkan ketiga langkah ini, potongan gambar yang awalnya bervariasi dan mungkin memiliki *noise* berwarna diubah menjadi input yang bersih, seragam, dan terstandarisasi. Bagian selanjutnya akan menganalisis apakah dugaan ini terbukti benar dan sejauh mana pra-pemrosesan ini mampu mengubah kombinasi OCR *Hybrid* yang gagal menjadi berhasil.

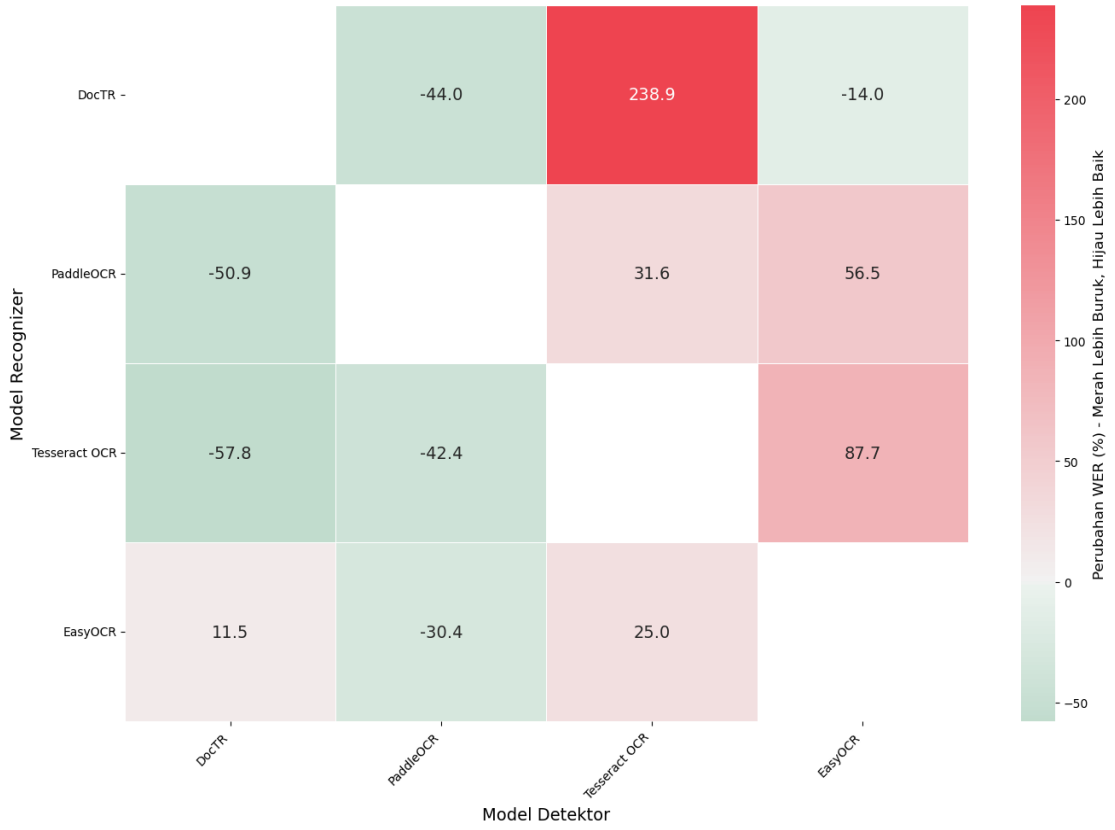


Gambar 4.34 Perbandingan Kinerja Model OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* dengan Pra-pemrosesan pada Skenario Original

Gambar 4.34 di atas menampilkan perbandingan dari tiga kondisi untuk setiap konfigurasi detektor pada Skenario Original (data bersih) dengan kinerja model *Standalone* sebagai acuan, kinerja *Hybrid* Awal sebelum perbaikan, dan kinerja *Hybrid* setelah penerapan pra-pemrosesan. Dari Gambar 4.34 tersebut, dapat diamati dua temuan yang menonjol. Yang pertama, Pada bagian Detektor DocTR dan Detektor Paddle OCR, terlihat pola yang jelas di mana kinerja OCR *Hybrid* Awal (batang biru) sangat buruk, dengan WER yang jauh lebih tinggi dibandingkan *Standalone* (batang abu-abu gelap) yang membuktikan bahwa terjadi ketidakcocokan. Namun, setelah pra-pemrosesan diterapkan, kinerja OCR *Hybrid* Ditingkatkan (batang hijau) tidak hanya jauh membaik, tetapi dalam beberapa kasus seperti kombinasi Detektor DocTR dengan *Recognizer* Paddle OCR bahkan berhasil melampaui kinerja

Standalone aslinya (WER 0,4568 vs. 0,5181). Ini membuktikan bahwa pra-pemrosesan berhasil bertindak sebagai metode alternatif yang efektif.

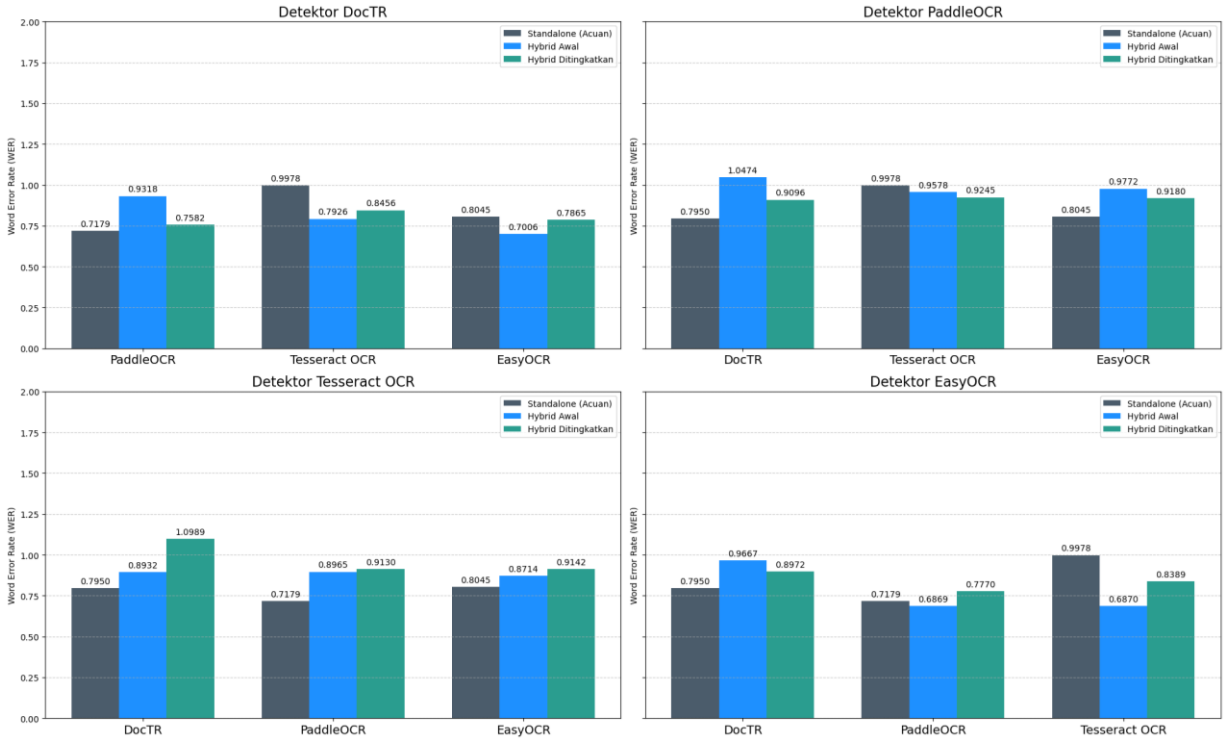
Sebaliknya, pada bagian Detektor Easy OCR dan Detektor Tesseract OCR, terjadi sebuah hal yang unik. Pada kombinasi OCR *Hybrid* Awal tertentu, seperti Detektor Easy OCR dengan *Recognizer* Tesseract OCR, terjalin kecocokan yang sangat baik (WER 0,2995), jauh mengungguli model *Standalone* (WER 0,5448). Anehnya, ketika pra-pemrosesan yang sama diterapkan, kinerja OCR *Hybrid* Ditingkatkan justru memburuk. Hal yang sama terjadi pada bagian Detektor Tesseract OCR, di mana pra-pemrosesan justru menyebabkan peningkatan WER yang sangat tinggi, terutama pada *Recognizer* DocTR (WER 1,8434). Ini menunjukkan bahwa keberadaan pra-pemrosesan juga dapat mengganggu kecocokan yang sudah ada.



Gambar 4.35 Heatmap Kinerja OCR *Hybrid* dengan Pra-pemrosesan pada Skenario Original

Gambar 4.35 di atas menunjukkan persentase perubahan WER, dengan kinerja *Hybrid* Awal sebagai titik acuan. Terdapat beberapa temuan yang bisa diamati dari gambar diatas. Bagian yang berwarna hijau paling pekat terlihat pada kolom Detektor DocTR, terutama saat dipasangkan dengan *Recognizer* Tesseract OCR (-57,8%) dan Paddle OCR (-50,9%). Ini menegaskan bahwa pra-pemrosesan paling efektif ketika diterapkan pada output dari detektor yang kompleks (DocTR) untuk kemudian diteruskan ke *recognizer* lain. Di sisi lain, bagian berwarna merah paling pekat terlihat pada kolom Detektor Tesseract OCR saat dipasangkan dengan *Recognizer* DocTR (+238,9%). Ini menunjukkan kasus terburuk di mana pra-pemrosesan menyebabkan kegagalan yang luar biasa. Selain itu, pada kolom Detektor Easy OCR, terlihat bahwa kombinasi yang awalnya memiliki kinerja baik (dengan *Recognizer* Paddle OCR dan Tesseract OCR) justru mengalami kenaikan WER setelah pra-pemrosesan (+56,5% dan +87,7%).

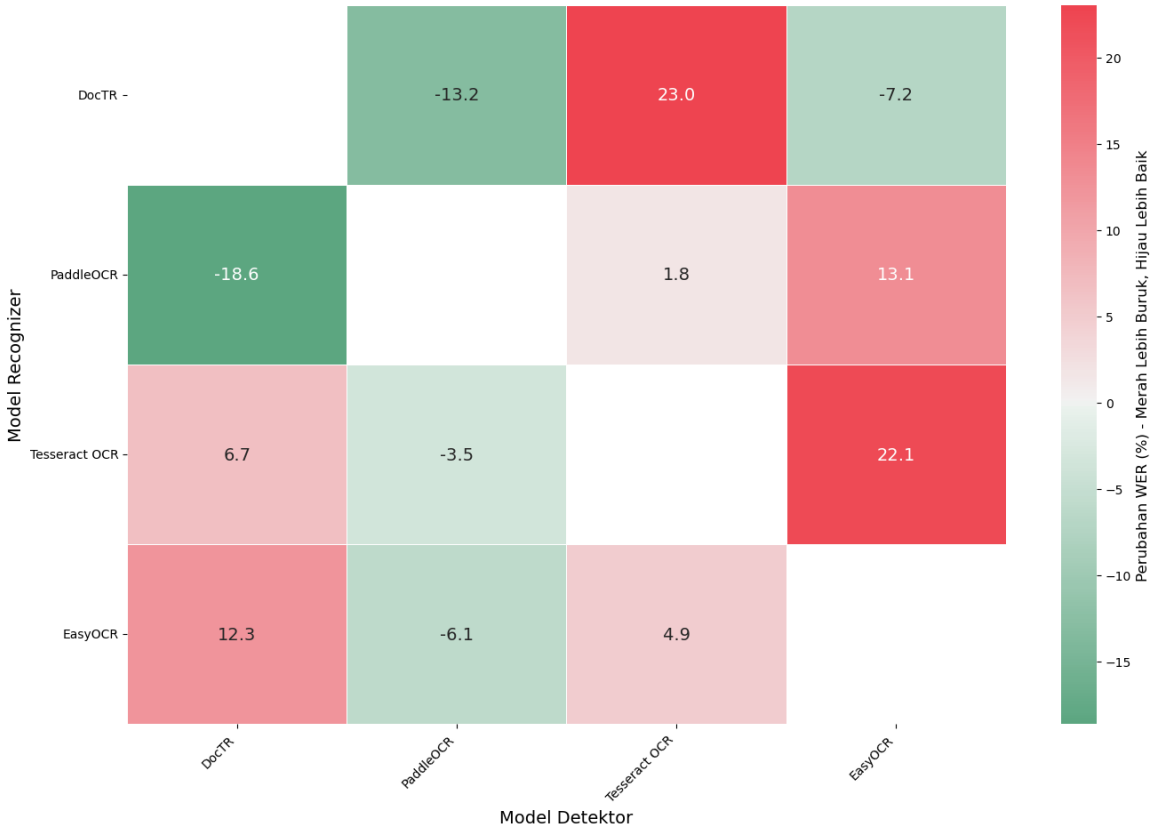
Analisis terhadap Gambar 4.34 dan 4.35 mengarahkan kepada kesimpulan yaitu, pra-pemrosesan pada alur OCR *Hybrid* dapat bersifat selayaknya pedang bermata dua. Tidak dapat diterapkan kepada semua skenario dan kombinasi yang tersedia. Di satu sisi, menjadi sangat berhasil memperbaiki kinerja ketika ada ketidakcocokan antar komponen (misalnya, detektor kompleks dengan *recognizer* yang lebih sederhana). Namun juga menjadi perusak ketika diterapkan pada sistem yang sudah memiliki kecocokan tersendiri atau ketika output detektor awal memiliki karakteristik yang tidak cocok untuk diolah lebih lanjut.



Gambar 4.36 Perbandingan Kinerja Model OCR *Standalone* vs. OCR *Hybrid* vs. OCR *Hybrid* dengan Pra-pemrosesan pada Skenario *Noise* Tinggi

Setelah menguji pada kondisi original, pengujian dilanjutkan ke Skenario *Noise* Tinggi. Tahap ini merupakan uji ketahanan yang bertujuan untuk memahami bagaimana setiap model dari *Standalone*, *Hybrid* Awal, dan *Hybrid* Ditingkatkan dapat bertahan di bawah kondisi dokumen yang dipenuhi dengan *noise* tinggi. Gambar 4.36 di atas menampilkan perbandingan kinerja ketiga pendekatan pada Skenario *Noise* Tinggi. Berbeda dengan Skenario Original, di sini faktor ketahanan (*robustness*) menjadi penentu keberhasilan dalam mempertahankan kinerja masing-masing model.

Pada bagian Detektor Tesseract OCR, semua kombinasi *Hybrid* dengan Pra-pemrosesan gagal meningkatkan kinerja. Ini menunjukkan bahwa detektor ini tidak memiliki ketahanan yang memadai terhadap *noise*. Ketika detektor gagal memberikan *bounding box* yang akurat karena gangguan pada gambar, *recognizer* secanggih apa pun tidak dapat memperbaikinya, sehingga kinerja keseluruhan menurun. Namun, Sebuah temuan yang mengejutkan terlihat pada dampak *Hybrid* Ditingkatkan (batang hijau). Pada hampir semua kasus, terutama pada kombinasi yang awalnya berhasil, seperti Detektor DocTR dan *Recognizer* Tesseract OCR, pra-pemrosesan justru memperburuk kinerja. Langkah pra-pemrosesan yang bersifat umum justru mengganggu mekanisme yang sudah terjalin antara detektor dan *recognizer*.



Gambar 4.37 Heatmap Kinerja OCR *Hybrid* dengan Pra-pemrosesan pada Skenario *Noise* Tinggi

Gambar 4.37 di atas merangkum dampak dari pra-pemrosesan pada Skenario *Noise* Tinggi. Bagian yang berwarna hijau tidak sepekat pada Skenario Original, menunjukkan bahwa perbaikan yang terjadi tidak terlalu signifikan dibandingkan dengan peningkatan error yang terjadi. Perbaikan terbaik terlihat pada kombinasi Detektor DocTR dengan *Recognizer* Paddle OCR (-18,6%) dan DocTR (-13,2%). Ini menunjukkan bahwa pra-pemrosesan hanya sedikit membantu pada kombinasi yang awalnya sudah sangat tidak cocok. Di sisi lain, bagian yang berwarna merah mendominasi, menegaskan bahwa pra-pemrosesan lebih sering merusak kinerja pada kondisi *noise*. Kasus terburuk ada terdapat pada kombinasi Detektor Easy OCR dengan *Recognizer* Tesseract OCR, di mana terjadi kenaikan WER sebesar 22,1%. Ini adalah bukti terkuat bahwa campur tangan pra-pemrosesan mengganggu sinergi yang paling berhasil dalam menangani *noise*.

Dalam menghadapi dokumen dengan *noise*, faktor terpenting dalam sistem *hybrid* adalah kemampuan detektor untuk mengatasi *noise*. Detektor yang memiliki ketahanan baik (*robust*) dapat meningkatkan kinerja *recognizer* yang lemah. Ini tentunya berbeda dengan Skenario Original yang mana pada kondisi *noise* tinggi, pra-pemrosesan pada alur OCR *Hybrid* lebih sering memberikan kegagalan. Hal ini kemungkinan karena kecocokan dalam menangani *noise* bersifat lebih kompleks dan mudah terganggu oleh proses standardisasi seperti yang telah kita ujikan. Dengan demikian, untuk aplikasi dunia nyata yang melibatkan dokumen *scan* berkualitas buruk, strategi yang paling disarankan bukanlah mencoba memperbaiki kombinasi *hybrid* dengan pra-pemrosesan, melainkan memilih kombinasi OCR *Hybrid* Awal yang memang memiliki ketahanan *noise* terbaik, yaitu pasangan yang melibatkan detektor yang terbukti memiliki ketahanan yang bagus seperti DocTR atau Easy OCR.

Jika kualitas dokumen bersih, langkah selanjutnya adalah mengevaluasi kinerja awal dari kombinasi OCR *Hybrid* Awal. Ketika pasangan detektor-*recognizer* tidak cocok dan menghasilkan WER yang tinggi, maka diperlukan langkah lanjutan. Direkomendasikan untuk menerapkan pra-pemrosesan sebagai strategi optimal yang terbukti efektif dalam kasus Detektor DocTR, di mana pra-pemrosesan berhasil meningkatkan kinerja secara drastis. Apabila sebuah kombinasi secara tak terduga sudah menunjukkan sinergi yang baik (WER rendah), maka strategi terbaik adalah menggunakannya apa adanya. Penggunaan alternatif pra-pemrosesan justru berisiko mengganggu kecocokan yang sudah ada, seperti yang terjadi pada kasus Detektor Easy OCR dengan *Recognizer* Tesseract OCR.

Untuk dokumen berkualitas buruk, prioritas utama berubah dari memperbaiki menjadi bertahan. Strategi yang paling disarankan adalah memilih kombinasi OCR *Hybrid* Awal yang menggunakan detektor dengan ketahanan *noise* terbaik (misalnya, DocTR atau Easy OCR). Analisis membuktikan bahwa kemampuan detektor untuk memberikan *bounding box* yang baik di tengah *noise* adalah faktor penentu keberhasilan. Diagram ini menyarankan untuk menghindari pra-pemrosesan pada skenario ini. Hal ini didasarkan pada temuan bahwa pra-pemrosesan pada kondisi *noise* lebih sering memberikan dampak negatif. Secara keseluruhan, diagram alir ini menyimpulkan bahwa optimasi OCR *Hybrid* bukanlah tentang mencari satu-satunya solusi terbaik, melainkan tentang menerapkan pendekatan yang sesuai konteks.

BAB 5 Kesimpulan dan Saran

5.1 Kesimpulan

Berdasarkan seluruh hasil dapat ditarik beberapa kesimpulan utama yang secara langsung menjawab rumusan masalah dari penelitian ini.

1. Kinerja model OCR *standalone* menunjukkan adanya *trade-off* antara akurasi, kecepatan, dan ketahanan terhadap *noise*, sehingga tidak ada satu model pun yang dapat dinobatkan sebagai yang terbaik**.** Pada kondisi dokumen bersih (Skenario Original), PaddleOCR menunjukkan akurasi pengenalan kata (WER) tertinggi, sementara Tesseract OCR jauh lebih unggul dalam kecepatan pemrosesan. Namun, ketika dihadapkan pada dokumen dengan tingkat *noise* yang tinggi, hasil ini berubah. DocTR terbukti menjadi model yang paling tahan banting (*robust*), dengan penurunan kualitas hasil yang paling minimal. Sebaliknya, Tesseract OCR yang semula tercepat, justru menjadi model yang paling rentan dan mengalami penurunan kinerja paling drastis. Temuan ini menegaskan bahwa pemilihan model OCR *standalone* harus didasarkan pada beberapa faktor, akurasi pada data bersih, kecepatan pemrosesan, atau ketahanan terhadap kualitas input yang buruk.
2. Pendekatan OCR *Hybrid* terbukti bukan merupakan strategi yang tepat untuk meningkatkan akurasi. Penelitian ini menunjukkan bahwa sebagian besar dari kombinasi *hybrid* yang diuji justru menghasilkan kinerja yang lebih buruk. Sebagai contoh, kombinasi Detektor Paddle OCR dengan *Recognizer* DocTR meningkatkan *Word Error Rate* (WER) dari 0,5897 menjadi 1,1614, sebuah penurunan kinerja yang drastis. Kegagalan ini disebabkan oleh ketidakcocokan antara detektor dan *recognizer* yang berbeda. Meskipun demikian, ditemukan beberapa kasus kecocokan yang berhasil, seperti pada kombinasi Detektor Easy OCR dengan *Recognizer* Tesseract OCR yang berhasil menekan WER dari 0,5447 (kinerja *standalone*) menjadi 0,2995. Namun, karena keberhasilan ini sulit diprediksi, pendekatan OCR *Hybrid* tidak dapat direkomendasikan sebagai solusi yang mutlak.
3. Penambahan *pipeline* pra-pemrosesan juga memiliki hasil yang kurang lebih serupa dengan pendekatan OCR *Hybrid* sebelumnya. Pada dokumen bersih (Skenario Original), pra-pemrosesan terbukti mampu memperbaiki kombinasi detector dan *recognizer* yang sebelumnya tidak cocok. Contohnya, kombinasi DocTR dan Paddle OCR mengalami penurunan WER signifikan dari 0,7313 menjadi 0,4568, bahkan mengungguli model *Standalone* Paddle OCR (WER 0,5181). Namun, pada kombinasi yang sejak awal sudah cocok, seperti Easy OCR dan Tesseract OCR, penerapan pra-pemrosesan justru merusak kinerja, meningkatkan WER dari 0,2995 menjadi 0,5623 (+87,7%). Bahkan pada kombinasi Tesseract dan DocTR, WER melonjak hingga 1,8434 (+238,9%). Sebaliknya, pada dokumen dengan *noise* tinggi (Skenario *Noise* Tinggi), efektivitas *pipeline* menurun drastis. Hanya sedikit kombinasi yang menunjukkan perbaikan, seperti DocTR dan Paddle OCR dengan penurunan WER sebesar 18,6%, sedangkan sebagian besar mengalami penurunan performa. Oleh karena itu, *pipeline* pra-pemrosesan bukan solusi menyeluruh, tetapi strategi perbaikan yang efektif hanya dalam kondisi tertentu.

Dengan demikian, dapat disimpulkan bahwa untuk meningkatkan akurasi OCR pada dokumen akademik yang diberikan *noise*, strategi menggabung-gabungkan model (*hybrid*) bukanlah jalan yang disarankan untuk diberlakukan secara menyeluruh, begitu pun juga untuk pendekatan pra-pemrosesan.

5.2 Saran

Berdasarkan kesimpulan yang telah ditarik dari hasil penelitian dan pembahasan, berikut adalah beberapa saran yang dapat diberikan. Saran ini terbagi menjadi dua, yaitu saran praktis yang dapat langsung diterapkan dan saran untuk pengembangan penelitian sejenis di masa depan.

1. Mengingat bahwa *pipeline* pra-pemrosesan pada penelitian ini hanya berhasil dalam skenario tertentu dan bahkan bersifat merusak pada kombinasi yang sudah cocok, maka arah penelitian selanjutnya disarankan untuk mengembangkan *pipeline* pra-pemrosesan yang bersifat adaptif. Artinya, proses prapemrosesan gambar seperti resizing, padding, hingga konversi grayscale sebaiknya tidak diterapkan secara seragam, melainkan disesuaikan berdasarkan karakteristik dari model OCR yang digunakan serta tingkat *noise* dari dokumen input. Misalnya, sistem dapat secara otomatis mendeteksi bahwa output *bounding box* dari detektor DocTR cenderung terlalu sempit dan memerlukan padding tambahan, sementara output dari Easy OCR mungkin cukup baik tanpa modifikasi tambahan. Dengan adanya *pipeline* yang mampu mengidentifikasi kapan dan bagaimana transformasi diterapkan, diharapkan sistem dapat menghindari risiko *overprocessing* dan meningkatkan efektivitas pra-pemrosesan, terutama ketika dihadapkan pada dokumen yang sangat bervariasi dari sisi kualitas visual.
2. Hasil penelitian ini menunjukkan bahwa pada skenario *noise* tinggi, bahkan *pipeline* terbaik sekalipun gagal menyelamatkan performa OCR *Hybrid*, terutama ketika detektor tidak mampu menghasilkan *bounding box* yang akurat. Hal ini menunjukkan bahwa dalam kondisi ekstrem, satu-satunya jalur penyelamatan informasi terletak pada tahap pasca-OCR, yaitu melalui koreksi teks berbasis Natural Language Processing (NLP). Oleh karena itu, saran penting untuk penelitian mendatang adalah menambahkan *post-processing* untuk melakukan koreksi terhadap output teks yang telah dihasilkan, baik dari model *standalone* maupun *hybrid*. Dengan demikian, penggabungan OCR dan NLP berpotensi menciptakan sistem yang lebih tahan terhadap kerusakan dokumen, memperbaiki hasil akhir, dan memberikan kualitas teks yang lebih dapat digunakan untuk keperluan informasi praktis.
3. Seluruh pengujian dalam penelitian ini menggunakan dataset hasil augmentasi dari dokumen digital, yang meskipun terstruktur rapi dan mewakili berbagai *noise* buatan, tetap belum sepenuhnya merepresentasikan kondisi sebenarnya dari dokumen scan fisik yang ditemukan di institusi pendidikan atau arsip publik. Oleh karena itu, saran penting berikutnya adalah untuk memperluasnya dengan menguji sistem terhadap dataset hasil scan asli dari dokumen kertas, termasuk yang mengalami degradasi nyata seperti blur goresan, lipatan, dan sebagainya. Dengan pendekatan evaluasi seperti ini pengembangan ke depan akan lebih matang dan relevan untuk diterapkan di lingkungan dunia nyata.

DAFTAR PUSTAKA

Abdelaziz, T. A. I., & Fazil, U. (2023). Applications of integration of AI-based *Optical Character Recognition* (OCR) and Generative AI in Document Understanding and Processing. Applied Research in Artificial Intelligence and Cloud Computing, 6(11), 1–16. Retrieved from https://researchberg.com/index.php/araic/article/view/171

Blecher, L., et al. (2023). Nougat: Neural Optical Understanding for Academic Documents. Retrieved from https://arxiv.org/abs/2308.13418

Kim, G., et al. (2022). OCR-free document understanding transformer. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 121-137)

Kumar, J. P., & Dharshan, H. D. (2025). *Handwritten Text Recognition Using Deep learning: A CNN-LSTM Approach*. International Journal of Intelligent Systems and Applications in Engineering (IJISAE), Original Research Paper

Prasetiadi, A., et al. (2022). *Deep learning* Approaches for Nusantara Scripts *Optical Character Recognition* using *Convolutional Neural Network*, ConvMixer, and Visual Transformer. Indonesian Journal of Computing and Cybernetics Systems, 16(3), (pp. 325-336)

Rakshit, A., Mehta, S., & Dasgupta, A. (2023). *A Novel Pipeline for Improving Optical Character Recognition through Post-processing Using Natural Language Processing*. Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati.

Sánchez, S., Zariquiey, R., & Oncevay, A. (2024). *Unlocking Knowledge with OCR-Driven Document Digitization for Peruvian Indigenous Languages*. In *Proceedings of the 4th Workshop on Natural Language Processing for Indigenous Languages of the Americas* (pp. 103–111). Association for Computational Linguistics

Sinha, R., & Rekha, R. B. (2025). *Digitization of Document and Information Extraction using OCR*. RV College of Engineering.

Sridhar, S., et al. (2021). Character Recognition Using *Deep learning* Algorithm. Turkish Journal of Computer and Mathematics Education, 12(14), (pp. 1165–1174)

Suhajda, A., et al. (2020). Challenges and application opportunities of *Optical Character Recognition* using multilayer perceptron models in the accounting domain – Economics & Working Capital. Retrieved from http://eworkcapital.com/challenges-and-application-opportunities-of-optical-character-recognition-using-multilayer-perceptron-models-in-the-accounting-domain/

Vanitha, M., Sumathi, G., Logesh, P., Chakravarthi, S. M., & Surya, V. (2025). *AI-Powered OCR for Digitizing Historical Document in Regional Languages*. International Journal of Innovative Research in Science, Engineering and Technology, 14(3), 2917–2921

Halaman ini sengaja dikosongkan.

LAMPIRAN

Halaman ini sengaja dikosongkan.

BIODATA PENULIS

Penulis dilahirkan di Sidoarjo, 23 Mei 2002, merupakan anak pertama dari 2 bersaudara. Penulis telah menempuh pendidikan formal yaitu di TK Kristen Petra 12 Sidoarjo, SD Kristen Petra 12 Sidoarjo, SMP Kristen Petra 4 Sidoarjo dan SMA Kristen Petra 4 Sidoarjo. Setelah lulus dari SMA pada tahun 2020, Penulis mengikuti SM dan diterima di Departemen Teknik Informatika FTEIC- ITS pada tahun 2020 dan terdaftar dengan NRP 5025201087.

Di Departemen Teknik Informatika Penulis sempat aktif di beberapa kegiatan kepanitiaan lepas yang diselenggarakan oleh beberapa organisasi seperti Lembaga Minat dan Bakat (LMB ITS) dan Badan Eksekutif Mahasiswa (BEM ITS) dan aktif sebagai Asisten Mata Kuliah Dasar Pemrograman serta menjadi grader praktikum mata kuliah tersebut.