BAB II

TINJAUAN LITERATUR

2.1 Penelitian Terkait

Tinjauan pustaka dalam penelitian ini bertujuan untuk mengkaji dan menganalisis berbagai penelitian terdahulu yang relevan dengan topik ekstraksi data otomatis dari Kartu Tanda Mahasiswa (KTM) menggunakan Optical Character Recognition (OCR) dan Regular Expression (RegEx) yang diimplementasikan dalam antarmuka Graphical User Interface (GUI) berbasis Tkinter (Kurnia, 2024). Kajian ini mencakup studi tentang metode OCR untuk pengenalan teks pada citra kartu, penggunaan Regular Expression sebagai teknik pencocokan pola untuk menyaring dan mengekstraksi informasi spesifik seperti nama, NIM, dan program studi, serta penerapan Tkinter sebagai library Python untuk membangun GUI yang interaktif dan mudah digunakan. Melalui tinjauan ini, diharapkan dapat ditemukan kelebihan dan keterbatasan dari pendekatan-pendekatan sebelumnya, sehingga penelitian ini dapat merancang sistem ekstraksi data KTM yang lebih akurat dan efisien (Nasution et al., 2025).

Penelitian yang dilakukan oleh Latifa Khoirani dkk, bertujuan untuk mengotomatisasi ekstraksi informasi dari struk transaksi keuangan menggunakan algoritma OCR, karena di banyak usaha kecil dan menengah pencatatan bukti transaksi sering terabaikan akibat bukti yang hilang atau tidak tercatat sejak awal. Metode yang digunakan meliputi pengumpulan 20 foto struk menggunakan CamScanner, preprocessing berupa cropping untuk menghilangkan elemen seperti

barcode, kemudian ekstraksi teks dengan OCR dan penyimpanan hasil ke Microsoft Word. Hasil penelitian menunjukkan bahwa algoritma OCR mampu mengekstrak informasi penting seperti tanggal, jumlah transaksi, dan rincian lainnya dengan akurasi tinggi, serta aplikasi yang dikembangkan dapat menghemat waktu dan meminimalkan kesalahan input data manual. Saran dari peneliti adalah mengintegrasikan machine learning untuk meningkatkan akurasi pada struk berkualitas rendah dan melakukan pengujian dengan skala data yang lebih besar (Latifa Khoirani, Rino Ariansyah and Supiyandi Supiyandi, 2024).

Penelitian oleh Muhammad Nashiruddin dkk, bertujuan untuk mengimplementasikan sistem OCR berbasis Tesseract guna mengekstrak data dari Kartu Tanda Mahasiswa (KTM) Universitas Muhammadiyah Klaten secara otomatis, karena input data manual dinilai tidak efisien dan rawan kesalahan. Metode yang digunakan meliputi pra-pemrosesan citra dengan OpenCV melalui konversi grayscale dan binarisasi Otsu, kemudian teks dikenali menggunakan Tesseract OCR Engine dan dipisahkan per field seperti Nama, NIM, dan Program Studi menggunakan Regular Expression (Regex). Hasil pengujian menunjukkan sistem mampu mengekstrak informasi dengan akurasi rata-rata di atas 95%, bahkan beberapa field mencapai 100%. Kesimpulannya, sistem berbasis Tesseract ini layak digunakan untuk otomatisasi data KTM secara lokal, namun disarankan adanya pengembangan metode pasca-pemrosesan seperti algoritma koreksi teks untuk menangani kesalahan pengenalan karakter agar keandalan sistem lebih meningkat (Nashiruddin et al., 2025).

Penelitian oleh Marshanda Dwi Maharani Saraun dkk, bertujuan menerapkan teknologi OCR berbasis Google Vision API untuk mentransformasi formulir perbankan Bank Sulutgo menjadi data digital. Metode yang digunakan adalah kuantitatif eksperimen dengan sistem OCR yang dibangun menggunakan Google Vision API, kemudian dievaluasi menggunakan confusion matrix untuk mengukur akurasi, precision, recall, dan F1-score dengan membandingkan hasil ekstraksi terhadap data ground truth. Hasil penelitian menunjukkan bahwa sistem OCR mampu mengekstraksi sebagian besar data penting pada formulir perbankan dengan tingkat akurasi yang bervariasi, tergantung kualitas hasil pemindaian dan karakteristik tulisan tangan; pada dokumen dengan kualitas gambar baik dan struktur teks jelas, sistem menunjukkan kinerja optimal, namun pada formulir berkualitas rendah atau tulisan tidak konsisten masih ditemukan kesalahan pembacaan. Kesimpulannya, penerapan sistem ini meningkatkan efisiensi input data dan mengurangi ketergantungan pada proses manual, dan disarankan untuk penelitian selanjutnya meningkatkan akurasi melalui optimasi pra-pemrosesan citra serta memperluas variasi dataset formulir termasuk kondisi dokumen yang kurang ideal (Dwi et al., 2026).

Penelitian oleh Azwar Yusuf dkk, bertujuan untuk mengembangkan sistem ekstraksi metadata artikel (judul, nama penulis, afiliasi, dan abstrak) dari file PDF pada arsip jurnal Teknoif menggunakan metode Regular Expression (RegEx), karena selama ini proses pencarian informasi metadata dilakukan secara manual dengan membuka dokumen satu per satu yang memakan waktu. Metode pengembangan sistem menggunakan Extreme Programming (XP) dengan dua kali

iterasi yang berfokus pada pengkodean, melalui tiga tahap utama yaitu ekstraksi teks dari file PDF, pembersihan data (cleaning) dengan tiga kriteria (menghapus header dan footer), serta ekstraksi metadata menggunakan pola RegEx untuk mengidentifikasi judul, penulis, afiliasi, dan abstrak berdasarkan pola teks yang umum. Hasil penelitian menunjukkan bahwa sistem berhasil mengekstraksi informasi penting secara otomatis, seluruh proses berjalan sesuai fungsionalitas yang diharapkan, dan pengujian black-box tidak menemukan bug signifikan. Kesimpulannya, sistem ini dapat membantu pengelolaan arsip jurnal Teknoif, dan disarankan untuk pengembangan selanjutnya agar sistem dapat dioptimalkan tidak hanya pada format jurnal Teknoif tetapi juga berbagai format jurnal lain yang memiliki variasi struktur berbeda (Yusuf et al., 2025).

Penelitian oleh Kurnia Meiliyani dkk, bertujuan untuk menerapkan ekspresi regular (regex) dalam meningkatkan efisiensi pencarian produk pada aplikasi e-commerce, karena fitur pencarian yang cepat dan akurat sangat penting untuk memberikan pengalaman pengguna yang optimal di tengah meningkatnya jumlah produk yang tersedia. Metode yang digunakan adalah pendekatan eksperimen kuantitatif dengan membandingkan pencarian berbasis regex terhadap metode pencarian tradisional pada dataset produk e-commerce, memanfaatkan fitur regex seperti wildcard untuk fleksibilitas pencarian, anchors untuk menentukan posisi pola, serta quantifiers untuk mengatur jumlah kemunculan karakter. Hasil penelitian menunjukkan bahwa penerapan regex mampu mengurangi waktu pencarian hingga 30% dan meningkatkan akurasi pencocokan kata kunci hingga 95%, terutama karena kemampuannya menangani variasi input

seperti salah eja atau format yang tidak konsisten. Kesimpulannya, regex terbukti lebih efisien dan akurat dibandingkan metode pencarian tradisional, (Kurnia Meiliyani *et al.*, 2025).

Kesimpulan mengenai keterbaruan yang akan dikerjakan terletak pada penggabungan OCR dan Regular Expression dalam sebuah aplikasi desktop berbasis GUI Tkinter yang khusus ditujukan untuk ekstraksi data dari Kartu Tanda Mahasiswa (KTM). Penelitian ini menawarkan kontribusi berupa GUI interaktif menggunakan Tkinter yang memudahkan pengguna dalam mengunggah citra KTM, memproses ekstraksi secara otomatis, serta menampilkan dan mengedit hasil ekstraksi field seperti Nama, NIM, Program Studi, dan Fakultas dalam satu platform terintegrasi. Dengan demikian, penelitian Anda tidak hanya menguji akurasi OCR dan Regex, tetapi juga menghasilkan aplikasi siap pakai yang lebih user-friendly dibandingkan penelitian-penelitian sebelumnya.

2.2 Kartu Tanda Mahasiswa (KTM)

Kartu Tanda Mahasiswa (KTM) merupakan dokumen identitas resmi yang diterbitkan oleh perguruan tinggi sebagai bukti status kemahasiswaan seseorang. KTM memuat informasi vital mahasiswa seperti Nomor Induk Mahasiswa (NIM), nama lengkap, program studi, fakultas, dan tempat tanggal lahir (Reswan *et al.*, 2024). Informasi tersebut tercetak dengan format yang bervariasi antar institusi, namun secara umum mengikuti pola penulisan label diikuti tanda titik dua dan nilai data.

Proses pencatatan data dari KTM yang masih dilakukan secara manual oleh petugas administrasi memakan banyak waktu dan berpotensi menimbulkan

kesalahan entri data, terutama ketika volume data yang harus diproses cukup besar. Kesalahan kecil dalam pencatatan NIM atau nama mahasiswa dapat berdampak signifikan pada proses berikutnya seperti verifikasi kehadiran, pencetakan sertifikat, ijazah, hingga pengajuan beasiswa (Firin and Sriani, 2025). Oleh karena itu, otomatisasi proses ekstraksi data KTM menjadi kebutuhan yang mendesak.

2.3 Optical Character Recognation (OCR)

Optical Character Recognition (OCR) adalah teknologi yang memungkinkan komputer untuk mengenali dan mengubah teks yang terdapat pada gambar atau dokumen hasil pemindaian menjadi teks yang dapat diedit dan diproses secara digital. OCR bekerja dengan menganalisis pola visual dari karakter pada citra dan mencocokkannya dengan basis data karakter yang dikenal (Rahmadani, Rozikin and Karawang, 2021).

Proses kerja OCR secara umum melibatkan beberapa tahap, yaitu akuisisi citra, prapemrosesan citra, segmentasi, ekstraksi fitur, pengenalan karakter, dan pascapemrosesan. Performa sistem OCR sangat dipengaruhi oleh kualitas citra masukan, tingkat noise, pencahayaan, serta jenis dan ukuran font yang digunakan pada dokumen

2.4 Tesseract OCR Engine

Tesseract adalah mesin OCR open-source yang dikembangkan oleh Hewlett-Packard pada tahun 1985 dan kemudian dilanjutkan pengembangannya oleh Google sejak tahun 2006. Tesseract mendukung lebih dari 100 bahasa termasuk

bahasa Indonesia, dan menyediakan dua mode mesin utama melalui parameter --oem, yaitu LSTM (Long Short-Term Memory) dan mode warisan (legacy engine) (Kusnantoro, Kusumaningrum, Sulistya and Rohana, 2022).

Tesseract dapat dikonfigurasi dengan berbagai parameter untuk mengoptimalkan hasil pengenalan karakter. Parameter --psm (Page Segmentation Mode) menentukan cara Tesseract memandang halaman, di mana nilai 6 berarti mengasumsikan satu blok teks seragam yang cocok untuk dokumen dengan format terstruktur seperti KTM. Parameter --oem 3 menggunakan mode LSTM yang memberikan akurasi lebih tinggi dibandingkan mode warisan. Dalam penelitian ini, konfigurasi yang digunakan adalah --oem 3 --psm 6 -l ind+eng untuk mendukung pengenalan karakter bahasa Indonesia dan Inggris secara bersamaan.

Pada Python, Tesseract diakses melalui pustaka pytesseract yang berfungsi sebagai wrapper. Fungsi utama yang digunakan adalah image_to_string() yang mengambil citra sebagai masukan dan mengembalikan string teks hasil pengenalan OCR.

2.5 Prapemrosesan Citra Digital

Prapemrosesan citra merupakan serangkaian operasi yang diterapkan pada citra digital sebelum diproses lebih lanjut oleh sistem OCR. Tujuannya adalah meningkatkan kualitas citra sehingga karakter pada dokumen lebih mudah dikenali oleh mesin OCR. Dalam penelitian ini, tahapan prapemrosesan citra yang diterapkan adalah sebagai berikut.

1. Resize Citra

Citra diubah ukurannya agar berada pada rentang resolusi yang optimal untuk Tesseract OCR. Citra yang terlalu besar (lebar lebih dari 1600 piksel) diperkecil untuk menghemat waktu pemrosesan, sedangkan citra yang terlalu kecil (lebar kurang dari 400 piksel) diperbesar agar karakter lebih mudah dikenali (Inal Achyar, Yasir and Satria, 2023).

2. Konversi Grayscale

Konversi grayscale mengubah citra berwarna (BGR) menjadi citra abu-abu satu kanal dengan menghitung rata-rata tertimbang dari komponen warna merah, hijau, dan biru. Konversi ini menyederhanakan data citra dan mengurangi kompleksitas komputasi pada tahap berikutnya, karena OCR pada dasarnya hanya membutuhkan informasi intensitas piksel, bukan informasi warna, berikut rumus (Pramudiya *et al.*, 2024).

$$Y = 0.299R + 0.587G + 0.114B$$

Keterangan:

R = nilai intensitas kanal warna merah (Red) piksel, rentang 0–255
G = nilai intensitas kanal warna hijau (Green) piksel, rentang 0–255
B = nilai intensitas kanal warna biru (Blue) piksel, rentang 0–255
Y = nilai intensitas piksel grayscale hasil konversi, rentang 0–255

3. Gaussian Blur (Reduksi Noise)

Gaussian blur diterapkan untuk mengurangi noise pada citra dengan menghaluskan variasi intensitas piksel yang tidak diinginkan. Kernel Gaussian berukuran 3x3 piksel digunakan dalam penelitian ini. Proses ini bekerja dengan

mengganti nilai setiap piksel dengan rata-rata tertimbang Gaussian dari piksel-piksel di sekitarnya, sehingga noise berfrekuensi tinggi dapat ditekan tanpa terlalu mengaburkan tepi karakter yang penting (Qonita *et al.*, 2026).

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Keterangan:

G(x,y) = nilai bobot kernel Gaussian pada koordinat (x, y)

σ (sigma) = standar deviasi yang mengatur tingkat penghalusan citra

x, y = jarak piksel dari titik pusat kernel

4. Binarisasi Otsu

Binarisasi Otsu adalah metode thresholding adaptif yang secara otomatis menentukan nilai ambang batas optimal untuk mengubah citra grayscale menjadi citra biner (hitam dan putih). Metode ini bekerja dengan memaksimalkan varian antar kelas (antara piksel latar depan dan latar belakang) sehingga pemisahan teks dari latar belakang menjadi lebih optimal (Sebatubun and Haryawan, 2024). Binarisasi Otsu sangat cocok untuk dokumen seperti KTM yang memiliki kontras tinggi antara teks dan latar belakang.

$$\sigma_B^2(t) = \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^{2t^*} = argmax_t^2 \sigma_B^2(t)$$

Keterangan:

σ^2_B(t) = variansi antar kelas pada nilai ambang t

$\omega_0(t)$, $\omega_1(t)$ = probabilitas piksel kelas latar belakang dan kelas objek pada nilai ambang t

$\mu_0(t)$, $\mu_1(t)$ = nilai rata-rata intensitas piksel masing-masing kelas

t^* = nilai ambang optimal yang dipilih (memaksimalkan σ^2_B)

2.5 OpenCV

OpenCV (Open Source Computer Vision Library) adalah pustaka open-source untuk pengolahan citra dan visi komputer yang dikembangkan oleh Intel. OpenCV menyediakan lebih dari 2500 algoritma yang dioptimalkan untuk berbagai tugas pengolahan citra termasuk filtering, transformasi geometris, segmentasi, dan deteksi fitur.

Dalam penelitian ini, OpenCV digunakan untuk seluruh tahap prapemrosesan citra, yaitu cv2.imread() untuk membaca citra, cv2.resize() untuk mengubah ukuran, cv2.cvtColor() untuk konversi grayscale, cv2.GaussianBlur() untuk reduksi noise, dan cv2.threshold() dengan flag THRESH_OTSU untuk binarisasi adaptif.

2.6 Regular Expression (Regex)

Regular Expression atau Regex adalah urutan karakter yang mendefinisikan pola pencarian dalam teks. Regex merupakan alat yang sangat kuat untuk pencarian, pencocokan, dan ekstraksi pola teks tertentu dari string yang lebih panjang (Yusuf et al., 2025). Dalam konteks penelitian ini, Regex digunakan untuk mengekstrak field-field data terstruktur dari teks mentah hasil OCR.

Python menyediakan modul re yang lengkap untuk operasi Regex. Fungsi re.search() digunakan untuk mencari pola pertama yang cocok dalam teks, sedangkan flag re.IGNORECASE memungkinkan pencocokan tanpa memperhatikan kapitalisasi. Pola Regex yang dirancang dalam penelitian ini

mempertimbangkan variasi format penulisan label pada KTM dari berbagai institusi, seperti variasi antara 'NIM', 'N.I.M', dan 'Nomor Induk Mahasiswa', serta variasi tanda pemisah seperti titik dua, tanda sama dengan, dan tanda hubung.

Tabel 2.1 berikut menunjukkan pola Regex yang dirancang untuk masing-masing field data KTM dalam penelitian ini.

Field NIM: Pola mendeteksi label 'NIM', 'N.I.M', atau 'Nomor Induk Mahasiswa' diikuti tanda pemisah opsional dan 8 hingga 14 digit angka.

Field Nama: Pola mendeteksi label 'NAMA', 'Nama', atau 'Name' diikuti nilai berupa string huruf alfabet minimal 4 karakter.

Field Program Studi: Pola mendeteksi label 'PROGRAM STUDI', 'Program Studi', 'PRODI', 'Prodi', 'JURUSAN', atau 'Jurusan' diikuti nilai hingga akhir baris.

Field Fakultas: Pola mendeteksi label 'FAKULTAS', 'Fakultas', 'Faculty', atau 'FAK' diikuti nilai hingga akhir baris.

Field TTL: Pola mendeteksi label 'Tempat/Tanggal Lahir', 'TTL', 'Tgl. Lahir', atau 'Lahir' diikuti nilai yang mengandung nama kota dan format tanggal.

2.7 Bahasa Pemprograman Python

Python adalah bahasa pemrograman tingkat tinggi yang bersifat *interpreted*, dinamis, dan mendukung berbagai paradigma pemrograman termasuk pemrograman berorientasi objek, prosedural, dan fungsional (Adawiyah Ritonga and Yahfizham Yahfizham, 2023). Python dipilih sebagai bahasa implementasi dalam penelitian ini karena ekosistem pustakanya yang luas untuk pengolahan citra (OpenCV), OCR (pytesseract), ekspresi reguler (modul *re* bawaan),

pembuatan antarmuka grafis (Tkinter), serta pengolahan berkas Excel (openpyxl). Sifat Python yang bersifat lintas platform juga memungkinkan aplikasi yang dibangun dapat dijalankan pada berbagai sistem operasi tanpa perubahan kode yang signifikan.

2.8 Tkinter

Tkinter merupakan pustaka antarmuka pengguna grafis (*Graphical User Interface*/GUI) standar bawaan Python yang menyediakan lapisan berorientasi objek di atas *toolkit* Tk. Karena statusnya sebagai pustaka bawaan, Tkinter tidak memerlukan instalasi tambahan dan menjadikannya pilihan yang praktis untuk membangun aplikasi desktop tanpa dependensi eksternal. Tkinter menyediakan berbagai *widget* siap pakai seperti label, tombol, kotak masukan teks, area teks, serta *notebook* untuk antarmuka bertab, yang seluruhnya dimanfaatkan dalam penelitian ini untuk membangun tiga tab fungsional yaitu Pengujian Satu Citra, Pengujian Seluruh Data, dan Confusion Matrix.

Struktur dasar aplikasi Tkinter terdiri atas jendela utama (*root window*) sebagai wadah, sekumpulan *widget* yang disusun menggunakan pengelola tata letak (*layout manager*) seperti *pack*, *grid*, atau *place*, serta *event loop* yang dijalankan melalui pemanggilan *mainloop()* untuk terus mendengarkan interaksi pengguna seperti klik tombol atau pemilihan berkas.

2.9 Evaluasi Kinerja Sistem (*Confusion Matrix)*

Confusion matrix adalah tabel yang digunakan untuk mengukur kinerja suatu sistem klasifikasi atau ekstraksi dengan membandingkan hasil prediksi sistem terhadap nilai sebenarnya (*ground truth*). Dalam konteks penelitian ini, *confusion matrix* diterapkan pada level per-*field* data (NPM, Nama, Program Studi, Fakultas, dan TTL) untuk mengetahui seberapa akurat hasil ekstraksi OCR dan Regex dibandingkan data referensi yang telah diverifikasi secara manual. Empat komponen utama *confusion matrix* yang digunakan adalah:

1. *True Positive* (TP) — nilai field berhasil diekstraksi dan sesuai dengan ground truth.
2. *False Positive* (FP) — nilai field berhasil diekstraksi namun tidak sesuai dengan ground truth (salah baca atau salah ekstrak).
3. *False Negative* (FN) — ground truth memiliki nilai namun sistem gagal mengekstraksinya (kosong).
4. *True Negative* (TN) — baik ground truth maupun hasil ekstraksi sama-sama kosong (field memang tidak tersedia pada kartu).

Berdasarkan keempat komponen tersebut, penelitian ini menghitung empat metrik

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

mengukur proporsi keseluruhan prediksi yang benar.

$$Precision = \frac{TP}{TP + FP}$$

mengukur seberapa tepat nilai yang diekstraksi sistem dibandingkan seluruh nilai yang diekstraksi.

$$Recall = \frac{TP}{TP + FN}$$

mengukur seberapa lengkap sistem berhasil mengekstraksi nilai yang seharusnya ada.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

rata-rata harmonik antara Precision dan Recall yang memberikan gambaran keseimbangan keduanya.

.