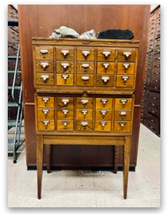
An open-source OCR workflow for digitizing legacy card catalogs

Anna Jerve, Chloe Cheng & Christine N. Garcia
Stanford University, Stanford, CA
Corresponding author: ajerve@stanford.edu
0000-0002-9276-1346, 0009-0005-2205-1284, 0000-0002-9728-3670

Background



- Stanford's mineral collection established with the first professor, JC Branner in 1891
- Medium-sized research & teaching collection
- Lack of dedicated staff for several decades despite continued growth

Purpose

To **digitize**, **OCR**, and database Stanford's mineral collection's original typed card catalog. Once **cleaned**, the resulting dataset will serve as a tool for **inventorying** the collection and be the backbone of our mineral database.

Benefits

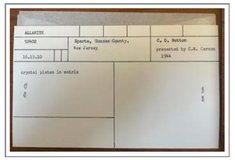
- Obtain structured data from analog source
- Create shareable, inexpensive and reusable open source tool

Stats

Total number of card catalog cards: **17,530**
Python script development & testing: **60 hrs**
Preparation: **60 hrs**, ~2500-3000 stapled cards
Scanning: **150 minutes** (5 min/drawer)
Processing through Python: **24 hrs**
Data **cleaning**: **150 hrs**

Inventorying: **to date, 1050 specimens, 2.5 high school interns, 288 hrs combined**

1 Preparation



Card catalog card with stapled envelope holding label on back

Staples removed, catalog number added

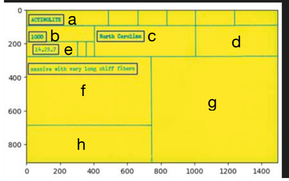
2 Scanning

- Fujitsu fi-7700 scanner
- File type: tif
- Color image
- Resolution: 300 dpi

3 Processing

- Input requirements pre-cropped OpenCV-readable images, stored in Google Drive
- Output: data recorded on mineral card will import automatically into Google Sheets; original image files renamed to match catalog number
- Image characteristics (contrast, color calibration) temporarily modified during the process to optimize OCR output

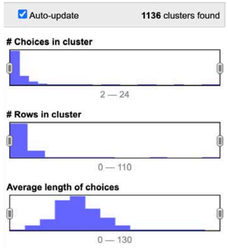
```
img_path = os.path.join(scan_dir, original_filename)
img = cv2.imread(img_path, 0)
(height, width) = img.shape
print(f'Scan {i+1} of {len(scans_list)}. Processing image "{original_filename}".')

lines = resize_lines_template(img, template) # using template card image
no_lines = remove_lines(img, lines)
bboxes = get_bounding_boxes(img, no_lines)
```




```
# Read contents of each field
a species = ''
b catalog_num = ''
c locality = ''
d source = ''
e chemical_index = ''
f description_1 = ''
g description_2 = ''
h description_3 = ''
```

Python
Tesseract 0.3.13

4 Cleaning



- Data clustered by mineral name, locality, source & description (1,2,3)
- Mineral name and locality columns organized by number of occurrences. Any with < 5 occurrences were manually checked
- All catalog numbers manually checked against original scanned image
- All blank mineral names and localities manually checked

OpenRefine

5 Inventorying

- Specimens rehoused
- Staples on labels removed
- Label information captured verbatim in spreadsheet, incl. those affixed to specimen
- 10% of inventoried specimens, to date, not in card catalog and entered manually

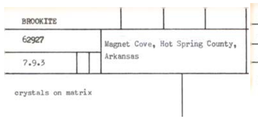
Problems

1. Numbers were read and parsed incorrectly and inconsistently, so each catalog number had to be manually checked

	❌	✅
MICROCLINE	26,300	**28300**
MICROCLINE	28,553	**28333**
MICROCLINE	29,725	**29723**
MANGANOPHYLLITE	re65,92516.15.13	**63925**
MANGANOPHYLLITE	oyeee16.16.15	**63926**
MANGANOPHYLLITE	63,92716.16.13	**63927**

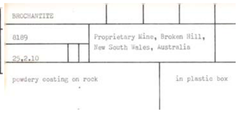
2. Correct output was dependent on how high the contrast was between the card and the text, but this was often inconsistent and unpredictable

High contrast



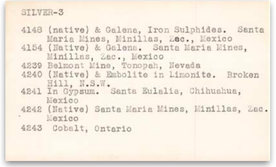
Name: Brookite
Catalog #: [not recorded]
HEY #: 709.5
Location: Magnet Cove, HotSpringCounty,Arkansas
Description: Crystals on matrix

Low contrast



Name: Brochantite
Catalog #: 6189
HEY #: [not recorded]
Location: Proprietary kane, Broken dill,New South wales, australia
Description: Powdery coating on rock
Description 2: in plastic box

3. Some cards did not follow the standardized formatting and required manual transcription



Future directions

- Improve code/procedure for future **processing**
- Ongoing **inventorying**
- Reconcile labels with specimens
- Standardize fields for import into database