Textdigitalisierung

Daniel Stökl Ben Ezra

https://orcid.org/0000-0001-5668-493X

Abstract Textdigitalisierung beschreibt die Konvertierung digitaler Bilddaten beschrifteter Objekte in maschinenlesbare Texte.

Keywords HTR, OCR, Layoutanalyse, Handschriften, Maschinelles Lernen, Neuronale Netzwerke

1. Einleitung

Als Textdigitalisierung möchten wir hier die Konvertierung digitaler Bilder beschrifteter Objekte jedweder Art (Handschriften, Inschriften, Keilschrifttafeln, Drucke, etc.) in maschinenlesbare Texte bezeichnen, wobei „Text" weit über Buchstabensequenz hinausgeht. Datenbanken mit Volltextsuchoptionen in retrodigitalisierten Drucken wie JSTOR (seit 1994) oder Google-Books (seit 2004) haben die Arbeitsweise in allen wissenschaftlichen Fächern grundlegend verändert. Die letzten zehn Jahre haben durch den bahnbrechenden Fortschritt in der automatischen Dokumentenanalyse v. a. im maschinellen Lernen *(machine learning)* die Möglichkeiten revolutioniert, auch für Forschende nicht nur schwierige Drucke der wichtigsten Kulturschriften, sondern sogar historische Handschriften automatisch zu analysieren.

Transkription ist nicht die einzige Analyseebene. Gerard Genette (1982) hat die Bedeutung des Layouts und der Nicht-Haupttextabschnitte für das (Vor)verständnis von Texten eindrücklich herausgearbeitet. Layout enthält kritische Informationen, wie z. B. die Unterscheidung zwischen Titel und Haupttext, Haupttext und Anmerkungen, Sprechern bei Dramen, Versen bei Dichtung oder Textverbindungen bei Übersetzungen oder Kommentaren. Auch Wahl von und Wechsel in Schriftart, -schnitt, -breite, -lage (normal, kursiv, schräg), -stärke, -farbe, Drucktypen, Alphabeten sind essentielle Informationsträger, die die Analysetiefe eines analysierten Textes über die einfache Buchstabensequenz maßgeblich vertiefen können (Beinert 2021). Dazu kommen Möglichkeiten, über maschinelle Paläographie, Layoutuntersuchung und Kodikologie all diese feinen Unterschiede zur Netzwerkanalyse, Datierung und Verortung der Einzelobjekte auszuwerten. Über dieses *big data* gewinnen die traditionellen Hilfswissenschaften eine völlig neue Stellenbedeutung.

Daniel Stökl Ben Ezra: Textdigitalisierung, in: Christopher A. Nunn und Frederike van Oorschot (Hrsg.): Kompendium Computational Theology 1. Forschungspraktiken in den Digital Humanities, Heidelberg: heiBOOKS, 2024, S. 107–125. DOI: https://doi.org/10.11588/heibooks.1459.c21906

Der hier vorliegende Aufsatz wird nach einigen kurzen Einleitungsfragen zur Bild- und Textkodierung, Neuronalen Netzwerken sowie zu existierenden Programmen zunächst auf die Layoutanalyse und die Leseordnung, dann auf Computerpaläographie und Texterkennung eingehen. Die Diskussion verläuft in permanenter Interaktion mit der in der Informatik „Bildverarbeitung" genannten Disziplin (Maier et al. 2021).

2. Kodierung von Bildern

Computer können nur die Werte 0 und 1 unterscheiden. Sowohl Texte als auch Bilder müssen also zunächst als Sequenzen von Nullen und Einsen dargestellt werden. Gewöhnlich wird das Bild mit einem Raster in eine Tabelle/Matrix mit Zeilen und Spalten übertragen, in der jede Zelle einen Wert für einen Bildpunkt (Pixel) enthält. Hochauflösende Bilder haben mehr Pixel für die gleiche Objektoberfläche als Bilder mit niedriger Auflösung. Das einfachste Bildformat sind schwarz-weiß Bilder, die für jedes Pixel nur 0 oder 1, also beispielsweise Vordergrund (Tinte) oder Hintergrund (Papier), kennen. Das führt, vor allem bei niedriger Auflösung oder starker Vergrößerung, zu den bekannten Treppenstufen (Abb. 1).

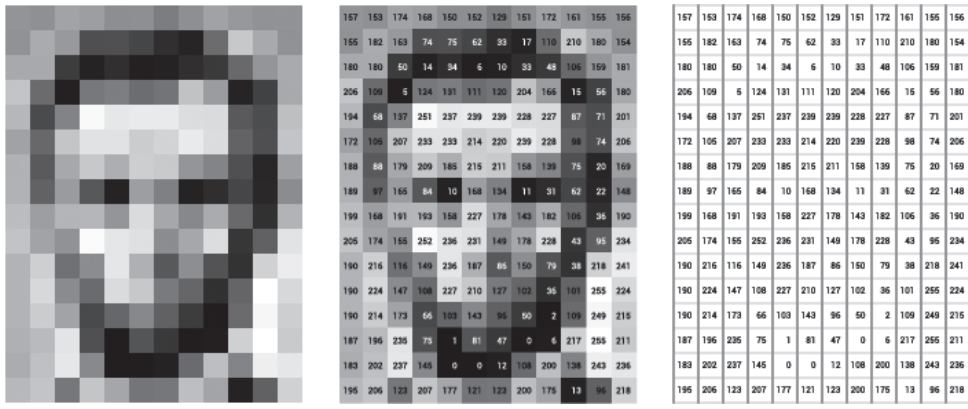

Abb. 1 Abraham Lincoln als Graustufenbild mit den Werten auf einer Skala von 0 – 255.

Wer feiner nuancieren möchte, verwendet Graustufenbilder, die Zwischenstufen ermöglichen. 0 steht weiterhin für Schwarz, doch statt 1 wird nun 255 für Weiß verwendet und alle Zahlen dazwischen bezeichnen Grauwerte, je nachdem ob sie näher an Weiß oder an Schwarz sind. Dies sind sogenannte 8-Bit Bilder, also eine Kombination von acht Speicherzellen (Bits) für jedes Pixel ($256 = 2^8$). Wer noch feiner abstufen

möchte, kann auch umfangreichere Graustufenskalierungen verwenden und dann mit Werten von 0 bis 65555 (16-Bit – 2^{16}) oder mehr arbeiten. Wiederum wird der tiefste Wert für Schwarz und der höchste Wert für Weiß verwendet, und die dazwischenliegenden Werte stellen die möglichen Graustufen dar.

Für Farbbilder wird eine Kombination von drei übereinanderliegenden Graustufenbildern mit einer 8-Bit Skala von 0–255 verwendet, deren Farbkomposition (z. B. rot-grün-blau = RGB) für unsere Augen die Mischfarben erzeugt, z. B. violett, braun, orange oder rosa. Statt RGB können auch andere Farbkanäle verwendet werden, z. B. Cyan, Magenta und Gelb, wie in Farbdruckern. Und wie bei Graustufenbildern kann man statt einer 8-Bit Skala auch eine wesentlich feinere 16-Bit Skala wählen.

Je größer Höhe und Breite, Auflösung oder Farbenskala, desto mehr Speicher braucht ein Bild. Um Platz zu sparen oder größere Geschwindigkeit bei der Verarbeitung zu gewinnen, werden Bilddaten oft komprimiert. Jeder Kamera- oder Scannerhersteller hat sein firmeneigenes Format *(Raw).* Bei Export wird dies gewöhnlich in TIFF-, PNG- oder JPEG-Bilddateien konvertiert. Je nach Kompression benötigen TIFF und PNG wesentlich mehr Speicherraum als JPEG Bilddateien (Extension jpg), die mit einem Kompressionsalgorithmus erzeugt werden, der mehr Informationsverlust in Kauf nimmt, um weniger Speicherraum einzunehmen. Dies erzeugt sogenannte Artefakte, die man bei starker Vergrößerung leicht daran erkennen kann, dass das Bild wie Badezimmerkacheln aussieht. Bei guter Bildauflösung (mindestens 30 Pixel für ein ‚a' oder ‚א', bei komplexeren Schriften wie chinesisch wesentlich mehr) reichen jpgs für Layoutanalyse und Texterkennung heutzutage oft aus. Automatische Paläographie oder Schreiberzuordnung erreicht bei TIFF- und PNG-Dateien bessere Ergebnisse.

3. Kodierung von Texten

Auch Text ist im Computer als Sequenzen von 0 und 1 abgespeichert. Früher, als Speicher noch wesentlich teurer war als heute, wurden für alle Varianten zusammen für jedes Zeichen nur 8 Bit reserviert, sodass Computer nur 256 unterschiedliche Werte (Kodepunkte) kannten, die je nach sprachlicher oder geographischer Arbeitsumgebung immer nur eine Auswahl von Lateinisch, Griechisch, Kyrillisch oder Hebräisch darstellen konnten. Dies war die sogenannte Extended ASCII-Tabelle. Ein zusätzlicher Code in der Textdatei gab dabei an, welche Alphabete durch die Rohzahlen in der Datei gemeint waren. So konnte man sowohl in Deutschland wie auch in Bulgarien, Israel oder Saudi-Arabien mit den lokalen Alphabeten arbeiten, aber nie mit allen Schriften gleichzeitig. Komplexere Schriften, bei denen die Gesamtzahl der Zeichen größer als 256 war, wie z. B. Chinesisch, waren unmöglich. Wenn der Schlüsselcode unbekannt war, half nur ein Ausprobieren der Kodierungen, bis der Text lesbar dargestellt war. Diese Begrenzung auf 256 unterschiedliche Zeichen war für philologische Arbeiten in

der Theologie, wo neben Griechisch und Hebräisch oft auch Syrisch, Arabisch, Armenisch, Georgisch oder Koptisch, manchmal auch Akkadisch, Ägyptisch, Äthiopisch oder Sanskrit verwendet wurden, eine große Herausforderung.

Die Einführung des Unicode-Standards seit 1991 hat einen großen Schritt zur Lösung dieses Problems beigetragen. Ähnlich wie bei den Graustufenbildern mit feinerer Nuancierung wurde zunächst einfach der für jedes Zeichen reservierte Speicher von 8 Bit auf 16 Bit verdoppelt, was 2^{16} = 65,536 unterschiedliche Kodepunkte in der Tabelle ermöglicht. 2022 konnten mit dieser auch als UTF-8 bezeichneten Kodierung insgesamt 161 Schriften in einer einzigen Tabelle definiert werden. Auch wenn dies ein großer Schritt vorwärts war, sind damit noch nicht alle Schwierigkeiten für die digitale Modellierung historischer Schriften gelöst: Beispielsweise gibt es noch keinen Unicode für die babylonische Vokalisierung hebräischer Texte und Ägyptisch oder Akkadisch sind bislang nur zum Teil standardisiert.

Eine Auseinandersetzung mit den Feinheiten von Unicode sei allen Geisteswissenschaftler*innen angeraten, denn die zu lösenden Grundfragen von Schriftkodierung sind trickreich. Auf weitere Fragen wie Leseordnung bei bidirektionalen Texten, die Hebräisch und Lateinisch mischen, oder Zeichenkombinationskodierung kommen wir unten bei der Texterkennung zurück.

4. Neuronale Netzwerke

Der rasante Fortschritt in der automatischen Dokumentenanalyse der letzten Jahre hat fünf voneinander abhängige Hauptursachen: Hardware (Speicher, Geschwindigkeit), Software (Neuronale Netzwerke), Trainingsdatenmengen, Massendigitalisierung und Open-Source-Politik. Prozessoren sind viel schneller geworden und können noch dazu durch gewachsene Arbeitsspeicher wesentlich größere Datenmengen gleichzeitig verarbeiten. Auch Festplattenspeicher und Internetdatentransmission (Glasfaser) sind preislich billiger und qualitätsmäßig um vieles besser als noch vor zehn Jahren. Massendigitalisierungsprojekte von kulturell bedeutenden Handschriftensammlungen, Archiven und Bibliotheken haben zu einer Flut von Bilddaten geführt. Das Interesse verschiedener Großkonzerne an der Verarbeitung großer schriftlicher und mündlicher Textmengen (Google-Books, Youtube, Netflix, Zoom), haben nicht nur bekannte Algorithmen verbessert sondern auch derer neue entwickelt. Ein Teil dieser Algorithmen ist in open-source Paketen der wichtigsten Programmiersprachen öffentlich frei zugänglich (z. B. *pytorch* von Facebook, *TensorFlow* von Google). Dazu kommen schließlich verschiedene Forschungsprojekte, die ihre Trainingsdaten mit offenen Lizenzen veröffentlicht haben, und somit anderen Akteuren ermöglichen, mit ihnen neue Algorithmen zu entwickeln oder zu optimieren.

Für fast alle Etappen automatischer Dokumentenanalyse werden unterschiedliche Formen sogenannter künstlicher neuronaler Netzwerke verwendet. Das Grund-

prinzip war seit Rosenblatt 1958 bekannt, doch erst die oben erwähnte Konstellation vom simultanen Fortschritt bei Hardware, Software, Daten und open-source hat zu ihrem schwindelerregenden Siegeszug geführt, angefangen mit Jürgen Schmidhubers und Yann LeCuns Arbeiten in den frühen 90ern). Das Grundprinzip ist eine sehr komplexe Formel mit Tausenden, Millionen oder sogar Milliarden von Parametern, die vom Computer in einem „Training" genannten Lernprozess selbst optimiert werden. Das Endergebnis einer trainierten Netzwerk-Architektur wird als „Modell" bezeichnet, da es die Fragestellung mathematisch modelliert (Abb. 2).

Künstliche neuronale Netzwerke emulieren in gewisser Weise die Arbeitsweise von Gehirnen. Die drei relevantesten Netzwerktypen sind zur Zeit *Convolutional Neural Networks* (CNN), *Recurrent Neural Networks* (RNN) und *Transformers* (die deutschen Termini sind ungebräuchlich). Das wichtigste gemeinsame Prinzip ist die Einbeziehung des Kontextes für jeden Datenpunkt. CNNs sind besonders für Bilder interessant, weil hier ein Datenpunkt (d. h. ein Pixel) im Kontext eines Rechtecks betrachtet wird. So kann der Computer z. B. abstrakte Konzepte wie Kurven unterschiedlicher Krümmung und Linien in unterschiedlichen Winkeln und Ausrichtungen und Kombinationen lernen. RNNs sind für Sequenzen wie Audioaufnahmen oder Texte besonders interessant, da sie darin flexibel sind zu lernen, wieviel Kontext für ein gewisses Phänomen einzubeziehen ist. Bei längeren Sequenzen verwendet man dafür gerne sogenannte LSTMs, *Long-Short Term Memory Neural Networks.* Im Deeplearning werden mehrere Ebenen von CNNs und RNNs miteinander verbunden, so dass äußerst komplexe Architekturen entstehen, die große Datenmengen und oft viel Zeit benötigen, um trainiert zu werden, aber auch hervorragende Ergebnisse liefern, sowohl für Layoutanalyse als auch für Transkription und viele andere Aufgaben wie zum Beispiel Klassifikation.

Transformer sind vor allem über die jüngsten Großen Sprachmodelle (*Large Language Models,* LLMs) wie BERT *(Bidirectional Encoder Representations from Transformers)* und GPT *(Generative Pre-Trained Transformer)* bekannt geworden. Bei richtiger Konfiguration und optimalem Lernprozess können sie lernen, Layoutanalyse gleichzeitig mit der Transkription auszuführen, Eigennamen zu erkennen *(Named Entity Recognition),* die entstandenen Texte zu übersetzen oder zusammenzufassen oder sogar Fragen zu beantworten.

Beim Trainieren unterscheidet man zwischen überwachten *(supervised),* selbstüberwachten *(self-supervised)* und nicht-überwachten *(unsupervised)* Trainingsprozessen. Beim überwachten Trainingsprozess zeigt man dem Computer viele *Fragen* mit den dazugehörigen *Antworten* und der Computer versucht, schrittweise die Parameter der Formel zu optimieren. Frage-Antwort Paare richten sich nach dem Lernziel und können je nach Aufgabe grundsätzlich ganz unterschiedlich sein: (1) das Bild einer Textzeile und die dazugehörige Transkription. (2) das Bild einer Handschriftenseite und die dazugehörigen Polygone des Layouts. (3) das Bild einer Buchseite und die dazugehörige Drucktype, etc. Sowohl Frage als auch Antwort werden natürlich als eine Zahl (Skalar), Zahlensequenz (Vektor) o. ä. dargestellt, denn etwas Anderes

kennen Computer nicht. Am Anfang sind alle Parameter zufällig initialisiert. Nach jedem Lernvorgang (Fragestellung – Antwortberechnung) wird die Distanz zur richtigen Antwort gemessen und die Parameter werden so angepasst, dass die berechnete Antwort der richtigen näherkommt. Wenn dies möglich ist (und es ist nicht immer so), spricht man davon, dass das Modell konvergiert. In regelmäßigen Abständen wird die Computerformel einem Test ausgesetzt, in dem ihr Fragen-Antwort Paare vorgelegt werden, ohne dass der Computer aus ihnen lernt, also ohne Formelveränderung, sondern nur um die augenblickliche Präzision des gegenwärtigen Modells zu ermitteln. Die Kunst ist es, die Netzwerkarchitektur so zu formulieren, dass sie mit möglichst wenig Trainingsmaterial und in der schnellstmöglichen Rechenzeit die besten Modelle erreichen kann. Das Training wird zumeist dann abgebrochen, wenn der Benutzer anhand der Testergebnisse feststellt, dass der Computer sich nicht weiter verbessert. Abschließend wird jenes Modell gespeichert, welches im Vergleich die besten Ergebnisse lieferte.

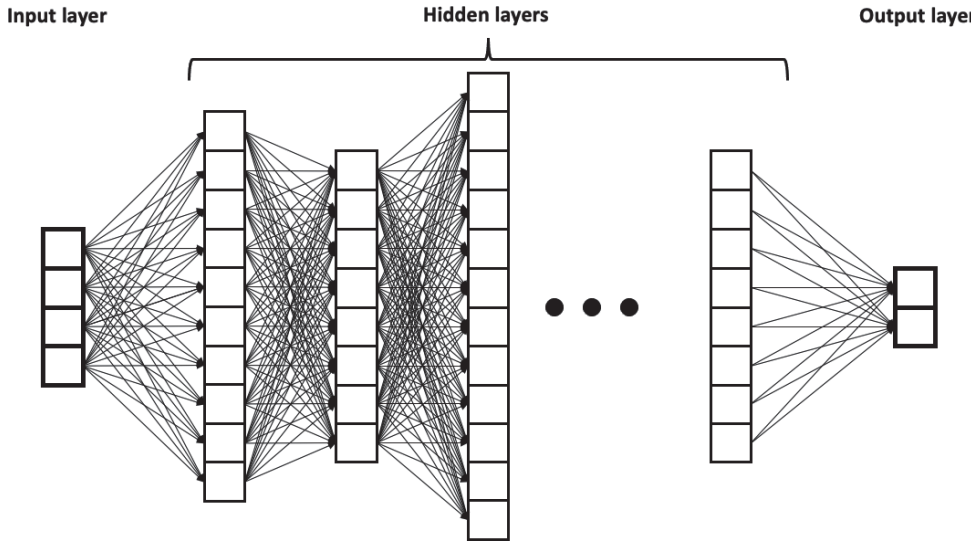

Abb. 2 Beispielschema eines neuronales Netzwerks mit einer eindimensionalen Eingabe-Schicht (mit vier Variablen), die über ein komplexes Netz zu einer Ausgabe-Schicht mit zwei Variablen führt. Jede der Linien ist eine mathematische Operation, deren Parameter optimiert werden müssen.

Auch der Trainingsprozess ist nicht uniform. Vielleicht hilft die folgende bildliche Vorstellung: Ein Raumschiff soll so weit wie möglich durch ein komplexes Höhlenlabyrinth fliegen, darf aber immer nur Wendungen in einem gewissen Winkel vornehmen und muss dann eine gewisse Distanz in diese Richtung vorwärtsfliegen. Ist die gewählte Distanz zu kurz, krebst das Raumschiff in der großen Eingangshöhle

herum, ohne einen der Eingänge zu finden. Ist die Distanz zu groß gewählt, kann das Raumschiff nicht durch enge Kurven hindurchkommen. Winkel und Distanz gehören hier zu den *Hyperparametern* des Trainingsvorgangs, vielleicht am ehesten mit der *Lernrate* (orig. „Learning rate“) zu vergleichen.

Beim *selbst-überwachten* Trainingsprozess berechnet der Computer selbst die Antworten automatisch aus den Fragen. Zum Beispiel zeigt man dem Computer eine große Anzahl Bilder von Handschriftenzeilen, von denen ein Teil geschwärzt ist, und von ihm wird verlangt, ein Ersatzbild vorzuschlagen, wie der geschwärzte Zeilenabschnitt ausgesehen haben könnte. Mit Hilfe einer Analyse des Unterschiedes zwischen dem Computervorschlag und dem Originalbild wird das Modell optimiert. Hier lernt der Computer allmählich selbst das Prinzip, welche Pixelcluster gewöhnlich zwischen welchen anderen Pixelclustern stehen. Selbst-überwachtes Training wird manchmal als Vortraining vor überwachtem Training verwendet, weil es erlaubt Grundprinzipien vorher zu lernen und die Menge an Trainingsdaten mit händisch erstellten Antworten zu verringern.

Eine Gefahr ist die Überanpassung (orig. „Overfitting“). Das kann man vergleichen mit einem Schüler, der die Antworten zu den Übungsfragen auswendig lernt, ohne das zugrundeliegende Prinzip zu begreifen. Er wird die Übungsfragen nahezu perfekt beantworten können, aber nicht dazu befähigt, ungesehene Fragen zu lösen. Dies kann z. B. geschehen, wenn das Trainingskorpus nicht der Netzwerkarchitektur angepasst ist (zu wenig Trainingsmaterial für eine zu komplexe Fragestellung) oder wenn die Lernrate, mit der der Computer die Parameter nach jedem Lernvorgang anzupassen versucht, zu hoch oder zu niedrig gesetzt ist. Natürlich haben wir hier nur einige wenige Hyperparameter beschreiben können.

5. Existierende Programme

Bis vor wenigen Jahren arbeiteten die meisten an OCR interessierten Firmen und Gelehrten mit kommerziellen Programmen wie *ABBYY Finereader,* das sehr erfolgreich mit modernen englischen oder deutschen Texten umgehen konnte, aber an handschriftlichem Material oder selteneren Druckschriften wie den syrischen Alphabeten, für die der Markt zu klein erschien, völlig scheiterte. Googles weit verbreitetes open-source Programm *Tesseract* wird nur für OCR aber nicht für HTR empfohlen.[1] Heutzutage (Juli 2023) gibt es mehrere Programme, die mit großem Erfolg zur Massendigitalisierung auch handschriftlichen Materials eingesetzt werden.

Seit 2016 ermöglicht *Transkribus* über eine komplexe JAVA-App oder eine simplifizierte Web-App die automatische Layoutanalyse und Transkription von Schrift-

1 FAQ: https://tesseract-ocr.github.io/tessdoc/FAQ.html#can-i-use-tesseract-for-handwriting-recognition, zuletzt aufgerufen am 15.06.2024.

objekten, die manuelle Korrektur von Layoutanalyse und Erkennung sowie das Trainieren von eigenen Transkriptionsmodellen anhand der eingegebenen Daten mit ausgezeichneten Resultaten (Kahle et al. 2017, Mühlberger et al. 2019). Das ursprünglich in mehreren europäischen Forschungsprojekten entwickelte Programm wurde 2019 in Form einer europäischen Genossenschaft kommerzialisiert.[2] Zurzeit zahlen Benutzer*innen pro Seite für automatische Layoutanalyse und/oder automatische Texterkennung. Die Plattform und trainierte Modelle sind dementsprechend closed-source. Andere kommerzielle Programme sind z. B. *Ocelus* und *Calfa* (Vidal Gorène 2021a).

Auf der open-source Seite war das von Thomas Breuel (2008) entwickelte *OCRopus/ocropy* ein entscheidender Schritt nach vorne. Obgleich es, wie der Name schon sagt, nur für OCR entwickelt worden war, setzte unser Pariser Team es mit Hilfe von Marcus Liwicki seit 2015 auch für Handschriftenerkennung ein. Allerdings brauchten interessierte Personen Programmierkenntnisse. Es gab nur eine sehr rudimentäre Möglichkeit, Transkriptionen einzugeben, nur für gerade Zeilen und nur eine sehr einfache Segmentierung. Dafür wird seit 2018 *eScriptorium* um Benjamin Kiesslings *Kraken* entwickelt (Kiessling et al. 2019, Stokes et al. 2021). Es ist das bislang einzige open-source Programm für Handschriftenanalyse mit ergonomischer Nutzungsoberfläche für Layout- und Transkriptionskorrektur sowie Text-Alignierung. Es kann direkt auf Linux und Mac-OS und per WSL *(Windows Subsystem for Linux)* auch auf Windows Rechnern installiert werden. Um in einer Forschungsgruppe zusammen an den gleichen Dokumenten arbeiten zu können braucht man einen Server. Wer Layout- oder Transkriptions-Modelle trainieren möchte, braucht eine GPU mit ausreichend Arbeitsspeicher.

6. Layout-Analyse

Computergestützte Layout-Analyse hat zwei Ziele. Zum einen ist sie – bislang – ein notwendiger Schritt *vor* der Texterkennung. Zum anderen enthält das Layout essentielle Informationen für die Hierarchie, Leseordnung (s. u.), unterschiedliche Texttypen, Unterscheidung von Bild und Text, etc. So kann die Layoutanalyse auch nach der Texterkennung noch einmal eine wichtige Rolle für das Textverständnis spielen.

Früher benutzte man für die Layoutanalyse vielfach morphologische Operationen, um unterschiedliche Textblöcke und Zeilen zu erkennen. Heute wird dies quasi überall von neuronalen Netzwerkarchitekturen übernommen, die sowohl die Segmentierung von Blöcken und ihre Einteilung in Typen (Spalte, Kopfzeile, Marginalie, Illustration, Tabelle, Apparat etc.) als auch die Erkennung von Zeilen und ihre

2 S. https://readcoop.eu/a-short-history-of-transkribus-with-gunter-muhlberger, zuletzt aufgerufen am 15.06.2024.

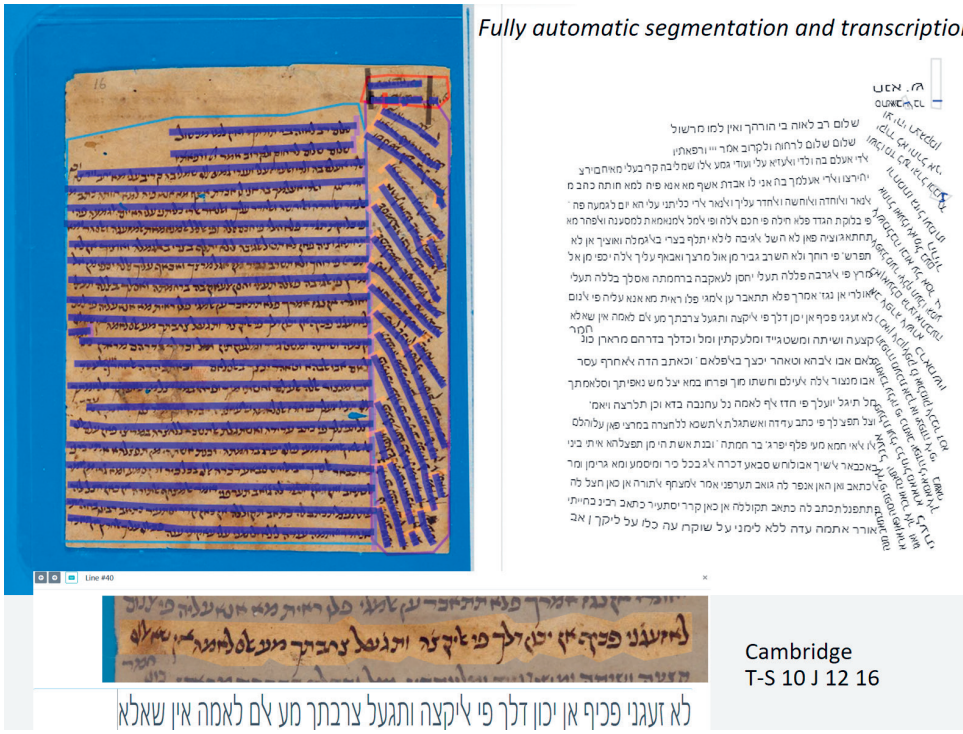

Abb. 3 Vollautomatische Layoutsegmentierung und Transkription des Genizahfragmentes Cambridge, T.-S. 10 J 12 16 mit *eScriptorium*, erstellt im HTR4PGP Projekt

Einteilungen in Typen (Haupttextzeile, interlineare Zeile) und die Schreibrichtung (horizontal, vertikal, auf dem Kopf) bewerkstelligen (Abb. 3). Dabei legt eine sogenannte Segmentierungsontologie fest, welche Block- und Zeilentypen für welche Phänomene zur Anwendung kommen können. Zonen müssen übrigens nicht unbedingt Textblöcke sein. Benutzer können die Bildsegmentierung auch zur Ortung von Bibliotheksstempeln, Münzen, Illustrationen o. ä. benutzen.

Für die Blocksegmentierung gibt es zurzeit zwei unterschiedliche Ansätze. Der eine Ansatz benutzt Prinzipien für die Objekterkennung wie Ampeln oder Schilder bei selbstfahrenden Autos (Clérice 2022). Dieser Ansatz funktioniert sehr gut bei Textobjekten mit ausschließlich rechteckigen Blöcken, die genau ohne Rotation digitalisiert worden sind. Bei komplexerem Layout, z. B. L-förmigen Blöcken, oder bei kleinen Rotationen gibt es jedoch schnell Probleme.

Der andere Ansatz ist ein Pixelklassifikator (Kiessling 2020). Alle Bildpixel werden keiner, einer oder mehreren gewünschten Regionstypen zugewiesen. Anschließend wird die Pixelwolke für jeden Regionstyp ermittelt und daraus ein oder mehrere Polygone rekonstruiert. Dieser Ansatz kommt besser mit komplexem Layout oder rotierten Digitalisaten zurecht, hat aber den Nachteil, Pixelgrupppen des gleichen

Typs, die einander sehr nah stehen, dem gleichen Polygon zuzuweisen. Daher neigt er z. B. dazu, zwei eng nebeneinanderstehende Haupttextspalten als eine einzige Zone einzuordnen.

Neben oder nach der Blocksegmentierung erfolgt die Zeilensegmentierung (Grüning et al. 2019). Auch hier gibt es zwei Ansätze. Entweder wird ein neuronales Netzwerk trainiert, erst die Grundlinie und Schreibrichtung jeder Zeile zu entdecken und dann ein Polygon zu berechnen, das diese Grundlinie so umgibt, dass möglichst alle Tintenspuren der Zeichen dieser Zeile inklusive eventueller Punkte und Striche über oder unter ihnen eingeschlossen werden. Oder das neuronale Netzwerk wird trainiert, direkt das Zeilenpolygon zu erkennen und dann die Schreibrichtung abzuleiten.

In *kraken*/*eScriptorium* kann man sehr komplexe Segmentierungsmodelle sogar mit 20 unterschiedlichen Regions- und Zeilentypen trainieren, solange die Trainingsdaten homogen und zahlreich genug sind (Stökl Ben Ezra 2022b). Das Trainieren einfacher spezifischer Segmentierungsmodelle sind auch mit einigen wenigen Trainingsseiten möglich.

7. Leseordnung

Unterschiedliche Textsorten divergieren hinsichtlich ihrer Leseordnung: Die Artikelabschnitte einer oder mehrerer Zeitungsseiten, eine kritische Edition, eine zweisprachige Ausgabe, eine Tabelle, eine Handschrift mit einem Grundtext oder eine Handschrift mit einem Grundtext und einem oder mehreren Kommentaren, ein Brief oder Postkarten haben oft sehr unterschiedliche Weisen, in welcher Reihenfolge der oder die Texte gelesen werden sollen. Poetische Texte werden oft in Stichen geschrieben, die aussehen, als wären es zwei Spalten eines Prosatextes. Leseordnungsanalyse hängt eng mit der Layoutanalyse zusammen. In *kraken* kann sie zur Zeit in einer noch in Entwicklung befindlichen Version trainiert werden, deren Einbindung in *eScriptorium* noch einige Monate in Anspruch nehmen wird, um die notwendige ergonomische Nutzerumgebung zu konzipieren und zu realisieren (zur Methode, s. Quirós & Vidal 2022).

8. Computer-Paläographie

Ein Gebiet, in dem neuronale Netzwerke Menschen sehr nahekommen oder sie sogar übertreffen, ist die Klassifizierung von Objekten. Computer-Paläographie kann zu unterschiedlichsten Zwecken eingesetzt werden. Objekte können nach verwendeten Drucktypen oder Schriften, Schulen oder Schreibern eingeordnet oder auch zeitlich

datiert oder geographisch verortet werden (Seuret et al. 2021, Droby et al. 2022, Popovic et al. 2021). Banken benutzen diese Technik schon seit langem zur Verifikation von Unterschriften. Man kann so auch analysieren, ob eine Handschrift von einer oder mehreren Personen geschrieben wurde. Besonders das Friedberg Genizah Project hat schon sehr früh derartige Techniken benutzt, wo Nutzende, die nach ähnlichen Fragmenten zu einem Fragment X suchten, die eventuell aus dem gleichen Kodex stammen, vom System aus den ca. 300 000 Fragmenten andere Kandidaten vorgeschlagen bekam, was tatsächlich zu einer Vielzahl von „Joins" geführt hat (Wolf et al. 2010).

9. Texterkennung

Früher war es für Texterkennung notwendig, zunächst Zeichen für Zeichen zu isolieren. Moderne neuronale Netzwerke basieren hingegen auf ganzen Textzeilen und erzielen damit bessere Ergebnisse, da der Kontext jedes Datenpunktes viel besser einbezogen werden kann. Gerade bei Handschriften, kann man Buchstaben oft nicht isolieren und selbst die richtige Entzifferung ganzer Worte hängt vom Kontext der vorhergehenden und nachfolgenden Worte und Zeichen ab.

Es ist beim gegenwärtigen Stand der Forschung nicht möglich genau anzugeben, wie viele Zeilen man benötigt, um ein Modell zu trainieren, oder wie oft ein Zeichen im Trainingsdatensatz vorkommen muss, damit der Computer es lernen kann. Dies hängt von Faktoren wie Komplexität des Netzwerkes, Zahl der unterschiedlichen zu erlernenden Schriftzeichen, Lesbarkeit und Einheitlichkeit der Schrift, Kontrast, Einheitlichkeit der Orthographie und des Vokabulars, Bildqualität u. v. m. ab. Beim Erstellen des Trainingskorpus ist vor allem bei kollektiver Annotationsarbeit oder der Affiliation von unterschiedlichen Projekten auf strikte Homogenität der Daten zu achten. Das gleiche Phänomen soll auch immer gleich transkribiert werden. Wenn man dem Computer beibringen möchte, Abkürzungen aufzulösen oder orthographische Varianten normalisiert zu transkribieren, braucht man dementsprechend mehr Trainingsmaterial. Im Prinzip kann man den Computer auch trainieren, zwischen unterschiedlich gesetzten Schriftzeichen zu unterscheiden, also z. B. zwischen kursiv, fett und normal oder zwischen unterschiedlichen Schrifttypen.

Vielfach lohnt sich eine Analyse des Endzwecks. Wenn ähnlich aussehende Schriftzeichen unterschiedlich kodiert werden, bedeutet dies für den Computer, dass sie für ihn genauso differieren, wie für uns visuell völlig unterschiedliche Zeichen. Müssen wirklich alle unterschiedlichen Arten von Anführungszeichen (‚„"«»"") exakt so differenziert werden, oder dürfen einige oder alle vereinheitlicht werden? Je nach Projekt kann die eine oder die andere Richtung vorteilhafter sein.

Hier lohnt es sich, noch einmal auf Unicode zurückzukommen. Manche Buchstaben sind in der Unicodetabelle mehrfach deklariert, z. B. gibt es *s* und *ſ*. Diese und

andere feine Unterschiede nutzend, kann man damit eine allographische oder hyperdiplomatische Transkription trainieren, die Allographen differenziert. Dies weist auch auf eine Grundfrage hin, die sich jedes Editionsprojekt aber auch das Unicodekonsortium stellen muss. Was ist ein gleiches Zeichen, wo sind Unterschiede zwingend und wo muss man Nutzer*innen Möglichkeiten lassen, selbst zu unterscheiden?

Ein Kreis kann u. v. A. ein lateinisches O oder o, ein griechisches O oder ο, ein hebräisches ס oder eine arabische ه sein. Was ein Kreis in einem historischen Dokument ist, entscheidet der Kontext. Manchmal unterscheidet die Unicode-Tabelle zwischen unterschiedlichen Alphabeten, aber nicht immer. Für Lateinisch, Englisch und Deutsch werden die gleichen Kodierungen für O benutzt. Ö ist entweder ein O, kombiniert mit einem Trema (¨) oder ein einziges Zeichen Ö. Visuell lässt sich dies nicht unterscheiden. Aber wenn die eine Hälfte der Trainingsdaten mit der Kombination von O + ¨ und die andere Hälfte mit Ö erstellt worden sind, wird das Netzwerk verwirrt, denn für den Computer sind diese jeweils völlig andere Einheiten. Bei griechischen Akzenten oder hebräischen Diakritika und Vokalen kommt noch die Reihenfolge dazu, denn Sin, Sin-Punkt, Dagesh und ein Kamatz können in 24 unterschiedlichen Permutationen notiert werden. Daher gibt es die Möglichkeit einer Kodepunktnormalisierung, die entweder alle Zeichen soweit möglich aufspaltet und in die gleiche Ordnung bringt (NFD), und eine andere Normalisierung, die alle Zeichen in der gleichen Ordnung kombiniert (NFC).

Manche visuell identischen Schriftzeichen tauchen in der Unicode-Tabelle mehrfach auf. Lateinisch groß A und Griechisch A („Alpha“) sind visuell gleich, aber im Computer durch zwei unterschiedliche Kodierungspunkte repräsentiert. Für die graphisch identischen Buchstaben im Griechischen und im Koptischen existierte ursprünglich nur ein einziger gemeinsamer Kodepunkt in der Unicode-Tabelle. Erst mit der Version 4.1 im Jahr 2005 wurden jeweils eigene Kodierungspunkte eingeführt.[3] Auch sind die meisten Ziffern auf Persisch und Arabisch vom Aussehen her identisch, doch sind sie in der Unicode-Tabelle alle doppelt kodiert, einmal für Persisch, einmal für Arabisch.

Bei der Projektplanung kann es sich bezahlt machen, nicht mit den interessantesten, bislang noch untranskribierten Texten zu beginnen, sondern umgekehrt, zunächst einfachere Texte oder bereits bestehende digitale Texte zu verwenden.

Oftmals ist der beste Ausgangspunkt, für alle Dokumente einer Schrift zunächst ein Grundmodell *(base-model)* zu trainieren, das möglichst viele Texte ausreichend aber nicht perfekt transkribiert, dieses Grundmodell dann auf wenige Seiten einer neuen Handschrift anzuwenden, diese Seiten zu korrigieren und dann das Grundmodell mit diesen neuen Daten nachzutrainieren *(finetuning)*, um es für speziell diese Handschrift zu optimieren. Wenn es schon anderweitig Transkriptionen gibt, kann mit Text-zu-Text-Alignierung dann noch schneller ein großes Trainingskorpus erstellt werden.

3 S. https://en.wikipedia.org/wiki/Greek_and_Coptic, zuletzt aufgerufen am 15.06.2024.

Neuronale Netzwerke, die mit rein visuellen Daten trainiert worden sind, erlernen nur ein sehr primitives Sprachmodell, und zwar die Wahrscheinlichkeiten, mit denen ein gewisses Zeichen zwischen zwei oder mehr anderen Zeichen erscheint. Man kann aber auch während des Trainings oder als separaten Post-OCR Schritt ein komplexeres Sprachmodell verwenden oder dazu trainieren. Dies birgt bei historisch variablen Orthographien allerdings die Gefahr der Hyperkorrektur, v. a. bei der Verwendung von auf modernen Texten trainierten Sprachmodellen.

Transkriptionsprojekte sind inzwischen sehr zahlreich und betreffen mehr und mehr Sprachen und Schriftarten. Das bislang vielleicht größte für die Theologie relevante Projekt ist das im Oktober 2023 beginnende ERC Synergy Projekt MiDRASH, welches anstrebt, einen Großteil der im KTIV Projekt der Nationalbibliothek Israels zusammengetragenen ca. 100 000 digitalen Handschriften zu analysieren und zu transkribieren. Viele Projekte stellen ihre Daten inzwischen im Katalog von *HTRUnited* ein.[4] Über die bereits genannten Projekte und Publikationen hinaus gibt es in der Patristik Projekte zu griechischen Texten in Berlin (von Stockhausen) und zu koptischen in Berlin (Lincke 2019, 2021), Oklahoma (Schroeder) und Tokyo (Miyagawa 2018, 2019, 2020). Zu armenischen und georgischen Handschriften und Drucken wird viel auf CALFA gearbeitet (Vidal Gorène 2021). Zu anderen Datensätzen siehe auch Nikolaidou et al. 2022. Bullingers Briefwechsel werden in einem von Tobias Hodel geleiteten Projekt analysiert (Scius-Bertrand et al. 2023, Ströbel et al. 2023).

10. Text-zu-Bild Alignierung

Für paläographische Studien aber auch für digitale Editionen ist die Text-zu-Bild Alignierung unabdinglich. Sie ermöglicht für eine Transkriptionszeile die approximativen Regionen für jeden Buchstaben und jedes Wort zu berechnen. Bei bestimmten neuronalen Netzwerken ist dies Teil der automatischen Transkription. Allerdings wird diese natürlich manuell korrigiert. Mit Text-zu-Bild Alignierung kann man jedoch die ungefähren Positionen der Buchstaben und der Worte nachträglich wieder berechnen. Dies geschieht unter anderem in Projekten zu Stundenbüchern von Dominique Stutzmann (Hazem et al. 2020), zu Qumran (Stökl Ben Ezra et al. 2020), zu rabbinischen Texten (Stökl Ben Ezra & Lapin, im Druck), sowie zur Hebräischen Bibel (Bambaci et al. 2024, Stökl Ben Ezra et al. 2021).

4 S. https://htr-united.github.io, zuletzt aufgerufen am 15.06.2024.

11. Text-zu-Text Alignierung

Eine praktische Methode, bestehende qualitativ hochwertige elektronische Texte für die Erstellung von Trainingsdaten zu verwenden ist die Text-zu-Text Alignierung *(text2text alignment).* Nach der Korrektur einer automatischen Layoutanalyse wird das beste vorhandene Texterkennungsmodell appliziert. Anschließend berechnet der Computer, wie man den elektronischen Text am besten mit der fehlerhaften automatischen Transkription aligniert und tauscht letzteren mit ersterem aus. Wenn keine Zeilen- oder Seitenumbrüche vorhanden sind, kann der Computer sie einfügen.[5] Ähnlich arbeitet das *Text2Image* Tool in Transkribus. In der Zukunft erlaubt die Verbindung mit ganzen Textkorpora dann quasi eine automatische Erstellung von Trainingsmaterial (Smith et al. 2023)

12. Von der HTR-Plattform zur Edition

Der Weg von einer automatischen Transkription zu einer digitalen Edition ist (noch) keine Autobahn, sondern mit Stolpersteinen bepflastert. In Editionen werden interlineare und marginale Verbesserungen zumeist mit Klammern als Zusätze markiert, aber in den Fließtext integriert. In Transkriptionsplattformen sind diese hingegen in eigenständigen Zeilen. Für die Verwendung von Kollationsprogrammen für kritische Editionen von Texten mit mehreren Handschriften ist ein Fließtext unerlässlich. Erste Ansätze zu Lösungen verspricht eine Pipeline für hebräische Texte (Stökl Ben Ezra 2022a).

Verwendete Literatur

Bambaci, L., Stökl Ben Ezra, D. (vrs. 2024). Enhancing HTR of Historical Texts through Scholarly Editions. A Case Study from an Ancient Collation of the Hebrew Bible. In *Computational Humanities Research Conference* 2023. *CEUR Workshop Proceedings 2023* (S. 554–576). Paris: Computational Humanities Research. URL: https://ceur-ws.org/Vol-3558/paper6310.pdf [zuletzt aufgerufen am 15.06.2024].

Beinert, W. (2021). Schriftstil. In Ders. (Hrsg.), *Das Lexikon der Typografie.* URL: https://www.typolexikon.de/schriftstil [zuletzt aufgerufen am 15.06.2024].

Breuel, Th. (2008). The OCRopus open source OCR System. In B. A. Yanikoglu & K. Berkner (Hrsg.), *Document Recognition and Retrieval XV, part of the IST-SPIE*

5 S. https://github.com/dasmiq/passim, zuletzt aufgerufen am 15.06.2024.

Electronic Imaging Symposium, San Jose, CA, USA, January 29–31. https://doi.org/10.1117/12.783598 [zuletzt aufgerufen am 15.06.2024].

Droby, A., Irina, R., Vasyutinsky-Shapira, D., Kurar-Barakat, B., & El-Sana, J. (2022). Digital Hebrew Paleography. Script Types and Modes, *Journal of Imaging,* 8(5.143), 1–22. https://doi.org/10.3390/jimaging8050143 [zuletzt aufgerufen am 15.06.2024].

Genette, G. (1982). *Palimpsestes. La littérature au second degré.* Paris: Éditions du Seuil.

Grüning, T., Leifert, G., Strauß, T., Michael, J., & Labahn, R. (2019). A two-stage method for text line detection in historical documents, *International Journal of Document Analysis and Recognition,* 22(3), 285–302.

Hazem, A., Daille, B., Kermorvant, Ch., Stutzmann, D., Bonhomme, M.-L., Maarand, M., & Boillet, M. (2020). Books of Hours. The First Liturgical Data Set for Text Segmentation. In *Proceedings of the Twelfth Language Resources and Evaluation Conference* (776–784). Marseille: European Language Resources Association. URL: https://aclanthology.org/2020.lrec-1.97 [zuletzt aufgerufen am 15.06.2024].

Kahle, Ph., Colutto, S., Hackl, G., & Mühlberger, G. (2017). Transkribus. A Service Platform for Transcription, Recognition and Retrieval of Historical Documents. In *14th IAPR International Conference on Document Analysis and Recognition* (S. 19–24). Kyoto: IEEE Xplore. https://doi.org/10.1109/ICDAR.2017.307 [zuletzt aufgerufen am 15.06.2024].

Kiessling, B., Tissot, R., Stokes, P. A., & Stökl Ben Ezra, D. (2019). eScriptorium. An Open Source Platform for Historical Document Analysis. In *International Conference on Document Analysis and Recognition Workshops (ICDARW)* (S. 19–24). Sydney: IEEE Xplore. https://doi.org/10.1109/ICDARW.2019.10032 [zuletzt aufgerufen am 15.06.2024].

Kiessling, B. (2020). A Modular Region and Text Line Layout Analysis System. In *17th International Conference on Frontiers in Handwriting Recognition (ICFHR)* (S. 313–318). Dortmund: IEEE Xplore. https://doi.org/10.1109/ICFHR2020.2020.00064 [zuletzt aufgerufen am 15.06.2024].

Lincke, E.-S., Bulert, K., & Büchler, M. (2019). Optical Character Recognition for Coptic fonts. A multi-source approach for scholarly editions. In *DATeCH2019 – Proceedings of the 3rd International Conference on Digital Access to Textual Cultural Heritage* (S. 87–91). New York: Association for Computing Machinery. https://doi.org/10.1145/3322905.3322931 [zuletzt aufgerufen am 15.06.2024].

Lincke, E.-S. (2021). The State of the affairs in optical character recognition (ocr) for Coptic. In C. G. Zamacona & J. Ortiz-García (Hrsg.), *Handbook of Digital Egyptology. Texts* (S. 139–164). Alcalá de Henares: Universidad de Alcalá.

Maier, A., Christlein, V., Breininger, K., Würfl, T., Unberath, M., & Riess, Ch. (2021). Bildanalyse. In G. Görz, U. Schmid & T. Braun (Hrsg.), *Handbuch der*

Künstlichen Intelligenz (S. 673–712). Berlin/Boston: De Gruyter. https://doi.org/10.1515/9783110659948-015 [zuletzt aufgerufen am 15.06.2024].

Miyagawa, S., Bulert, K., Büchler, M., & Behlmer, H. (2019). Optical character recognition of typeset Coptic text with neural networks, *Digital Scholarship in the Humanities,* 34(1), 35–41. https://doi.org/10.1093/llc/fqz023 [zuletzt aufgerufen am 15.06.2024].

Miyagawa, S., Zeldes, A., Büchler, M., Behlmer, H., & Griffitts, T. (2018). Building Linguistically and Intertextually Tagged Coptic Corpora with Open Source Tools. In Suzuki, Ch. (Hrsg.), *Proceedings of the 8th Conference of Japanese Association for Digital Humanities. Leveraging Open Data* (S. 139–141). Tokyo: Center for Open Data in the Humanities.

Miyagawa, S. (2020). Digitization of Coptic Manuscripts and Digital Humanities. Tools and Methods for Coptic Studies, *The International Journal of Levant Studies,* 2, 29–61.

Mühlberger, G., Seaward, L., Terras, M., Ares Oliveira, S., Bosch, V., Bryan, M., Colutto, S., Déjean, H., Diem, M., Fiel, S., Gatos, B., Greinoecker, A., Grüning, T., Hackl, G., Haukkovaara, V., Heyer, G., Hirvonen, L., Hodel, T., Jokinen, M., Kahle, Ph., Kallio, M., Kaplan, F., Kleber, F., Labahn, R., Lang, E. M., Laube, S., Leifert, G., Louloudis, G., McNicholl, R., Jean-Meunier, L., Michael, J., Mühlbauer, E., Philipp, N., Pratikakis, I., Puigcerver Pérez, J., Putz, H., Retsinas, G., Romero, V., Sablatnig, R., Joan-Sánchez, A., Schofield, Ph., Sfikas, G., Sieber, Ch., Stamatopoulos, N., Strauß, T., Terbul, T., Toselli, A. H., Ulreich, B., Villegas, M., Vidal, E., Walcher, J., Weidemann, M., Wurster, H., & Zagoris, K. (2019). Transforming scholarship in the archives through handwritten text recognition, *Journal of Documentation,* 75(5), 954–976.

Nikolaidou, K., Seuret, M., Mokayed, H., Liwicki, M. (2022). A survey of historical document image datasets, *International Journal of Document Analysis and Recognition,* 25(4), 305–338.

Quirós, L., & Vidal, E. (2022). Reading order detection on handwritten documents, *Neural Computer Applications* 34(12), 9593–9611.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain, *Psychological Review,* 65(6), 386–408. https://doi.org/10.1037/h0042519 [zuletzt aufgerufen am 15.06.2024].

Scius-Bertrand, A., Ströbel, Ph., Volk, M., Hodel, T., & Fischer, A. (2023). The Bullinger Dataset. A Writer Adaptation Challenge. In *ICDAR 2023: Document Analysis and Recognition. Conference Proceedings,* 1 (S. 397–410). San José: Springer. https://doi.org/10.1007/978-3-031-41676-7_23 [zuletzt aufgerufen am 15.06.2024].

Seuret, M., Nicolau, A., Rodríguez, D.-S., Weichselbaumer, N., Stutzmann, D., Mayr, M., Maier, A., & Christlein, V. (2021). ICDAR 2021 Competition on Historical Document Classification. In Lladós, J., Lopresti, D., Seiichi, U. (Hrsg.), *Document Analysis and Rcognition. ICDAR 2021* (S. 618–634). Lausanne: Springer. https://doi.org/10.1007/978-3-030-86337-1_41 [zuletzt aufgerufen am 15.06.2024].

Smith, D., Murel, J., Parkes-Allen, J., & Miller, M. T. (2023). Automatic Collation for Diversifying Corpora. Commonly Copied Texts as Distant Supervision for Handwritten Text Recognition. In *Computational Humanities Research Conference. CEUR Workshop Proceedings* 2023 (S. 206–221). Paris: Computational Humanities Research. URL: https://ceur-ws.org/Vol-3558/paper1708.pdf [zuletzt aufgerufen am 15.05.2024].

Stokes, P., Kiessling, B., Tissot, R., Gargem E., & Stökl Ben Ezra, D. (2021). The eScriptorium VRE for Manuscript Cultures. In C. Clivaz, & G. V. Allen (Hrsg.), *Ancient Manuscripts and Virtual Research Environments* (o. S.) [= *Classics@Journal* 18(1)]. URL: https://classics-at.chs.harvard.edu/classics18-stokes-kiessling-stokl-ben-ezra-tissot-gargem [zuletzt aufgerufen am 15.06.2024].

Stökl Ben Ezra, D., Brown-DeVost, B., Dershowitz, N., Pechorin, A., & Kiessling, B. (2020). Transcription Alignment for Highly Fragmentary Historical Manuscripts. The Dead Sea Scrolls. In *International Conference on Frontiers in Handwriting Recognition* (S. 361–366). Dortmund: IEEE Xplore. https://doi.org/10.1109/ICFHR2020.2020.00072 [zuletzt aufgerufen am 15.06.2024].

Stökl Ben Ezra, D., & Lapin, H. (im Druck). From HTR to Digital Critical Scholarly Edition. Reflexions on the Use of Machine Learning. Computational and Digital Humanities in the Sofer Mahir Project. In U. Henny-Krahmer et al. (Hrsg.), *Machine Learning and Data Mining for Digital Scholarly Editions.* Norderstedt: Books on Demand [= *SIDE*].

Stökl Ben Ezra, D., Lapin, H., Brown DeVost, B., & Jablonski, P. (2022a). HTR2CritEd. A Semi-Automatic Pipeline to Produce a Critical Digital Edition of Literary Texts with Multiple Witnesses out of Text Created through Handwritten Text Recognition. In *Digital Humanities 2022. Responding to Asian Diversity* (S. 690–691). Tokyo: DH2022 Local Organizing Committee. URL: https://dh2022.dhii.asia/dh2022bookofabsts.pdf [zuletzt aufgerufen am 15.06.2024].

Stökl Ben Ezra, D., Rustow, M., & Witty, D. (2022b). Segmentation Mode for Archival Documents with Highly Complex Layout. In *Conference Documents anciens et reconnaissance automatique des écritures manuscrites.* École national des chartes, Paris: YouTube. URL: https://www.youtube.com/watch?v=dE1XUXiuitU (7:07–7:30) [zuletzt aufgerufen am 15.06.2024].

Ströbel, Ph., Hodel, T., Fischer, A., Scius, A., Wolf, B., Janka, A., Widmer, J., Scheurer, P., & Volk, M. (2023). Bullingers Briefwechsel zugänglich machen. Stand der Handschriftenerkennung. In A. Busch, & P. Trilcke (Hrsg.), *DHd 2023. Open Humanities, Open Culture* (S. 98–102). Belval/Trier: Zenodo. https://doi.org/10.5281/zenodo.7688631 [zuletzt aufgerufen am 15.06.2024].

Wolf, L., Littman, R., Mayer, N., German, T., Dershowitz, N., Shweka, R., & Choueka, Y. (2010). Identifying Join Candidates in the Cairo Genizah, *International Journal of Computer Vision,* 94(1), 118–135.

Weiterführende Literatur

Camps, J.-B., Vidal-Gorène, Ch., & Vernet, M. (2021). Handling Heavily Abbreviated Manuscripts. HTR Engines vs Text Normalisation Approaches. In E. H. Barney Smith & U. Pal (Hrsg.), *Document Analysis and Recognition – ICDAR 2021 Workshops. ICDAR 2021* (S. 306–316). Cham: Springer [= *Lecture Notes in Computer Science,* 12917]. https://doi.org/10.1007/978-3-030-86159-9_21 [zuletzt aufgerufen am 15.06.2024].

Chagué, A., & Thibault, C. (2023). I'm here to fight for ground truth. HTR-United, a solution towards a common for HTR training data. In *Digital Humanities 2023. Collaboration as Opportunity.* Graz: Zenodo. https://doi.org/10.5281/zenodo.8107449 [zuletzt aufgerufen am 15.06.2024].

Perdiki, E. (2023). List of manuscripts containing John Chrysostom's Homilies and the relevant manual transcriptions, 1(2). *Zenodo.* https://doi.org/10.5281/zenodo.8102662 [zuletzt aufgerufen am 15.06.2024].

Popović, M., Dhali, M. A., & Schomaker, L. (2023). Artificial intelligence based writer identification generates new evidence for the unknown scribes of the Dead Sea Scrolls exemplified by the Great Isaiah Scroll (1QIsaa), *PLoS ONE,* 16(4. e0249769), 1–28. https://doi.org/10.1371/journal.pone.0249769 [zuletzt aufgerufen am 15.06.2024].

Thibault, C. (2022). You Actually Look Twice At it (YALTAi). Using an object detection approach instead of region segmentation within the Kraken engine, *Journal of Data Mining and Digital Humanities,* 1–13. https://doi.org/10.48550/arXiv.2207.11230 [zuletzt aufgerufen am 15.06.2024].

Vidal-Gorène, Ch., Dupin B., Decours-Perez A., & Riccioli T. (2021a). A Modular and Automated Annotation Platform for Handwritings. Evaluation on Under-Resourced Languages. In Lladós, J., Lopresti, D., Seiichi, U. (Hrsg.), *Document Analysis and Rcognition. ICDAR 2021* (S. 507–522). Lausanne: Springer. https://doi.org/10.1007/978-3-030-86334-0_33 [zuletzt aufgerufen am 15.06.2024].

Vidal-Gorène, Ch., & Decours-Perez, A. (2021b) A Computational Approach of Armenian Paleography. In E. H. Barney Smith & U. Pal (Hrsg.), *Document Analysis and Recognition – ICDAR 2021 Workshops. ICDAR 2021* (S. 295–305). Cham: Springer [= *Lecture Notes in Computer Science,* 12917]. https://doi.org/10.1007/978-3-030-86159-9_20 [zuletzt aufgerufen am 15.06.2024].

Wick, Ch., Reul, Ch., & Puppe, F. (2018). Comparison of OCR Accuracy on Early Printed Books using the Open Source Engines Calamari and OCRopus, *Journal of Language Technology and Computational Linguistics,* 33(1), 79–96.

Bildnachweise

Abb. 1: Abraham Lincoln als Graufstufenbild. CC0. Das Bild dürfte ursprünglich auf Leon Harmon zurückgehen, der 1971 herausfinden wollte, auf wieviel visuelle Information ein Bild verzichten kann, um noch erkennbar zu sein.

Abb. 2: Wikipedia © „BrunelloN“ CC-BY-SA 4.0

Abb. 3: Screenshot Daniel Stökl Ben Ezra. Handschrift © CC-BY Cambridge University Library

Funded by the European Union (ERC, MiDRASH, Project No. 101071829). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.