Für das Landesarchiv Baden-Württemberg entwickelte Stephan Lenartz im Rahmen seiner Masterarbeit einen Workflow zur automatisierten Aufbereitung und Bewertung einer digitalen fotografischen Sammlung. Exemplarisch werden Anwendungsmöglichkeiten der Programmiersprache Python zum skriptbasierten Umgang mit Dateisammlungen aufgezeigt. Der flexible, plattformunabhängige Ansatz soll zur Nachahmung anregen.

10 - Aufbereitung und Bewertung digitaler Sammlungen mit Python

Gründe: Warum sich die Anwendung von Python lohnt

Was ist Python?

Python ist eine höhere Programmiersprache, die seit 1991 auch mit dem Ziel der einfachen Erlern- und Anwendbarkeit entwickelt wird. Python ist Open-Source. Weit verbreitet ist die Anwendung beispielsweise im Bereich Data Science und in der Analyse wissenschaftlicher Forschungsdaten. Zahlreiche Erweiterungs-Bibliotheken machen Python über diese Anwendungsfelder hinaus besonders auch deshalb attraktiv, da mit Skripten in vergleichsweise kompakt geschriebenem Code flexibel auf spezifische Automatisierungs-Anforderungen Bezug genommen werden kann.

Was ist Python nicht?

Ein bei nicht vorhandenen Programmierkenntnissen leicht und ohne tiefere Einarbeitung anwendbares Universalwerkzeug.

Der Einsatz von Python zur Aufbereitung digitaler Sammlungen

Im Rahmen ausgiebiger Tests zur Entwicklung eines geeigneten Workflows für die Bearbeitung einer digitalen Fotosammlung kristallisierten sich beim sukzessiven Einsatz unterschiedler Tools drei grundsätzliche Probleme heraus:

1. Funktionalität: Abhängigkeit von der Verfügbarkeit geeignet erscheinender Tools
2. Oftmals fehlender Datenexport zu Auswertungs- und Dokumentationszwecken oder
3. Wenn Datenexport möglich, nur in voreingestelltem, sich von Tool zu Tool unterscheidendem Format

Schlussfolgerungen: *Allgemein:* Je mehr Tools eingesetzt werden, desto schwieriger gestaltet sich die notwendige Systematisierung des Vorgehens. *Konkret*: Problematisches Datenmanagement, aufgrund der fehlenden oder nicht-einheitlichen Datenexportmöglichkeiten. D.h. einerseits lückenhafte oder zumindest uneinheitliche Dokumentation und zudem eng gesteckte Grenzen einer Automatisierung. Auch vor dem Hintergrund der im zugrundeliegenden Fall gewünschten Reduktionsrate (90%) erschien daher der Einsatz einer vom Ansatz her flexibleren, d.h. dynamischeren und mächtigeren Lösung, wünschenswert.

Anwendungsbeispiel: Digitaler Fotonachlass

Ausgangslage

Die bearbeitete Dateiablage, der Vorlass eines Amateurfotografen, umfasste rund 200.000 Digitalfotos im JPEG-Format. Übernommen wurde eine externe Festplatte (HDD, USB-Schnittstelle). Die vorab als archivwürdig bewertete Sammlung sollte für den Ingest aufbereitet werden. Es wurde im Zuge der weiteren Bewertung eine Reduktion des Datenvolumens von ursprünglich rund 450 Gigabyte um 90% angestrebt. Da ein Bearbeiten dieser und vergleichbarer Dateiablagen mit herkömmlichen, weitgehend manuellen Mitteln Archive erfahrungsgemäß an die Grenzen des Leistbaren bringt, sollte ein Verfahren mit Ansätzen der Automatisierung entwickelt werden.

Struktur der Sammlung

Die Sammlung wurde vom Fotografen ohne die Hilfe einer speziellen Fotoverwaltungssoftware auf einem Windows-System angelegt. Sie bestand zu 99,3 % Prozent aus JPEG-Dateien, die sich in insgesamt 1664 Verzeichnissen befanden. Mit einer maximalen Schachtelung von drei Verzeichnissen wies die Sammlung eine vergleichsweise flache Struktur auf. Die 0,7 % Nicht-

Bilddateien konnten mit Hilfe von TreeSize Professional (TSP) schnell identifiziert und bewertet werden, sodass sich die weitere Bearbeitung im Wesentlichen auf die Bilddateien konzentrieren konnte.

Eingesetzte Werkzeuge

Der finale Workflow zielte aus oben genannten Gründen auf den Einsatz weniger Tools ab. Neben TSP wurde auf einige bewährte Anwendungen zurückgegriffen: Textverarbeitungs- und Tabellenkalkulationsprogramme (Microsoft Office / LibreOffice), Notepad++, IrfanView und TotalCommander. Die automatisierbaren Schritte wurden allesamt in der Programmiersprache Python (Release 3.6.1.) realisiert. Insgesamt kamen 16 Skripte zum Einsatz, die zusätzlich zur Python-Standardbibliothek folgende Bibliotheken einbanden: natsort, numpy, opencv, pillow, pytesseract.

Bearbeitungsgrundsätze

Zwei Prinzipien wurden der automatisierten Vorgehensweise zugrunde gelegt. **1. Nachvollziehbarkeit / Dokumentation**: Sämtliche Bearbeitungsschritt sollten dokumentiert werden. Eine einheitliche Dokumentation wurde mittels CSV-Logdateien realisiert. **2. Korrigierbarkeit**: Jeder ausgeführte Schritt sollte reversibel sein. Das implementierte *Dateimanagement* erlaubte es, Zustände vor einem jeweiligen manipulativen Eingriff automatisiert wiederherzustellen. Darüber hinaus sollten händische Korrekturen jederzeit möglich sein.

Mit Python (teil-)automatisierte Bearbeitungsschritte

Löschen nicht archivwürdiger & Erhalten ganzer Verzeichnisse

Die Ordnerbezeichnungen gaben Aufschluss über Aufnahmeorte, Ereignisse und anhand von Vermerken zu fremden Fotografen auch über den Rechtsstatus enthaltener Fotografien. Per Skript wurden sämtliche Ordnerbezeichnungen extrahiert, anschließend mittels Stichwortsuchen diejenigen Verzeichnisse identifiziert, die aufgrund zuvor festgelegter Bewertungskriterien (Kassationskriterien: Keine Rechte, Aufnahmen außerhalb Baden-Württembergs; Erhalte ganzer Verzeichnisse bei Sonderfällen wie Luftbildserien) vollständig kassiert oder archiviert werden sollten und diese in gesonderten Listen abgelegt. Durch weiteren Skripteinsatz wurden diese Verzeichnisse während des Workflows entsprechend behandelt.

Löschen nicht archivwürdiger Dateiformate und leerer Ordner

Die als nicht archivwürdig erachteten Dateiformate (v.a. Systemdateien) und leere Verzeichnisse wurden durch Skripteinsatz automatisch entfernt.

Redigieren von Verzeichnisnamen

Extraktion sämtlicher Verzeichnisnamen. Anwendung der automatischen Rechtschreibkorrektur auf extrahierte Liste, Auflösen von Abkürzungen mittels „Suchen u. Ersetzen". Anschließend den vorgenommenen Korrekturen entsprechende automatisierte Umbenennung aller Verzeichnisse.

Einebnen der Verzeichnisstruktur (SIP-Fokus)

Die Verzeichnisstruktur wurde per Skript auf eine zuvor bestimmte Ebene reduziert, um SIPs zu formieren. Tiefer liegende Verzeichnisse wurden automatisch aufgelöst, darin liegende Dateien jeweils auf die zu erhaltende Ebene verschoben und mit einem dynamischen Präfix (nach dem Muster: „Ordnerbezeichnung(en) des/r aufgelösten Verzeichnis(se)" + „_" + „Dateiname") versehen.

Zufallssampling

Es wurde ein randomisiertes Samplingverfahren auf Verzeichnisebene ausgeführt. Aufgrund der durchgängig einheitlichen Binnenstrukturierung wurde eine gewisse Anzahl der am Anfang eines jeden Verzeichnisses liegenden Bilder erhalten und anschließend jedes x-te Foto zur Archivierung ausgewählt.

Löschen unscharfer Fotos

Automatisiertes Erkennen unscharfer Fotografien. Nach Begutachten und Nachselektion des Suchergebnisses (automatisches Öffnen in IrfanView-Thumbnail-Ansicht) erfolgte per Skript das Löschen der nunmehr erstellten Bildauswahl.

Automatisierte Selektion bestimmter Bilder mithilfe von Referenzen

Vorab wurde bestimmt, dass Bilder eines bestimmten Typs (hier: Einfarbiger Hintergrund mit Textinformation) Teil der zu archivierenden Sammlung sein sollten. Mittels zuvor angelegter Referenzen wurde die Sammlung nach diesen Bildern per Skript durchsucht und die gefunden Bilder archiviert.

Texterkennung

Der auf bestimmten Bildern (siehe vorheriger Schritt) enthaltene Text wurde per Skript erkannt (OCR) und zu Erschließungszwecken abgelegt.

Ergebnis

Nach Abschluss der Vorbereitungen (Entwickeln des Workflows und Schreiben der Skripte) konnte die Sammlung binnen dreier Arbeitstage aufbereitet und bewertet werden. Nach der Begutachtung des Ergebnisses wurde der Ingest in DIMAG und ScopeArchiv ausgeführt.

Potenziale: Übertragbarkeit des Ansatzes

Der Einsatz von Python-Skripten zur Aufbereitung und Bewertung eines fotografischen Vorlasses am Landesarchiv Baden-Württemberg dürfte in ähnlicher Form auf andere Sammlungen anwendbar sein. Seine Stärke liegt in der Flexibilität, die es erlaubt, auf individuelle Eigenschaften einer Sammlung zu reagieren und Aufbereitungs- wie auch Bewertungsprozesse in einem einheitlichen Verfahren zu bewerkstelligen. Zudem sichert Python eine weitgehende Systemunabhängigkeit. Arbeitsabläufe können dementsprechend leicht auf verschiedene Systemumgebungen (Windows, Linux, Mac) übertragen werden. Über den realisierten Umfang hinaus wären im Rahmen der Fotobewertung beispielsweise Anwendungsszenarien mit implementierter Gesichtserkennung, Rückgriff auf EXIF-Metadaten und ggf. eine verstärkte Kombination verschiedener Bewertungskriterien denkbar. Weiterhin ist auch die Anwendung auf andersartige Dateisammlungen möglich, wobei die Anforderungen wiederum je nach Dateizusammensetzung näher zu bestimmen sind. Auch eine Implementierung einzelner Bearbeitungsschritte mit Python in bereits bewährte Arbeitsabläufe ist denkbar.

Stephan Lenartz
mail@stephanlenartz.de