Der vorliegende Artikel vergleicht die Qualität der automatischen Texterkennung aus dem Google-Books-Projekt mit Alternativen aus dem Open-Source-Umfeld: Tesseract, OCRopus, Calamari und Kraken. Für den Vergleich wurde ein Testset aus Digitalisaten von Drucken gebildet, die in den retrospektiven Nationalbibliografien VD16, VD17 und VD18 verzeichnet sind. Der zugrundeliegende Datenbestand ist in Hinblick auf Entstehungszeitraum, Sprache, Druck- und Vorlagenqualität sehr heterogen. Dies wirkt sich sowohl auf die Erstellung der Ground Truth als auch auf die automatische Auswertung aus. Die damit verbundenen Herausforderungen werden im Artikel beleuchtet und die Ergebnisse aus dem Vergleich vorgestellt. Als zentrales Ergebnis lässt sich festhalten, dass Google OCR, Tesseract und Calamari insgesamt sehr gute und im Vergleich mit den anderen OCR-Engines die besten Ergebnisse liefern.

This article compares the quality of automatic text recognition from the Google Books project with results from open source alternatives: Tesseract, OCRopus, Calamari and Kraken. For the comparison, a test set was created from digitised versions of printed publications listed in the retrospective national bibliographies VD16, VD17 and VD18. The underlying database is highly heterogeneous in terms of the period of origin, language and the quality of the original printed document. This affects both the production of the »ground truth« and the automatic evaluation. The article highlights the challenges involved and presents the results of the comparison. The main result is that Google OCR, Tesseract and Calamari deliver the best results in comparison with the other OCR engines, and very good results overall.

JOHANNES BAITER, MARCUS BITZL, SEBASTIAN MANGOLD, KATHARINA SCHMID

Google Books vs. Open Source

Ein Vergleich der OCR-Qualität auf Basis von Drucken aus VD16, VD17 und VD18

Einleitung

Bei der Digitalisierung von gedruckten und handschriftlichen Dokumenten spielt die automatisierte Erkennung von Textinformation eine zentrale Rolle. Das reine Ablichten beispielsweise eines Buches durch Scannen oder Fotografieren und das anschließende Speichern in einem digitalen Bildformat führt noch nicht zu einem digitalen Text. Das Lesen des Textes bleibt weiterhin dem Menschen überlassen, nicht anders als beim Original. Erst die Erkennung und Überführung dieser Textinformation in maschinenlesbare Form, die sogenannte optische Zeichenerkennung (engl. Optical Character Recognition, kurz OCR) erweitert die Nutzungsmöglichkeiten des Digitalisats enorm, hin zu Volltextsuche, Eigennamenerkennung (engl. Named-entity Recognition, kurz NER), automatischer Textanalyse, Training von Sprachmodellen und nicht zuletzt zu mehr Barrierefreiheit durch mögliche Umwandlung in hörbaren Text, um nur einige Beispiele zu nennen.

Die Bayerische Staatsbibliothek (BSB) digitalisiert in Kooperation mit Google seit 2007 ihren Altbestand an Druckwerken. Bis Oktober 2023 wurden knapp 2,7 Millionen Objekte auf diesem Wege digital verfügbar gemacht. Die Digitalisierung durch Google umfasst immer auch die Texterkennung. Google entwickelt seine OCR-Programme kontinuierlich weiter und stellt den Bibliothekspartnern die jeweils neueste Version des Ergebnisses zur Verfügung. Auf diese Weise erhalten auch Objekte mit länger zurückliegender Bilddigitalisierung fortwährend Updates hinsichtlich des erkannten Textes. Das ist in Hinblick auf die Qualitätssteigerung aufgrund der stetigen technologischen Weiterentwicklung in den vergangenen Jahren ein unschätzbarer Vorteil.

Auch im Open-Source-Bereich wurden Fortschritte erzielt und OCR-Programme weiter- oder neuentwickelt. Die DFG förderte mit OCR-D ein Projekt, »dessen Hauptziel die konzeptionelle und technische Vorbereitung der Volltexttransformation der VD ist.«[1] Dazu wurden Open-Source-Entwicklungen gebündelt, ergänzt und weiterentwickelt sowie Workflow-Implementierungen vorgenommen.

Ziel des vorliegenden Beitrags ist es, die Qualität der automatischen Texterkennung aus dem Google-Books-Projekt mit Alternativen aus dem Open-Source-Umfeld zu vergleichen. Für den Vergleich wurde ein Testset aus Digitalisaten von Drucken gebildet, die in den retrospektiven Nationalbibliographien VD16, VD17 und VD18[2] verzeichnet sind. Aussagen in diesem Artikel beziehen sich ausschließlich auf Drucke aus dem Bereich VD16, VD17 und VD18 und lassen sich nur unter Vorbehalt auf die Erkennungsqualität anderer Drucke über-

tragen. Um die Qualität der OCR beurteilen zu können, wurde eine möglichst fehlerfreie Transkription des zu erkennenden gedruckten Textes erstellt. Mit dieser Musterlösung, der sogenannten Ground Truth, wurden die verschiedenen OCR-Ergebnisse verglichen.

Der Prozess, um von einer gescannten Buchseite zu ihrem Text in maschinenlesbarer Form zu kommen, umfasst neben der reinen Texterkennung mit einer OCR-Engine eine Reihe weiterer Verarbeitungsschritte. Dazu gehören die Optimierung des zu »lesenden« Bildes, die Erkennung des Layouts darauf und der einzelnen Zeilen sowie eventuell eine Nachkorrektur. Bei der vorliegenden Evaluierung liegt der Fokus allein auf der zeilenbasierten Zeichenerkennung mit OCR-Engines. Eine Bewertung aller weiteren Komponenten des OCR-Prozesses war nicht Gegenstand dieser Untersuchung. Als Ausgangspunkt für die Evaluierung wurden pragmatisch die von Google erkannten Zeilenbilder festgelegt. Dies ermöglicht einen praktikablen Vergleich der Texterkennung verschiedener OCR-Programme.

Datensatz und Ground Truth

Objekt- und Seitenauswahl

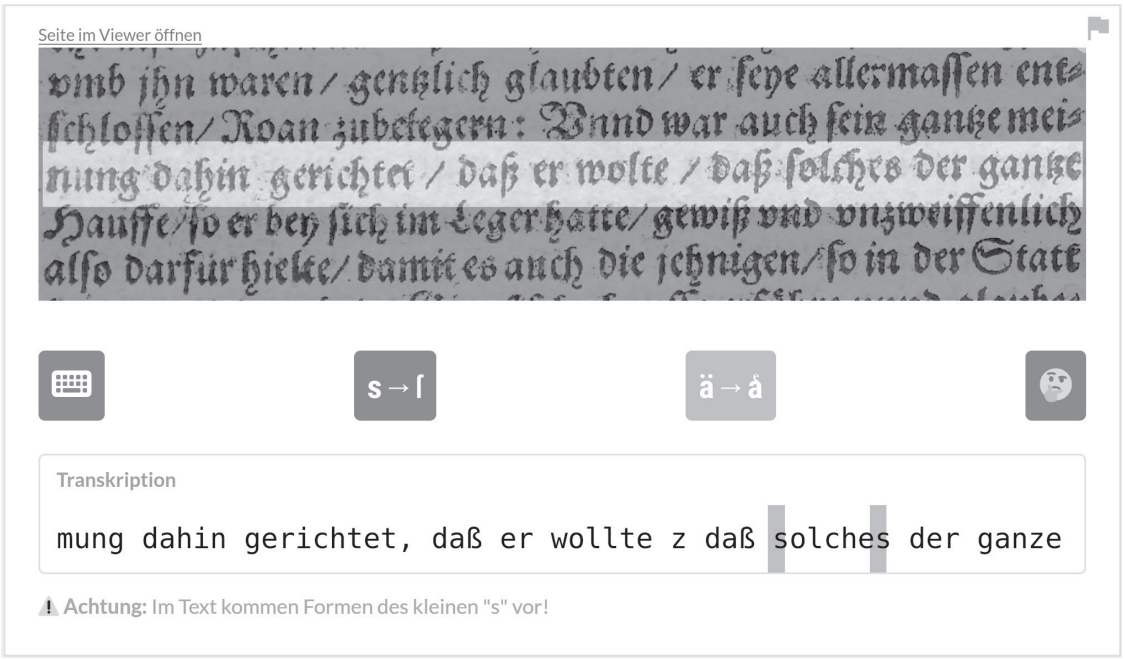
Für das Testset wurden 92 Objekte ausgewählt. Jedes Jahrzehnt des Bereichs VD16–18 sollte mit mindestens einem Objekt vertreten sein. Für die intellektuelle Erstellung der Ground-Truth-Daten wurde aus diesen Objekten je eine Seite nach dem Zufallsprinzip ausgewählt. Auf diese Weise sollten subjektive Auswahlkriterien minimiert werden. Die Auswahl der Objekte wurde gemäß folgender Kriterien getroffen:

- Digitalisate repräsentieren Objekte aus den Verzeichnissen der Drucke des 16., 17. und 18. Jahrhunderts, weisen also eine VD-Nummer als Identifikator auf,
- Objekte sind von Google digitalisiert und entsprechend mit OCR-Daten von Google versehen,
- Objekte sind möglichst gleichmäßig über den Zeitraum 1501–1800 verteilt,
- die Sprachauswahl ist möglichst repräsentativ bzgl. der in den VDs dokumentierten Objekte (ganz überwiegend Latein und Deutsch, etwas Griechisch, Hebräisch, Italienisch, Französisch),
- zahlenmäßig herausragende Autoren, z. B. Luther, sind berücksichtigt.

Ground-Truth-Erstellung

Um die Zeichenerkennungsqualität möglichst unabhängig von der Qualität der Layouterkennung zu messen, wurde die Google OCR als Segmentierungs-Quelle festgelegt. Folglich sind die von der Google OCR erkannten Zeilen Grundlage der Ground Truth.

Ground-Truth-Daten werden üblicherweise ausschließlich intellektuell erzeugt. Eine dabei häufig verwendete, akzeptierte Methode für die Gewinnung von Ground-Truth-Textdaten ist das sogenannte Double-Keying-Verfahren.[3] Dabei wird die Transkription der Vorlage jeweils von zwei unterschiedlichen Teams erstellt und anschließend abweichende Textstellen der beiden Versionen überprüft.



1 Tool zur OCR-Korrektur
Abb.: BSB / MDZ

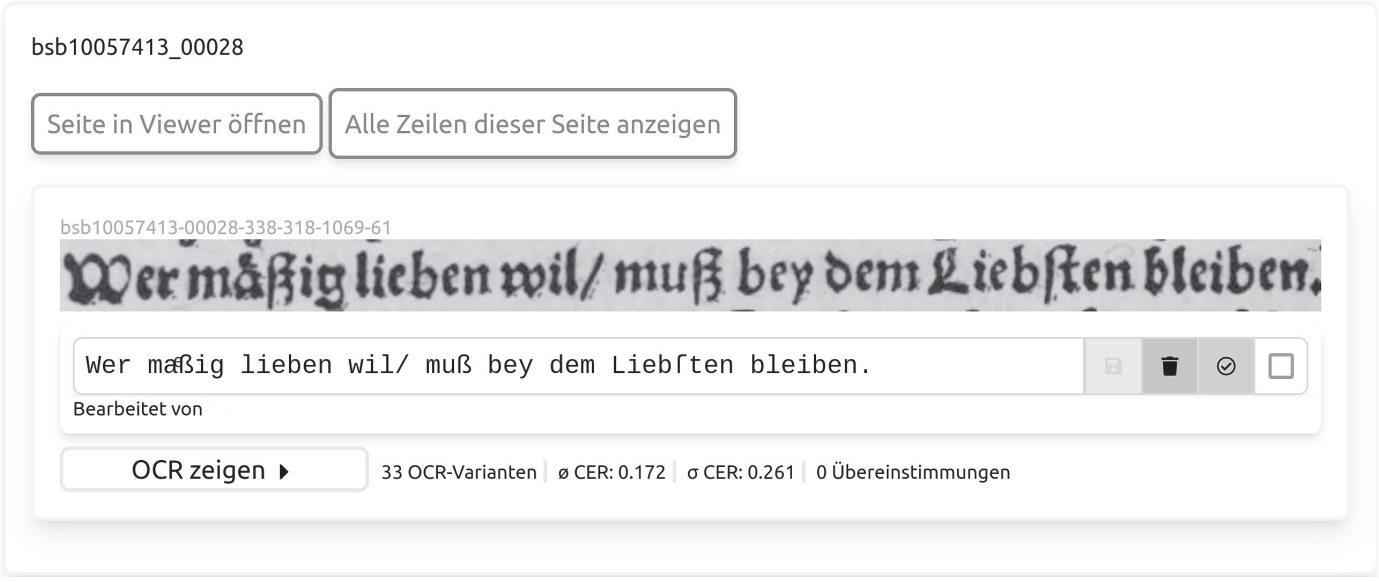
Da dieses Verfahren äußerst zeitaufwendig ist, kam für die vorliegende Evaluation ein alternatives Vorgehen zum Einsatz: Die Erstellung der Ground-Truth-Daten für die Zeilen der 92 ausgewählten Seiten wurde auf mehrere Personen aufgeteilt. Kolleg*innen mit Lesefähigkeit im Bereich alter Drucke sowie Spezialist*innen für Griechisch und Hebräisch erstellten die Daten.[4] Um die Erstellung zu vereinfachen und zu beschleunigen, korrigierten die beteiligten Mitarbeitenden die vorhandene Google OCR, anstatt von Grund auf zu transkribieren. Mit dieser Optimierung wurde in Kauf genommen, dass die entstandene Ground Truth möglicherweise zugunsten von Google OCR beeinflusst ist. Um die Mitarbeitenden zu einer möglichst gleichförmigen Korrektur anzuleiten, wurden sie in die Funktionsweise des Transkriptionstools und in die Transkriptionsrichtlinien eingeführt, die sich am OCR-D Transkriptionslevel 2[5] orientieren. Außerdem standen die Autor*innen dieses Beitrags durchgängig für Rückfragen zur Verfügung.

Für die Korrektur der Google OCR wurde ein Tool auf Basis der Software Prodigy[6] entwickelt. Das Tool ist unter einer Open-Source-Lizenz auf dem GitHub-Auftritt des Münchener Digitalisierungszentrums (MDZ) der Bayerischen Staatsbibliothek veröffentlicht.[7] Damit konnten mehrere Anwender*innen gleichzeitig Zeilen korrigieren, wobei jede Zeile nur einer Person zur Korrektur vorgelegt wurde. Zur größtmöglichen Vermeidung von Transkriptions-/Korrekturfehlern wurden automatisch typische, häufig vorkommende potenzielle OCR-Fehler farbig hervorgehoben (z. B. s statt ſ; s. Abb. 1).

Anders als bei modernen, industriell hergestellten Drucken stellt die Übertragung der Zeichenvorlage in alten Drucken eine Herausforderung dar. So kann eine Textstelle aufgrund von Druckqualität, Schadstellen etc. nicht eindeutig lesbar sein, oder den in der historischen Vorlage verwendeten Zeichen fehlt eine Entsprechung im modernen Zeichensatz. Gerade bei alten Drucken stößt man immer wieder auf Grenzfälle, bei denen eine eindeutige Entscheidung schwierig bis unmöglich ist. In derartigen Fällen konnten die Transkribierenden einzelne Abschnitte als »unsicher« markieren[8] und so von der Bewertung ausschließen.

Im Bereich Griechisch und Hebräisch war die Unterstützung von Spezialist*innen aus den jeweiligen Fachabteilungen unabdingbar. Für die Transkription von Frühdrucken in Griechisch sind vertiefte Kenntnisse der Sprache und der Sonderzeichen in diesen Drucken und klare (auch fachlich fundierte) Vorgaben zu dem zu erzielenden Ergebnis bzw. den zu korrigierenden Fehlern nötig. Für manche Druckzeichen gibt es zudem keine Unicode-Entsprechung. In diesen Fällen wurden nicht für alle im Testset vorgesehenen Zeilen Ground-Truth-Daten erzeugt.

Um eine mögliche Beeinflussung zugunsten der Google OCR zu minimieren, wurden die Ground-Truth-Daten im Anschluss einer zusätzlichen, technisch basierten Prüfung unterzogen, bei der die Ergebnisse der anderen OCR-Engines mit einbezogen wurden. Dazu wurde ein eigenes Tool entwickelt, mit dem die korrigierten Google OCR-Daten mit allen anderen OCR-



2 Review-Tool zur Nachkorrektur der Ground Truth
Abb.: BSB / MDZ

Daten verglichen wurden. Neben der Revision Zeile für Zeile konnte man damit insbesondere gezielt nach Unterschieden zwischen OCR-Daten und korrigierten Daten filtern, um so leichter und schneller eventuell falsch korrigierte Stellen oder übersehene Fehler zu finden.

Überblick zur Texterkennung mit verschiedenen OCR-Engines

Gegenstand der Evaluierung waren neben der Google-eigenen Closed-Source-OCR die Open-Source-OCR-Engines Tesseract, OCRopus, Kraken und Calamari:

Tesseract[9] wurde 2005 als quelloffene OCR-Engine veröffentlicht. In Version 4.0.0 wurde die Texterkennung grundlegend überarbeitet und basiert nun auf tiefen neuronalen Netzwerken. Tesseract kann sowohl als Kommandozeilen-Werkzeug als auch als Softwarebibliothek in den OCR-Workflow eingebunden werden.

OCRopus[10] ist eine Sammlung von Kommandozeilen-Werkzeugen zur automatischen Texterkennung. In seiner ursprünglichen Version, die ab 2007 entstanden ist, nutzt OCRopus ein flaches neuronales Netzwerk für die Texterkennung. In der vorliegenden Evaluierung wurde eine Weiterentwicklung dieser ersten Version eingesetzt, die Python 3 unterstützt und weitere Verbesserungen bei der Bildvorverarbeitung bietet.[11]

Die OCR-Engine *Kraken*[12] ist speziell für nicht-lateinische Schriftarten optimiert. Wie Tesseract kann Kraken sowohl als Kommandozeilen-Werkzeug als auch als Softwarebibliothek in den OCR-Workflow eingebunden werden. In seiner aktuellen Version (4.3.12) implementiert Kraken tiefe neuronale Netzwerke und bietet GPU-Unterstützung, um insbesondere das Trainieren von Netzwerkmodellen zu beschleunigen.

Calamari[13] ist die jüngste[14] der eingesetzten OCR-Engines. Im Unterschied zu den anderen Engines unterstützt Calamari lediglich die Texterkennung auf der Grundlage von Zeilenbildern, vorhergehende Arbeitsschritte wie Layouterkennung und Zeilensegmentierung sind nicht implementiert. Calamari kann sowohl als Kommandozeilen-Werkzeug als auch als Python-Modul in einen umfassenderen OCR-Workflow eingebunden werden. Die OCR-Engine implementiert tiefe neuronale Netzwerke und bietet wie Kraken GPU-Unterstützung.

Für die Evaluation wurde ein eigenes Kommandozeilen-Werkzeug in Python entwickelt, welches Tesseract, OCRopus und Kraken nutzt, um den Text einzelner Zeilen zu erkennen. Texterkennung mit der OCR-Engine Calamari wurde in einem separaten Python-Skript implementiert, da die benötigten Softwarebibliotheken nicht miteinander kompatibel waren.

Die konkrete Implementierung der Vorverarbeitung wurde aus den Softwarebibliotheken der jeweiligen Engines übernommen. Für Calamari wurde die Vorverarbeitung von OCRopus verwendet, da die meisten Calamari-Modelle auf Basis der OCRopus Vorverarbeitung trainiert wurden.

Layoutanalyse und Segmentierung als zusätzliche Verarbeitungsschritte entfallen, da der Evaluationsdatensatz bereits aus vorsegmentierten Zeilen besteht. Googles Layoutanalyse liefert rechteckig segmentierte Zeilen, die Artefakte anderer Zeilen enthalten können. Engines wie Tesseract, Calamari und Kraken können auf der Basis dieser vergleichsweise groben Segmentierung die Zeichenerkennung durchführen. Nur OCRopus erwartet genau umgrenzte Zeilen in Polygonform als Eingabe für die Zeichenerkennung. OCRopus bietet eine Resegmentierung von Zeilen an, um eine grobe Segmentierung zu verfeinern und Artefakte anderer Zeilen zu entfernen. Da OCRopus jedoch keine Schnittstelle für die Einbindung in andere Programme bietet und der Einsatz im vorliegenden Evaluationsframework nur mit erheblichem Mehraufwand möglich gewesen wäre, wurde im Rahmen dieser Auswertung auf die Resegmentierung verzichtet.

Alle ausgewerteten OCR-Engines unterstützen eine Vielzahl von sogenannten Modellen. Ein solches Modell ist ein durch Verfahren des maschinellen Lernens trainiertes neuronales Netz, das die eigentliche Texterkennung durchführt. Das Hauptunterscheidungsmerkmal zwischen den verschiedenen Modellen ist die für das Training verwendete Datenbasis, die maßgeblich die Erkennungsqualität beeinflusst. Aus den verfügbaren Modellen für die verschiedenen OCR-Engines wurden diejenigen ausgewählt, welche den Eigenschaften des Evaluationsdatensatzes am besten entsprechen. Insbesondere wurden Modelle berücksichtigt, die neben Texten in Antiqua auch mit Texten in Frakturschrift trainiert wurden, und solche, welche die verschiedenen Zeichensätze aus dem Evaluationsdatensatz (Griechisch, Hebräisch und Lateinisch) abdecken.

Wenn möglich, wurden bei der Texterkennung mehrere Modelle miteinander kombiniert, um ihr »Spezial-

3 Beispiel für ein von Google segmentiertes Zeilenbild mit deutlichen Artefakten der vorausgehenden und nachfolgenden Zeile
Abb.: BSB / MDZ

wissen« zu bündeln und ein möglichst breites Spektrum an Schriftarten, Zeichensätzen und Sprachen zu berücksichtigen. Grundsätzlich unterstützen die Engines Tesseract, Calamari und Kraken den gleichzeitigen Einsatz von mehreren Modellen für die Texterkennung. Kraken wählt aus einer Menge von Modellen das beste Modell abhängig davon aus, welcher Zeichensatz in der jeweiligen Zeile erkannt wurde. Die automatische Erkennung des Zeichensatzes war in einer früheren Version von Kraken implementiert, ist aktuell jedoch nicht funktionsfähig. Aus diesem Grund kamen für Kraken keine Modellkombinationen zum Einsatz. Calamari kombiniert verschiedene Modelle, indem die Engine mehrere Modelle auf die Daten anwendet, die Ergebnisse vergleicht und anhand von Wahrscheinlichkeit gewichtet, um so die beste Texterkennung zu ermitteln (Confidence Voting).[15]

Verwendete Methoden bei der Evaluation

Die Evaluierung der erzeugten OCR-Ergebnisse erfolgte in einem mehrstufigen Prozess. Zunächst wurden sowohl Ground Truth als auch OCR-Ergebnisse in eine einheitliche Unicode-Repräsentation überführt. Anschließend erfolgte ein weiterer Normalisierungsschritt, bei dem verschiedene Textmerkmale in eine einheitliche Form umgewandelt wurden. Im vorliegenden Projekt wurden drei verschiedene Normalisierungsstufen definiert:

- **minimal:** Keine weitere Normalisierung; möglicher Use Case: Edition, buch- und druckwissenschaftliche Fragestellungen.
- **mittel:** Normalisierung auf OCR-D Level 2 Transkriptionsrichtlinien.[16] Normalisierung von Leerzeichen um Satzzeichen herum: Kein Leerzeichen vor Satzzeichen, genau eines danach. Use Case: Detailgetreue Darstellung nah am Original in Viewer-Anwendungen, die leicht kopiert werden kann.
- **hoch:** Normalisierung auf OCR-D Level 1 Transkriptionsrichtlinien.[17] Wie Level 2, aber zusätzlich werden alte Zeichenformen (langes S, alte Umlautformen) in moderne Äquivalente umgewandelt und verschiedene Trennzeichen auf die geläufige moderne Form vereinheitlicht. Use Case: Indexierung umfangreicher, heterogener Korpora für Volltextsuche.

Im Umfeld großer digitaler Bestände dürfte die hohe Normalisierung den meisten Anforderungen genügen. Je höher die Erwartung an die Wiedergabe spezifischer Druckeigenheiten ist, desto aufwendiger gestaltet sich die automatische Erkennung.

Nach der Normalisierung wurden die Fehlerraten der Texterkennungs-Ergebnisse ermittelt. Hierzu wurde mittels des häufig verwendeten Levenshtein-Algorithmus[18] der Editierabstand zwischen Ground Truth und OCR-Ergebnis bestimmt, d. h. die Anzahl der Operationen (Einsetzen, Ersetzen, Löschen), die notwendig wären, um das Ergebnis identisch zur Ground Truth zu machen. Editieroperationen, die Abschnitte aus der Ground Truth betreffen, die als unsicher markiert wurden, wurden dabei nicht gezählt. Die daraus resultierende Zahl wurde durch die Anzahl der Zeichen im Ground-Truth-Text geteilt, um so den Fehleranteil ausgedrückt in Prozent zu erhalten.

Dieses Verfahren wurde einmal auf Zeichenebene durchgeführt, um die sogenannte *Character Error Rate (CER)* zu ermitteln, d. h. den Anteil an falsch oder nicht erkannten Zeichen im Verhältnis zur Ground Truth. Weiterhin wurde das Verfahren auch auf Wort-Ebene durchgeführt, um die sogenannte *Word Error Rate (WER)* zu ermitteln. Hierbei wurde der Zeilentext vor dem Vergleich nach Unicode-Regeln in Wörter aufgeteilt. Ein Wort wurde als Fehler gezählt, wenn mindestens ein Zeichen im Wort fehlerhaft war.

Beispiel

Ground Truth	Hallo Welt
OCR-Resultat	Hall**e** W**o**lt
Editier-Distanz	2
Zeichenanzahl Ground Truth	10
Wortanzahl Ground Truth	2
Character Error Rate	2 / 10 = 20 %
Word Error Rate	2 / 2 = 100 %

Die Ermittlung der OCR-Qualität über Levenshtein-Distanz und Zeichen- bzw. Wortfehlerrate ist im wissenschaftlichen Kontext etabliert.[19] Das Vorgehen entspricht dem des Evaluationstools dinglehopper,[20] das auch bei OCR-D zur Bewertung der OCR-Qualität eingesetzt wird.

Ergebnisse

Jede Zeile aus dem Datensatz wurde mit der gewählten Kombination aus OCR-Engine und Modell in Text konvertiert. Sowohl der erkannte Text wie auch der Ground-Truth-Text aus dem Datensatz wurden für jede der drei oben beschriebenen Normalisierungsarten normalisiert. Die Berechnung der Fehlerraten für den Gesamt-Datensatz erfolgte durch Aufsummierung der individuellen Editierdistanzen und Teilen durch die Summe der Zeichen- und Wortanzahl aller Ground-Truth-Texte. Die so ermittelten Fehlerraten werten dadurch jeden Fehler gleich, unabhängig von der Länge der jeweiligen Zeile.

Die Tabellen auf den folgenden Seiten vergleichen die Qualität der Open-Source-OCR-Engines Tesseract, OCRopus, Kraken und Calamari. Die dunkelgrau hinterlegten Werte weisen das beste Ergebnis unter den Modellen der jeweiligen Open-Source-Engine aus:

♠ Bestes Ergebnis unter allen Modellen der jeweiligen Tabelle (Google OCR + Open-Source-Engine).

✶ Besseres Ergebnis als Google OCR.

Tesseract
Ausgewertete Modelle

- *deu:* Tesseract 5 Standardmodell für deutsche Sprache
- *frk:* Tesseract 5 Standardmodell für Fraktur
- *grc:* Tesseract 5 Standardmodell für Griechisch
- *heb:* Tesseract 5 Standardmodell für Hebräisch
- *gt4histocr*[21] *(-best* und *-fast):* Fraktur-Modell der UB Mannheim, trainiert auf Daten aus dem Deutschen Textarchiv
- *frak2021*[22] *(-best* und *-fast):* Fraktur-Modell der UB Mannheim, trainiert auf Daten des Deutschen Textarchivs, hauseigenen Transkriptionen der UB Mannheim und Daten aus dem »Austrian Newspapers« Datensatz des Library Labs der Österreichischen Nationalbibliothek

Die *best*- und *fast*-Varianten der *gt4histocr* und *frak2021* Modelle unterscheiden sich in ihrer Parameteranzahl, das *fast*-Modell ist jeweils schneller in der Auswertung und etwas ungenauer.

Vergleich Google OCR vs. Tesseract: Normalisierungsgrad **minimal**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**6,12 %** ♠	**29,34 %**
deu	15,82 %	56,30 %
frk	12,82 %	43,39 %
gt4histocr-fast	11,44 %	33,38 %
gt4histocr-best	15,36 %	39,72 %
frak2021-fast	6,96 %	22,90 % ✶
frak2021-best	**6,73 %**	**22,37 %** ✶ ♠
gt4histocr-fast, frk, deu, heb, grc	10,04 %	32,11 %
gt4histocr-best, frk, deu, heb, grc	11,22 %	35,00 %
frak2021-fast, frk, deu, heb, grc	7,29 %	24,25 % ✶
frak2021-best, frk, deu, heb, grc	7,72 %	25,57 % ✶
frak2021-fast, gt4histocr-fast, frk, deu, heb, grc	7,02 %	23,38 % ✶
frak2021-best, gt4histocr-best, frk, deu, heb, grc	7,61 %	24,95 % ✶

Vergleich Google OCR vs. Tesseract: Normalisierungsgrad **mittel**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**5,90 %** ♠	**28,80 %**
deu	15,51 %	54,73 %
frk	12,55 %	42,17 %
gt4histocr-fast	10,86 %	32,72 %
gt4histocr-best	14,99 %	39,21 %
frak2021-fast	6,31 %	22,06 % ✶
frak2021-best	**6,13 %**	**21,75 %** ✶ ♠
gt4histocr-fast, frk, deu, heb, grc	9,53 %	31,12 %
gt4histocr-best, frk, deu, heb, grc	10,79 %	34,22 %
frak2021-fast, frk, deu, heb, grc	6,65 %	23,35 % ✶
frak2021-best, frk, deu, heb, grc	7,20 %	24,78 % ✶
frak2021-fast, gt4histocr-fast, frk, deu, heb, grc	6,37 %	22,49 % ✶
frak2021-best, gt4histocr-best, frk, deu, heb, grc	7,08 %	24,20 % ✶

Vergleich Google OCR vs. Tesseract: Normalisierungsgrad **hoch**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**4,13 %** ♠	**20,26 %**
deu	14,87 %	53,12 %
frk	11,96 %	39,93 %
gt4histocr-fast	10,16 %	29,80 %
gt4histocr-best	14,52 %	37,63 %
frak2021-fast	5,93 %	19,98 % ✶
frak2021-best	**5,77 %**	**19,70 %** ✶ ♠
gt4histocr-fast, frk, deu, heb, grc	8,89 %	28,23 %
gt4histocr-best, frk, deu, heb, grc	10,25 %	31,80 %
frak2021-fast, frk, deu, heb, grc	6,19 %	20,77 %
frak2021-best, frk, deu, heb, grc	6,72 %	22,18 %
frak2021-fast, gt4histocr-fast, frk, deu, heb, grc	5,90 %	19,87 % ✶
frak2021-best, gt4histocr-best, frk, deu, heb, grc	6,61 %	21,62 %

Calamari

Ausgewertete Modelle[23]

- *gt4histocr:* trainiert auf dem gesamten GT4HistOCR Korpus
- *antiqua_historical:* Antiqua-Modell, trainiert auf Untermenge des GT4HistOCR Korpus
- *fraktur_historical*: Fraktur-Modell, trainiert auf Untermenge des GT4HistOCR Korpus
- *fraktur_19th_century:* trainiert auf einer Untermenge des GT4HistOCR Korpus (DTA19) und weiteren frei verfügbaren Datensätzen in Frakturschrift aus dem 19. Jahrhundert
- *deep3_lsh4:*[24] Modell mit tieferer Netzwerkstruktur, trainiert auf dem GT4HistOCR Korpus, französischen Antiquadrucken, dem OCR-D Korpus, frei verfügbaren Datensätzen in Frakturschrift aus dem 19. Jahrhundert und Daten aus verschiedenen weiteren Transkriptionsprojekten
- *deep3_antiqua_hist:* Antiqua-Modell auf der Basis von deep3_lsh4
- *deep3_antiqua-15-16-cent:* Antiqua-Modell auf der Basis von deep3_lsh4 mit Fokus auf dem 15. und 16. Jahrhundert
- *deep3_fraktur-hist:* Fraktur-Modell auf der Basis von deep3_lsh4
- *deep3_fraktur19:* Fraktur-Modell auf der Basis von deep3_lsh4 mit Fokus auf dem 19. Jahrhundert

Vergleich Google OCR vs. Calamari: Normalisierungsgrad **minimal**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**6,12%**	**29,34%**
antiqua_historical	18,93 %	54,77 %
deep3_antiqua-hist	10,22 %	40,65 %
deep3_antiqua-15-16-cent	8,77 %	37,09 %
deep3_fraktur-hist	5,82 % ✶	22,92 % ✶
deep3_fraktur19	9,17 %	31,76 %
deep3_lsh4	**4,67 % ✶ ♠**	**19,72 % ✶ ♠**
fraktur_historical	26,64 %	62,44 %
fraktur_19th_century	20,54 %	56,34 %
gt4histocr	6,76 %	22,71 % ✶
antiqua_historical, fraktur_historical	17,30 %	44,45 %
deep3_antiqua-15-16-cent, deep3_antiqua-hist, deep3_fraktur-hist, deep3_fraktur19	5,57 % ✶	21,10 % ✶

Vergleich Google OCR vs. Calamari: Normalisierungsgrad **mittel**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**5,90%**	**28,80%**
antiqua_historical	18,70 %	53,66 %
deep3_antiqua-hist	9,72 %	39,47 %
deep3_antiqua-15-16-cent	8,25 %	35,89 %
deep3_fraktur-hist	5,23 % ✶	21,99 % ✶
deep3_fraktur19	8,73 %	31,11 %
deep3_lsh4	**4,04 % ✶ ♠**	**18,59 % ✶ ♠**
fraktur_historical	26,64 %	61,72 %
fraktur_19th_century	20,40 %	56,29 %
gt4histocr	6,16 %	21,77 % ✶
antiqua_historical, fraktur_historical	17,06 %	43,60 %
deep3_antiqua-15-16-cent, deep3_antiqua-hist, deep3_fraktur-hist, deep3_fraktur19	5,06 % ✶	20,06 % ✶

Vergleich Google OCR vs. Calamari: Normalisierungsgrad **hoch**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**4,13%**	**20,26%**
antiqua_historical	18,32 %	52,17 %
deep3_antiqua-hist	9,22 %	38,21 %
deep3_antiqua-15-16-cent	7,79 %	34,46 %
deep3_fraktur-hist	4,27 %	17,76 % ✶
deep3_fraktur19	7,57 %	27,83 %
deep3_lsh4	**3,13 % ✶ ♠**	**14,63 % ✶ ♠**
fraktur_historical	26,24 %	60,29 %
fraktur_19th_century	19,84 %	55,12 %
gt4histocr	5,82 %	19,70 % ✶
antiqua_historical, fraktur_historical	16,73 %	41,85 %
deep3_antiqua-15-16-cent, deep3_antiqua-hist, deep3_fraktur-hist, deep3_fraktur19	4,18 %	16,47 % ✶

Kraken

Ausgewertete Modelle

- *austriannewspapers:*[25] trainiert auf österreichischen Zeitungen aus dem 19. und frühen 20. Jahrhundert
- *fraktur_all_2:*[26] trainiert auf österreichischen, schweizerischen und schwedischen Zeitungen aus den Sammlungen der Universitätsbibliotheken Mannheim und Göttingen und dem OCR-D Korpus

Vergleich Google OCR vs. Kraken: Normalisierungsgrad **minimal**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**6,12% ♠**	**29,34% ♠**
austriannewspapers	21,14 %	59,43 %
fraktur_all_2	**15,15%**	**48,82 %**

Vergleich Google OCR vs. Kraken: Normalisierungsgrad **mittel**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**5,90% ♠**	**28,80% ♠**
austriannewspapers	20,90 %	59,08 %
fraktur_all_2	**14,92%**	**48,40%**

Vergleich Google OCR vs. Kraken: Normalisierungsgrad **hoch**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**4,13% ♠**	**20,26% ♠**
austriannewspapers	20,50 %	57,63 %
fraktur_all_2	**12,73%**	**39,04%**

OCRopus

Ausgewertete Modelle

- *en-default*: Standardmodell
- *fraktur:* Standardmodell für Fraktur
- *LatinHist:*[27] trainiert auf 12 lateinischen Büchern in Antiquaschrift aus der Zeit zwischen 1471 und 1686
- *fraktur-jze:*[28] trainiert auf Texten in Frakturschrift aus dem 19. und frühen 20. Jahrhundert

Vergleich Google OCR vs. OCRopus: Normalisierungsgrad **minimal**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**6,12% ♠**	**29,34% ♠**
LatinHist	**25,85%**	**66,30%**
en-default	41,90 %	86,34 %
fraktur	38,51 %	81,85 %
fraktur-jze	44,44 %	88,30 %

Vergleich Google OCR vs. OCRopus: Normalisierungsgrad **mittel**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**5,90% ♠**	**28,80% ♠**
LatinHist	**25,69%**	**65,61%**
en-default	41,57 %	85,78 %
fraktur	38,35 %	81,69 %
fraktur-jze	43,99 %	88,16 %

Vergleich Google OCR vs. OCRopus: Normalisierungsgrad **hoch**

Modell(e)	Character Error Rate (weniger ist besser)	Word Error Rate (weniger ist besser)
Google OCR	**4,13% ♠**	**20,26% ♠**
LatinHist	**25,33%**	**64,60%**
en-default	41,35 %	85,59 %
fraktur	36,28 %	78,06 %
fraktur-jze	43,23 %	87,46 %

Zusammenfassung, Bewertung und Kontextualisierung der Ergebnisse

Google OCR ist immer konkurrenzfähig, da sie durchweg sehr gute Ergebnisse liefert und sehr robust ist angesichts des heterogenen Datenbestandes. Dadurch eignet sie sich gut für die Massenverarbeitung. Ein weiterer Vorteil für die Bayerische Staatsbibliothek ist, dass die Daten bereits vorliegen und alle zwölf Monate nach dem technischen State of the Art kostenfrei neu prozessiert und an die Bayerische Staatsbibliothek geliefert werden. Übrigens besteht für Nicht-Google-Books-Bibliotheken die Möglichkeit, OCR über die Google Cloud Vision API[29] zu beziehen.

Calamari liefert mit dem deep3-lsh4 Modell durchweg die besten Ergebnisse. Das Modell wurde auf sehr heterogenem Material trainiert, wodurch es sehr robust ist. Allerdings ist die Inbetriebnahme durch einen nicht mehr aktuellen Software-Stand kompliziert. Es lässt sich ohne Weiteres keine Aussage dazu treffen, wie lange eine komplette Neuverarbeitung dauern würde. Ein Einsatz von am MDZ vorhandenen GPU-Beschleunigerkarten ist mit dem aktuellen System nach aktuellem Stand (Juli 2023) jedenfalls nicht möglich.

Tesseract besitzt im Vergleich mit den übrigen OCR-D Engines das beste Verhältnis zwischen Ergebnissen, Performance und Einfachheit des Betriebes. Die frak2021 Modelle aus der UB Mannheim sind qualitativ nah an den Google OCR Ergebnissen, während Laufzeit und Skalierbarkeit besser sind als bei Calamari. Der Betrieb von Tesseract ist in verschiedenen Szenarien sehr einfach.

Einordnende Bemerkungen zur Untersuchung und ihren Ergebnissen

Der vorliegende Bericht leistet einen Beitrag zur vergleichenden Bewertung von Google OCR und Open-Source-OCR-Engines anhand von Daten, wie sie realistischerweise im Bereich der VD-(Massen-)Digitalisierung vorkommen. Insgesamt zeigt sich, dass Google OCR, Tesseract und Calamari sehr gute Ergebnisse angesichts des heterogenen und in Bezug auf Druck- und Scanqualität üblicherweise stark schwankenden Datenbestandes liefern. Diese Robustheit ist unabdingbar für einen Einsatz in der Massendigitalisierung, bei der eine feingranulare Auswahl von Modell und Bildvorverarbeitung pro Objekt nicht praktikabel ist.

Bei der Interpretation der Ergebnisse sollten folgende methodische Einschränkungen berücksichtigt werden: Trotz intensiven Reviews kann nicht von einer völlig fehlerfreien und einheitlichen Ground Truth ausgegangen werden. Ursache dafür ist der gewählte Ansatz, vorhandene OCR zu korrigieren und nur eine Person pro Zeile einzusetzen. Außerdem konnte Ground Truth für ausgewählte Griechischzeilen im Rahmen dieser Untersuchung nicht vollständig erarbeitet werden, insbesondere aufgrund der Schwierigkeit, Besonderheiten von Frühdrucken griechischer Texte methodisch sauber in Ground Truth zu überführen. Wünschenswert wäre die Erstellung einer umfangreicheren Ground-Truth-Menge unter stärkerer Berücksichtigung nicht lateinischer Schriften. Für letztere müssten ggf. erst geeignete und fachlich akzeptierte Transkriptionsregeln erstellt werden.

Die Festlegung auf die von Google vorgegebene Zeilen-Segmentierung ist problematisch, da häufig Artefakte von Nachbarzeilen im Zeilenbild vorhanden sind. Darunter leidet zumindest bei OCRopus offensichtlich auch die Erkennungsqualität. Schwierigkeiten ergaben sich auch bei der Vorverarbeitung der Zeilenbilder, da insbesondere OCRopus von Seitenbildern als Eingabe ausgeht. Zusätzlicher Aufwand wäre nötig, um die Vorverarbeitung an die Zeilenbilder anzupassen und mit verschiedenen Verfahren in diesem Bereich zu experimentieren.

Die vorliegende Untersuchung hat Aspekte der Layoutanalyse und deren Einfluss auf die OCR-Qualität bewusst ausgeklammert. Weiterführende Analysen sollten Verfahren aus diesem Bereich in den Fokus nehmen und die Qualität von Google OCR und quelloffenen OCR-Engines unter diesen Gesichtspunkten fortführend evaluieren.

Anmerkungen

1 OCR-D. Koordinierte Förderinitiative zur Weiterentwicklung von Verfahren der Optical Character Recognition (OCR) [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://ocr-d.de/de/about

2 Verzeichnis der im deutschen Sprachbereich erschienenen Drucke des 16. (VD16), des 17. (VD17) und des 18. (VD18) Jahrhunderts.

3 HAAF, Susanne, Frank WIEGAND und Alexander GEYKEN. Measuring the Correctness of Double-Keying: Error Classification and Quality Control in a Large Corpus of TEI-Annotated Historical Text. *Journal of the Text Encoding Initiative*. 2013, 4 [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://doi.org/10.4000/jtei.739

4 Den Kolleg*innen sei hier ausdrücklich für ihre Unterstützung gedankt.

5 Ground Truth Richtlinien [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://ocr-d.de/de/gt-guidelines/trans/index.html

6 Prodigy ist ein Annotierungstool, um Trainingsdaten für Machine Learning Modelle zu erstellen. Siehe: prodigy [Zugriff am 13. Oktober 2023]. Verfügbar unter: https://prodi.gy/

7 https://github.com/dbmdz/prodigy-recipes

8 Auf der OCR-Plattform Transkribus wird analog dazu ein Textual Tag »unclear« vorgeschlagen: »Use this tag when the text can not be transcribed since it is illegible.« Siehe: Textual Tags [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://help.transkribus.org/textual-tags

9 Tesseract OCR GitHub Repository [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://github.com/tesseract-ocr/tesseract; Tesseract User Manual [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://tesseract-ocr.github.io/tessdoc/

10 OCRopus GitHub Repository [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://github.com/ocropus-archive/DUP-ocropy

11 CIS OCR-D GitHub Repository [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://github.com/cisocrgroup/ocrd_cis

12 Kraken GitHub Repository [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://github.com/mittagessen/kraken; Die erste Version wurde 2015 released: https://github.com/mittagessen/kraken/tree/0.1.0; Kraken Dokumentation [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://kraken.re/main/index.html

13 WICK, Christoph, Christian REUL und Frank PUPPE. Calamari – A High-Performance Tensorflow-based Deep Learning Package for Optical Character Recognition. *Digital Humanities Quarterly*. 2020, 14(2). [Zugriff am: 13. Oktober 2023]. Verfügbar unter: http://www.digitalhumanities.org/dhq/vol/14/2/000451/000451.html; OCR GitHub Repository [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://github.com/Calamari-OCR/calamari; Calamari OCR Dokumentation [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://calamari-ocr.readthedocs.io/en/latest/

14 Version 0.1.0 erschien 2018 [Zugriff am: 26. Oktober 2023]. Verfügbar unter: https://github.com/Calamari-OCR/calamari/tree/v0.1.0

15 WICK, REUL und PUPPE, siehe Endnote 13.

16 Ground Truth Richtlinien: Level 2 [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://ocr-d.de/de/gt-guidelines/trans/level_2_2.html

17 Ground Truth Richtlinien: Level 1 [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://ocr-d.de/de/gt-guidelines/trans/level_1_4.html

18 LEVENSHTEIN, Vladimir I. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*. 1966, 10 (8), 707–710 [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://nymity.ch/sybilhunting/pdf/Levenshtein1966a.pdf

19 Für mehr Informationen zu diesen und alternativen Qualitätsmetriken siehe: NEUDECKER, Clemens, Karolina ZACZYNSKA, Konstantin BAIERER, Georg REHM, Mike GERBER und Julián MORENO SCHNEIDER. Methoden und Metriken zur Messung von OCR-Qualität für die Kuratierung von Daten und Metadaten. In: Michael FRANKE-MAIER und andere, Hrsg. *Qualität in der Inhaltserschließung*. 1. Aufl. Berlin, Boston: De Gruyter Saur, 2021, S. 137–166.

20 Dinglehopper GitHub Repository [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://github.com/qurator-spk/dinglehopper

21 WEIL, Stefan. Tesseract-Modell GT4HistOCR [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://ub-backup.bib.uni-mannheim.de/~stweil/tesstrain/GT4HistOCR/

22 WEIL, Stefan. Tesseract-Modell frak2021 [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://ub-backup.bib.uni-mannheim.de/~stweil/tesstrain/frak2021/

23 Calamari Models [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://github.com/Calamari-OCR/calamari_models; Calamari Models Experimental [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://github.com/Calamari-OCR/calamari_models_experimental

24 REUL, Christian, Christoph WICK, Maximilian NOETH, Andreas BUETTNER, Maximilian WEHNER und Uwe SPRINGMANN. Mixed Model OCR Training on Historical Latin Script for Out-of-the-Box Recognition and Finetuning. In: *Proceedings of the 6th International Workshop on Historical Document Imaging and Processing (HIP ,21)*. New York, NY, USA: Association for Computing Machinery, 2021, S. 7–12. [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://doi.org/10.1145/3476887.3476910

25 WEIL, Stefan. Fraktur model trained from enhanced Austrian Newspapers dataset [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://doi.org/10.5281/zenodo.6891851

26 KIESSLING, Benjamin. Preliminary Fraktur Model [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://doi.org/10.5281/zenodo.7089017

27 REUL, Christian, Christoph WICK, Uwe SPRINGMANN und Frank PUPPE. Transfer Learning for OCRopus Model Training on Early Printed Books. *Zeitschrift für Bibliothekskultur,* 2017, 5(1), 32–45 [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://doi.org/10.5281/zenodo.4705364

28 Fraktur Model for OCRopus [Zugriff am: 13. Oktober 2023]. Verfügbar unter: https://github.com/jze/ocropus-model_fraktur

29 https://cloud.google.com/vision [Zugriff am: 10. November 2023].

Verfasser*innen

Johannes Baiter, Softwareentwickler,
Münchener Digitalisierungszentrum (MDZ),
Bayerische Staatsbibliothek,
Ludwigstraße 16, 80539 München,
Telefon +49 89 28638-2970,
johannes.baiter@bsb-muenchen.de,
https://orcid.org/0000-0002-1807-9728
Foto: privat

Marcus Bitzl, Teamleiter Workflow und Suche, Münchener Digitalisierungszentrum (MDZ), Bayerische Staatsbibliothek,
Ludwigstraße 16, 80539 München,
Telefon +49 89 28638-2998,
marcus.bitzl@bsb-muenchen.de,
https://orcid.org/0000-0001-7414-3643
Foto: privat

Sebastian Mangold, Metadaten-Spezialist,
Münchener Digitalisierungszentrum (MDZ),
Bayerische Staatsbibliothek,
Ludwigstraße 16, 80539 München,
Telefon +49 89 28638-2752,
sebastian.mangold@bsb-muenchen.de,
https://orcid.org/0009-0000-1624-9888
Foto: privat

Katharina Schmid, Softwareentwicklerin,
Münchener Digitalisierungszentrum (MDZ),
Bayerische Staatsbibliothek,
Ludwigstraße 16, 80539 München,
Telefon +49 89 28638-2381,
katharina.schmid@bsb-muenchen.de,
https://orcid.org/0000-0001-6057-6640
Foto: FotoPhositiv