Vom Bild zum Text – praktische OCR für die DH

—

05.05.2021, 15–17 Uhr: OCR-D und OCR4all

Evenreihe

- Dienstag, 23.03.2021, 10–12 Uhr: Einführungsveranstaltung (Interessen/Bedarfe und OCR-Grundlagen)
- **Mittwoch, 05.05.2021, 15–17 Uhr: OCR-D, OCR4all, TEI-Konvertierung**
- Mittwoch, 12.05.2021, 15–17 Uhr: Evaluation, Transkription, Training
- Mittwoch, 19.05.2021, 15–17 Uhr: Postcorrection, Hackathon
- Mittwoch, 15.09.2021, 14–16 Uhr: Abschlussveranstaltung

OCR-D

Projekthistorie

DFG-Projekt seit 2015 mit Ziel: Volltextdigitalisierung der VD

- 2015–2017: Bestandsaufnahme
- 2018–2020: Entwicklung von Prototypen, acht Satellitenprojekte
- 2021–2023: Überführung in Produktivbetrieb, sieben Satellitenprojekte

Wolfenbüttel, Berlin, München, Karlsruhe, Göttingen, Leipzig, Mannheim, Erlangen, Mainz, Würzburg, Kaiserslautern, Halle, Dresden…

Kurzes Glossar

- **Prozessor**: Spezifikationsgemäß implementiertes Tool, das eine Aufgabe im Workflow übernimmt
- **Workspace**: Verzeichnis mit einer `mets.xml`, die das Werk beschreibt und in der alle Ergebnisse verzeichnet werden
- **File Group**: Eine Gruppierung von Dateien in der `mets.xml`, entspricht weitgehend lokalen Verzeichnissen
- **Parameter**: Tool-spezifische "Knöpfe an denen man drehen kann", um die Verarbeitung zu steuern
- **GPU**: Grafikkarten, die NVIDIA CUDA unterstützen, können Prozessoren, die auf neuronalen Netzen beruhen (die Mehrheit), enorm beschleunigen

OCR-D Prioritäten

- Massenverarbeitung > Ergonomie
- Transparenz > Perfektion
- Komponenten > Black Box
- Spezifikationen > Konventionen

Massenverarbeitung

- Auf Standards aufbauen (METS/MODS, PAGE-XML, JSON-Schema, XML Schema, ALTO-XML)
- Kommandozeilentools mit einheitlichem Interface
- Anschlussfähige Technologien (Python, Git, Docker)
- Austauschbare Komponenten und flexibel konfigurierbare Workflows

Transparenz

- Release early, release often: https://github.com/OCR-D
- Offener Chat: https://gitter.im/OCR-D/Lobby (und https://gitter.im/ag-ocr/community für die DHd AG OCR)
- Regelmäßige, offene, virtuelle Treffen:
  - zweiwöchentlich Mittwoch 2pm CET: Offener OCR-D Tech Call https://hackmd.io/OOMgg3ZeSqK4vfKL1wRbwQ (Technischer Austausch)
  - monatlich jeden 1. Freitag: OCR-D & Co https://ocr-d.de/en/2021/04/26/barcamp.html (Barcamp-Format für OCR-D Veteran:innen und Neulinge)
- Wiki für nutzergetriebene Dokumentation: https://github.com/OCR-D/ocrd-website/wiki
- Im Allgemeinen Kommunikation auf Englisch

Komponenten

- Gängige OCR-Engines führen verschiedene Operationen vor und nach der "eigentlichen" Texterkennung aus - nutzerfreundlich aber ungünstig für Perfektionist:innen wie uns
- 1 Schritt im Workflow → 1 Aufruf eines OCR-D Prozessors
- Alternative Implementierungen für Prozessoren

"Nimm die Binarisierung von Ocropus, die Segmentierung von tesseract und die Texterkennung von Calamari"

Spezifikationen

- Verbindliche Vorgaben wie sich die Prozessoren verhalten müssen
- Einheitliche Kommandozeilenschnittstellen
- Mit Schemasprachen Datenaustausch verifizieren
- Konventionen explizit und validierbar machen
- Referenzimplementierung und Tools zum Entwickeln von spezifikationsgemäßen Prozessoren: https://github.com/OCR-D/core

Installation

- Meta-Projekt https://github.com/OCR-D/ocrd_all das alle OCR-D Prozessoren beinhaltet
- Zielsystem: Ubuntu 18.04
  - In Windows 10 per WSL installierbar
  - Mac OSX: weitgehend kompatibel, aber nicht garantiert
- Auch als Docker-Container verfügbar
  - plattformunabhängig, funktioniert in Windows, OSX, Linux, BSD ...
  - GPU-Nutzung nicht trivial

Dokumentation finden

- alle Prozessoren haben ein mal mehr mal weniger detailliertes README
- alle Prozessoren unterstützen `--help`
  - bspw: `ocrd-tesserocr-recognize --help`
- Setup Guide beschreibt Installation
- Workflow-Guide beschreibt Zusammenspiel der Prozessoren
  - Klick auf den Prozessor-Namen öffnet Parameterliste
  - Beispiele für komplette Workflows am Ende der Seite
- Fragen? Probleme? https://gitter.im/OCR-D/Lobby

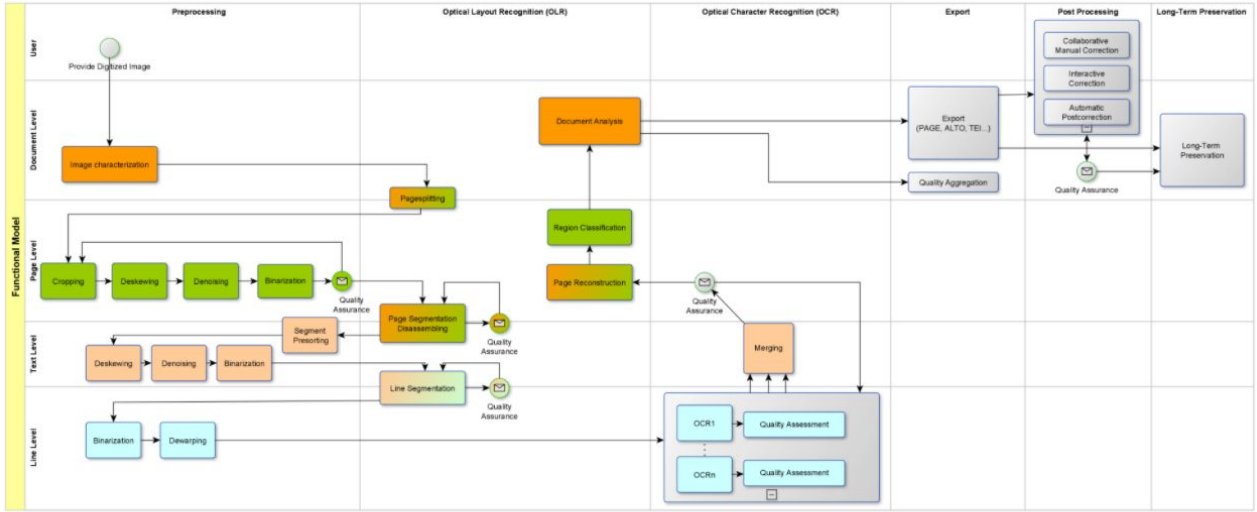
- **Preprocessing**: Binarisierung, Deskewing, Despeckling, Dewarping
- **Segmentierung**: Erkennung von Rändern, Regionen, Zeilen, Marginalien, Überschriften, Lesereihenfolge, ...
- **Texterkennung**: die "eigentliche" OCR
- **Evaluation und Nachkorrektur**: Erkennen und ggf. korrigieren von OCR-Fehlern (aber nicht historische Schreibweisen!)

Workflow-Schritte

< >
Workflows
Image Optimization (Page Level)
Step 0.1: Image Enhancement (Page Level, optional)
Available processors
Step 0.2: Font detection
Available processors
Step 1: Binarization (Page Level)
Available processors

Workflows

There are several steps necessary to get the fulltext of a scanned print. The whole OCR process is shown in the following figure:



The following instructions describe all steps of an OCR workflow. Depending on your particular print (or

Live Demo

https://github.com/kba/vdhd-2021-05-05

OCR4all

TEI-Konvertierung

Formate

PAGE (Page Analysis and Ground-Truth Elements): “XML-based page image representation framework that records information on image characteristics (image borders, geometric distortions and corresponding corrections, binarisation etc.) in addition to layout structure and **page content**”

TEI-XML (Text Encoding Initiative): “standard for the representation of **texts** in digital form”

Anforderungen

PAGE:

- Position von Zeichen (aus OCR und/oder Transkription) auf Seitenbildern
- Gliederung in Zeilen und Regionen
- Reihenfolge von Zeichen, Zeilen, Regionen

TEI-XML:

- Text und unterschiedlich komplexe Annotation
- Metadaten
- human readable/editable?

Möglichkeiten der Abbildung

- minimale Konvertierung: PAGE → <body> [TXT] </body>

- den Anforderungen angemessene Konvertierung: nach Bedarf Auszeichnung von Textstruktur, Integration möglichst vieler aus dem Layout erschließbarer Informationen über das Dokument

- maximale Konvertierung: PAGE → “Digital Facsimile”, “Embedded Transcription” in TEI-XML (vgl. fontane-nb.dariah.eu)

Welche Informationen sind im PAGE enthalten?

minimal:

- Zeilen und ihre Position auf der Seite

zusätzlich:

- Typen von Regionen?
- Reading Order der Zeilen/Regionen?
- Zuordnung von Initialen und Marginalien?
- Kommentare zu Zeilen/Wörter?

Wie verlässlich sind diese Informationen im Einzelfall, welche können mit geometrischen/textbasierten Heuristiken wie genau rekonstruiert werden, welche sollen übernommen werden?

Welche Informationen sollen ins TEI-XML?

- welches TEI-Subset?
- Zeilen oder Verse?
- Zeilennummerierung?
- Zeilenumbrüche und Art der Trennung?
- Zuordnung von Initialen und Marginalien?
- Fußnoten: Zuordnung, Weiterführung?
- Tabellen?
- Informationen über Bilder?
- Koordinaten in <facs>-Block?

Fazit: in der Praxis

- projektinterne Konventionen für die PAGE-XML aufstellen
- welche Informationen sollen im Zuge der Konvertierung hinzukommen?
- welche Kontrollheuristiken sollen bei der Konvertierung angewendet werden?
- je nach Homogenität des Materials kann eine spezifische Anpassung der Skripte auf das einzelne Werk sinnvoll sein
- Übernahme von XML-IDs (Seite+Zeile) ermöglicht Rückverfolgung
- **Werkzeuge**: XSLT (https://github.com/dariok/page2tei), Python/lxml, plain text/regex…