Dokumenten-Automatisierung-LLM

Dieses Buch beschreibt den Einsatz von Large Language Models zur intelligenten Verarbeitung und Automatisierung von Dokumenten-Workflows. Es behandelt praxisnahe Ansätze für Analyse, Extraktion, Klassifikation und Weiterverarbeitung von Dokumentinhalten.

OCR Vergleich

Lokale Open-Source-OCR-Werkzeuge im Vergleich

Stand: August 2026, basierend auf Quellen von Mai–Juni 2026 (max. ~3 Monate alt). Fokus auf Tools, die sich lokal/selbst hosten lassen (kein Cloud-Zwang). Vor Produktivsatz am eigenen Dokumentenbestand testen — jedes Benchmark ist auf akademischen/Standarddaten trainiert, reale Scans (Formulare, alte Archive, Faxqualität) sind eine andere Verteilung.

Kurzübersicht

Situation	Empfehlung
Digitale PDFs mit Textlayer	Docling oder direkt PyMuPDF/pdfplumber
Viele einfache Scans, keine GPU	Tesseract
Mehrsprachig, Tabellen, GPU vorhanden	PaddleOCR / PaddleOCR-VL
Komplexe Layouts → Markdown fürs RAG/LLM	MinerU
Handschrift, unregelmäßige Layouts	VLM-Ansatz (PaddleOCR-VL, Marker)
Lizenzprüfung durch Rechtsabteilung nötig	Docling (MIT) oder PaddleOCR (Apache-2.0) zuerst

---

Tesseract

Ansatz: Klassische OCR-Pipeline, Zeichen für Zeichen Lizenz: Apache-2.0 Hardware: CPU (kein GPU-Zwang, GPU/OpenCL experimentell und nicht produktionsreif)

Stärken: Extrem etabliert, läuft überall, kostenlos, hoher Durchsatz bei sauberem Drucktext, viele Sprachpakete. Schwächen: Schwach bei Handschrift, komplexen Layouts, Tabellen. Kein Strukturverständnis. Genauigkeit sinkt stark bei schlechten Scans oder Schräglage. Am besten für: Große Archive mit sauberem gedrucktem Text, Budget = 0, kein GPU-Server verfügbar.

EasyOCR

Ansatz: CNN+RNN-basierte klassische OCR Lizenz: Apache-2.0 Hardware: GPU empfohlen, CPU möglich

Stärken: Sehr schneller Einstieg (~5 Min Setup), gute Confidence-Scores, brauchbar bei gekrümmtem/fotografiertem Text. Schwächen: Schwächer bei Dokumentlayout und Tabellen als PaddleOCR. Am besten für: Schnelle Prototypen, Szenentext, einfache mehrsprachige Erkennung ohne Layoutanspruch.

PaddleOCR / PaddleOCR-VL

Ansatz: Modulare klassische Pipeline, plus VLM-Variante (PaddleOCR-VL 1.5+) für Layoutverständnis Lizenz: Apache-2.0 Hardware: GPU für Serverpräzision, CPU für Lightweight-Modelle

Stärken: 80+ Sprachen inkl. starkem CJK-Support, gute Tabellenerkennung (PP-Structure), hoher Durchsatz auf GPU (~120 Seiten/Min laut Herstellerangabe). PP-OCRv6 zeigt spürbaren Sprung ggü. Vorgänger. Schwächen: Mehr Konfigurationsaufwand als leichtere Libraries. Reine Basisversion ohne PP-Structure liefert nur Text, keine Struktur. Versions-Kompatibilität eng — ältere/neuere Kombinationen von paddlepaddle und paddleocr können sich gegenseitig brechen. Am besten für: Produktive mehrsprachige Pipelines mit Tabellen/Formularen und GPU-Zugriff.

Docling

Ansatz: Ensemble spezialisierter Modelle, einheitliche API Lizenz: MIT (permissiv) Hardware: CPU-fähig, dadurch langsamer als GPU-Alternativen

Stärken: Sehr sauberes, lesbares Markdown. Hervorragend bei digital-geborenen PDFs mit einfachem Layout. Deckt viele Formate ab (PDF, DOCX, PPTX, XLSX, HTML). Permissive Lizenz ohne Umsatzschwellen. Schwächen: Markdown-first-Design begrenzt Erhalt komplexer Layouts/strukturierter Daten. Schwächer bei Scans/Handschrift. Am besten für: Digitale PDFs, RAG-Ingestion, wenn Lizenzprüfung schnell gehen muss.

Marker

Ansatz: LLM-gestützte PDF-zu-Markdown-Pipeline Lizenz: Code Apache-2.0, Modell-Gewichte GPL-3.0 mit Umsatzschwelle (Hersteller nennt 5 Mio. USD) — Lizenzdatei der exakt gepinnten Version lesen Hardware: GPU für brauchbare Geschwindigkeit, CPU-Modus deutlich langsamer

Stärken: Gute Strukturerhaltung, LLM-gestützte Interpretation auch bei unsauberen Layouts. Schwächen: Ohne GPU langsam. Lizenzlage bei kommerziellem Einsatz separat prüfen, da Code- und Gewichte-Lizenz auseinanderfallen. Am besten für: Mittelgroße Dokumentenmengen mit GPU-Budget, wo Docling zu einfach strukturiert.

MinerU

Ansatz: Pipeline aus Layout-Erkennung, OCR (nutzt intern u. a. PaddleOCR-Modelle), Formel-/Tabellenerkennung, Zusammenbau in Lesereihenfolge Lizenz: Seit Version 3.1 (April 2026) MinerU Open Source License (Apache-2.0-Basis mit Zusatzbedingungen), davor AGPL-3.0 — der Wechsel erleichtert kommerzielle Nutzung deutlich Hardware: GPU/CUDA, NPU/CANN, MPS-Beschleunigung; CPU-Pipeline-Modus möglich, aber langsamer

Stärken: Sehr strukturiertes Markdown/JSON, gute Formel-(LaTeX)- und Tabellen-(HTML)-Erkennung, aktive Weiterentwicklung. Seit Version 3.4 (Juni 2026) PP-OCRv6 als Backend, laut Hersteller ~11% Genauigkeitsgewinn auf dem OmniDocBench-Leaderboard. Schwächen: Bei komplexen Layouts und Handschrift laut eigener Projekt-Dokumentation noch ausbaufähig.

Pipeline-Charakter (mehrere verkettete Modelle) macht Fehlersuche vielschichtiger als bei Einzelmodell-Tools. Am besten für: RAG-Ingestion-Layer vor Chunking/Embedding, wissenschaftliche/technische Dokumente mit Formeln.

Surya

Ansatz: VLM-nah — Layout, Leserichtung, Texterkennung, Tabellen in einem Modell Lizenz: Code Apache-2.0, Modell-Gewichte modifizierte OpenRAIL-M-Lizenz mit Umsatzschwelle (Hersteller nennt 5 Mio. USD) Hardware: GPU empfohlen

Stärken: Kompakt, gutes Kosten-Genauigkeits-Verhältnis in vergleichbaren Tests, 90+ Sprachen, layoutbewusste Ausgabe in einem Durchgang. Schwächen: Gewichte-Lizenz separat prüfen bei höherem Umsatz. Jünger und weniger etabliert als Tesseract/PaddleOCR. Am besten für: Layout-Analyse und OCR in einem Schritt, wenn die Lizenzschwelle unproblematisch ist.

GOT-OCR2

Ansatz: Kompaktes, spezialisiertes OCR-Modell (~580 M Parameter) Lizenz: Apache-2.0 Hardware: Ein einzelnes GPU-Modell ausreichend

Stärken: Gute Leistung bei Fließtext, Formeln, Tabellen. Sehr permissive Lizenz. Schwächen: Geringere Community-Adoption, weniger Tooling/Ökosystem als PaddleOCR oder MinerU. Am besten für: Schlanke Einzelmodell-Lösung ohne Pipeline-Overhead.

Nach Dokumenttyp

Dokumenttyp	1. Wahl	2. Wahl
Digitale PDFs (Verträge, exportierte Rechnungen)	Docling / PyMuPDF-Textlayer	—
Saubere Scans, gedruckter Text	PaddleOCR	Tesseract
Alte/verschlissene Scans	PaddleOCR-VL	MinerU
Formulare/Tabellen	PaddleOCR (PP-Structure)	MinerU
Technische Zeichnungen (A0–A2)	PaddleOCR-VL	Docling (bei vorhandenem Textlayer)
Wissenschaftliche Dokumente mit Formeln	MinerU	Marker
Handschrift	VLM-Ansatz (PaddleOCR-VL)	—
Mehrsprachig / CJK	PaddleOCR	Surya
RAG/LLM-Ingestion, gemischter Bestand	MinerU + Docling im Zwei-Stufen-Setup	—

Warum diese Reihenfolge: Bei digitalen PDFs kostet ein Textlayer-Zugriff Millisekunden statt Sekunden — kein OCR nötig. Bei Alt-Scans kompensiert der VLM-Ansatz Bildfehler besser als eine

klassische Pipeline; Vorverarbeitung (Kontrast, Entrauschen, Entzerren) davor ist trotzdem empfohlen. Bei Formeln/Tabellen zählt die dedizierte Erkennung (LaTeX/HTML-Ausgabe), nicht reine Texterkennung. Für gemischte Bestände hat sich in mehreren Quellen das Zwei-Stufen-Muster (günstiger Konverter zuerst, schwerer Parser nur bei Bedarf) als produktionsüblich herausgestellt.

Lizenz-Stolperfallen

Code- und Gewichte-Lizenz können auseinanderfallen. Bei manchen Anbietern (z. B. Marker, Surya) ist der Code Apache-2.0, die Modell-Gewichte laufen aber unter restriktiveren Lizenzen mit Umsatzschwellen. Eine Apache-Lizenz auf dem Repository sagt nichts über die Nutzbarkeit der Gewichte aus.

MinerU hat die Lizenz gewechselt: bis Version 3.0 AGPL-3.0 (Copyleft, für viele Firmen ein Ausschlusskriterium), seit Version 3.1 (April 2026) die MinerU Open Source License auf Apache-2.0-Basis mit Zusatzbedingungen — deutlich unternehmensfreundlicher, aber die genau eingesetzte Version sollte geprüft werden.

Immer die LICENSE-Datei der exakt gepinnten Version lesen, nicht nur einen Blogpost oder das Repository-Badge — Bedingungen ändern sich zwischen Minor-Versionen.

Hinweis zur Quellenlage

Diese Übersicht basiert auf öffentlich zugänglichen Vergleichsartikeln und Herstellerangaben von Mai bis Juni 2026. Konkrete Geschwindigkeits- und Genauigkeitsangaben (z. B. "~120 Seiten/Min", "~11% Genauigkeitsgewinn") stammen aus den jeweiligen Herstellerbenchmarks oder Testberichten Dritter und wurden nicht selbst nachgemessen — vor einer Entscheidung lohnt sich ein eigener Test mit repräsentativen Dokumenten aus dem eigenen Bestand.

Tool and Model Comparison

 Vollbild öffnen

https://trainer.ulrich-wiki.com/ocr-model-comparison-2026.html