Final Report: Transforming Legacy Engineering Reports into Useful Machine Readable Information

August 2023 - November 2023

Authors: Sandya Wijaya, Giovanni Celis Hernandez, Peiying Guan
Mentors: Kimberly Baldwin, Kevin Smith, Nadia Mondegari

Quick Links

Poster: BOEM Poster
Meeting Notes: Meeting Notes
Master folder with all code notebooks: BOEM Data Discovery FA23

Background

About BOEM
The mission of the Bureau of Ocean Energy Management (BOEM) within the Department of the Interior (DOI) is to manage development of the U.S. Outer Continental Shelf (OCS) energy and mineral resources in an environmentally and economically responsible way. The Pacific Region manages these resources in federal waters off the coasts of California, Oregon, Washington, and Hawaii. In the Office of Strategic Resources, part of our responsibility is to manage existing oil and gas leases, and appropriate access to oil and gas resources through resource evaluation, reserves estimation, and geological and geophysical data acquisition.

Drill Stem Test
A drill stem test measures a formations' pressure and fluids and can return data on formation properties such as permeability, flow rate, and estimates of productivity. The test data that is typically returned includes a date, elapsed time, temperature, and pressure. This data is extremely useful for evaluating a reservoir as well as certain geologic characteristics of the offshore. Not only does this information support ongoing oil and gas resource evaluation, but it can be used in new evaluations of potential carbon storage targets.

Starting Point: Past Work
BOEM's well test data is contained within many scanned pdfs from the 1970s and 1980s. Sometimes the computer can read the text, other times numbers are difficult to read to the human eye. Last semester, students at UC Berkeley worked on preprocessing images of tables in a higher quality well test report using OpenCV.

Abstract

The project objective is to extract data from diverse well test reports and convert this information into a machine-readable format. Divergent styling, fonts, headers, and subsections are prevalent, resulting in the difficult standardization task.

The documents exhibit various forms of degradation, such as smudging, blurring, and special characters. Recognizing this, we focus on the development of a robust model capable of adapting to and functioning with the considerable diversity inherent in the well test reports. We hoped to create a model to address the challenges posed by variations in document presentation and accommodate the full spectrum of document quality. This involved leveraging Optical Character Recognition (OCR) and complementary methodologies to extract data reliably and enhance the interpretability of information from documents that deviated from standard templates.

The end product is an accurate database of subsurface pressure and temperature information, linked to a specific location and depth and collected at a specific time, which can be used to evaluate reservoir properties for oil and gas and carbon storage assessments. Such achievement and insight is especially valuable because these well test reports are legacy data – it will likely never be collected in the same location again – and so if it was not made machine readable, it will not be available to future users.

Condensed Ver.

The project objective is to extract data from historical well test reports and convert this information into a machine-readable format.

- Divergent styling, fonts, headers, and subsections are prevalent, resulting in the difficult task of standardization.

Our focus is on developing a robust model adaptable to the inherent diversity in well test reports, addressing presentation variations.

- Leveraging optical character recognition, machine learning and computer vision techniques, we aim to extract data reliably, enhancing interpretability.

The end goal is an accurate database of subsurface pressure and temperature information linked to specific locations and depths, vital for evaluating reservoir properties in oil, gas, and carbon storage assessments.

- This achievement is particularly valuable as well test reports contain legacy data crucial for future usages and analysis.

Methods

Various methods

Overall Comparison Table

	Tabula	**pdftools in R**	**Custom Box Model**
Input type	PDF	PDF	Image
Accuracy	Depends on quality and	Depends on quality of	Tested and proven to be

	structure of input (as no preprocessing step)	input (as no preprocessing step)	accurate, especially if training data is expanded
Computation power	Low (rely on app)	Medium (read all texts at once)	High (trained on 10 pages)
Time needed	Low (< 2 min. Mostly for selecting areas for desired contents)	Low (generally < 2 min, varied on the pdf size)	High (> 1 hour to train, > 10 hours to create datasets)
Can it be expanded to other documents?	Ye s(i.e. csv, zsv, json, zip, script)	Yes (w/ text cleaning & table reconstruction)	Yes

1. Tabula

Link: R_and_Tabula.ipynb

Tabula is a free third-party tool designed for extracting tables from PDFs and converting them into machine readable formats like CSV and JSON (tabula.technology).

Main Steps:

- Download and install Tabula app(download here)
- Upload the pdf to Tabula app
- Manually select the areas from which we want to extract the contents
- Export the output in a format that you need (i.e. csv, zip, json)

Pros & Cons:

- Pros: It is free. Processes are easy and straightforward. Output could be exported in different formats.
- Cons: Needs manual input, and the outcome highly depends on the quality.

2.“pdftools” library in R

Link: Text_Extraction_Using_R.html

Main steps:

- Use pdf_text function to extract all text from a pdf (input is a pdf)
- Remove multiple spaces
- Use Regex to split and select the information that we need
- Output the list of text, and each element should be the content of a table that we want
- Further clean the text (hard to achieve due to the poor quality of pdf)
- Convert text to dataframe (haven’t got this far)

Pros & Cons:

- Pros: Input is a pdf file, so we don’t need to convert the pdf to images.
- Cons: The performance of the Regex method and pdf_text function highly depends on the quality of input.

3. Custom Box Training and Computer Vision (OCR)

This method draws upon principles of computer vision and machine learning.

This method involves creating a pre-trained Tesseract model in order to utilize a OCR system that passes this model as a parameter and returns an optimized output. By gathering and structuring each individual image file we were able to generate and annotate box files that represented identifiable characters such as letters, numbers and symbols. After manually correcting these box files we were able to execute a multi-step training process using tesseract commands. Our final output would be the obtained table as a dataframe from the respective document.

Steps Involved:

Gathering and Naming Image Files:

- Select and compile image files that accurately represent the data your model is likely to encounter.
- Create a structured file system and naming convention for images.

Generating Box Files:

- Utilize Tesseract to automatically generate box files, outlining the characters in the images.
- Recognize that this process may not be 100% accurate, prompting the need for subsequent manual annotation.

Annotating Box Files:

- Employ tools liked for manual annotation.
- Correct inaccuracies in box annotations and add missing ones to enhance model accuracy.

Training Tesseract:

- Read image and box files, creating a tuple for each.
- Execute Tesseract commands to generate training data, including .tr files and a unicharset file.
- Create a font_properties file, specifying font properties for each font in the images.
- Combine files and execute training commands (mftraining and cntraining).
- Generate a .traineddata file for the trained language and integrate it with the Tesseract source directory.

Final Implementation:

- Deploy trained model using Tesseract with the -lang option for enhanced character recognition.

Pros & Cons:

Pros:

- Very reliable when it comes to accurate results
- Allows for a pre-loaded accuracy function to test output on
- Adaptable model for documents with similar styling and subsection organization
- Potential for improving the model even more by adding more training data

Cons:

- Manually editing the box files takes an extensive amount of time to complete
- Training the custom model is computationally expensive and time consuming
- Model still struggles with generalizing diverse unseen data
- Dependent on adequate training data, more training data means more accurate results

Miscellaneous achievements

Function to score how many words detected

Link: [Gio] function to get % of words detected.ipynb

We have created a function called 'accuracy_on_keywords(ocr_result)' which takes in an OCR result as input, and outputs the accuracy of detecting specific keywords within it.

Keywords are pre-defined to be *{"TIME": 7, "FIELD": 1, "REPORT": 1, "#": 3, "TEST": 2, "PHASE": 2, "OF": 2, "DAY": 2, "ELAPSED": 2, "FLOW": 1, "SHUTIN": 1, "FINAL": 1, "FLOW": 1, "PAGE": 1, "BOT": 4, "HOLE": 4, "he.*o": 107}*. Then, the function iterates through each key in the keywords dictionary and uses regular expressions to find how many times keywords appear in the input.

Finally, it prints out 1) the correct sum of keywords, 2) the detected OCR keywords, and 3) the calculated accuracy as a percentage, which is a percentage based on how many keywords were correctly identified in the OCR result.

$$\text{accuracy} = \frac{\text{correctSum} - \text{absDifference}}{\text{correctSum}} \times 100$$

Notebook to choose pages from PDF

Link: [Sandya] pdf_to_needed_table_image.ipynb

In order to determine the pages that need to be fed into the model, we follow a 3-step process for each PDF. Confusion matrices and accuracy details are provided for each strategy attempted to show performance.

1. Conversion: Convert every page in the PDF into an image format using the pdf2image Python library.

2. Identification of Tables: Analyze image content for the presence of tables using any of the below methods

- Assess the amount of black on the image; higher black content typically indicates more information and likely represents a table – accuracy of 45%

```
[[11  5]
 [18  8]]
              precision    recall  f1-score   support

           0       0.38      0.69      0.49        16
           1       0.62      0.31      0.41        26

    accuracy                           0.45        42
   macro avg       0.50      0.50      0.45        42
weighted avg       0.53      0.45      0.44        42
```

- Use Canny Edge Detection to identify well-defined edges in the image, a signifier of a table's presence – accuracy of 62%

```
[[ 0 16]
 [ 0 26]]
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        16
           1       0.62      1.00      0.76        26

    accuracy                           0.62        42
   macro avg       0.31      0.50      0.38        42
weighted avg       0.38      0.62      0.47        42
```

- *Other methods I did not implement but potentially successful: Pattern matching, Display each page in a condensed form for a quick manual review, Utilize a pre-trained deep learning model like YOLO.*

3. Filtering: Filter and retain only those pages containing the desired tables by filtering in desired PDFs with 'test phase' markers using any of the following methods

- Simple OCR – accuracy of 83%

```
[[21  0]
 [ 7 14]]
              precision    recall  f1-score   support

           0       0.75      1.00      0.86        21
           1       1.00      0.67      0.80        21

    accuracy                           0.83        42
   macro avg       0.88      0.83      0.83        42
weighted avg       0.88      0.83      0.83        42
```

- Template Matching using OpenCV – accuracy of 55%

```
[[ 4 17]
 [ 2 19]]
              precision    recall  f1-score   support

           0       0.67      0.19      0.30        21
           1       0.53      0.90      0.67        21

    accuracy                           0.55        42
   macro avg       0.60      0.55      0.48        42
weighted avg       0.60      0.55      0.48        42
```

- Other methods that I did not implement but might prove successful: Regex, Pattern Matching

The output will be a curated list of page paths. Each path represents a page *relevant* and containing *tables*, which can then be fed into our final custom box OCR model to generate the digitized tables. Note that the methods involved in Step 2 and 3 need to be finetuned and fail some expected yes cases; it does better in returning a true negative than a true positive result.

Considerations

- **Multiple line headers**: Hard to capture headers exactly because everything below the first line is considered as observations in general.
  - Solution: Manually add headers.
- **Quality of pdfs**: The quality of some scanned pdf files are too low to read. It leads to many misreadings and makes Regex challenging.
  - Solution: Preprocessing and manually fixing
- **Accuracy metrics**: Hard to evaluate the performance of the model or the accuracy of the output because we don't have the perfect machine readable answer.
  - Solution: Manually create some perfect answer
- **Page filtering:** Besides manually selecting, we tried to find an effective way to only select pages that contain desired contents that did not need a lot of computational power.
  - Potential solutions: OCR, template matching, Regex, pattern matching

Final product

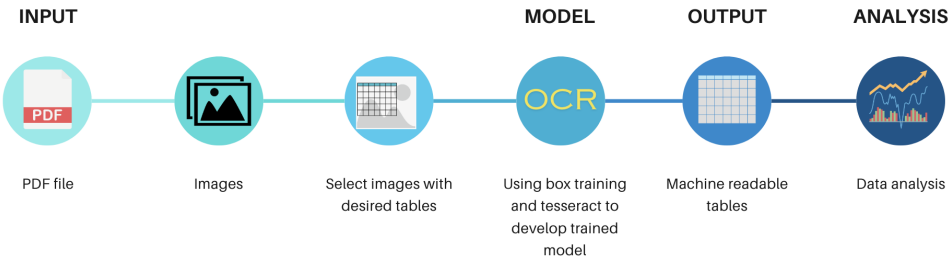
Deliverable Images
Original Images Folder

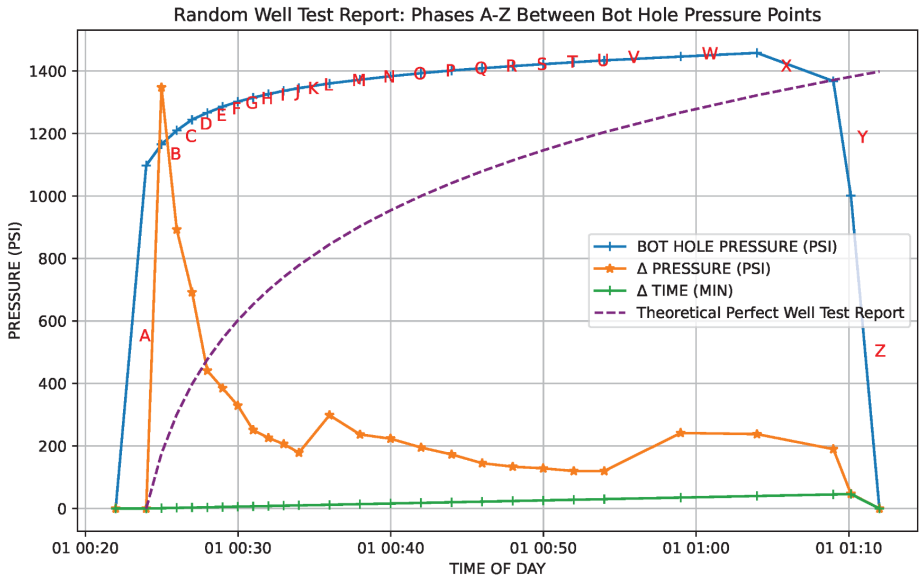
During_process.png
Box_editing_screenshot.png
Table_displays.pdf

Conclusion & Future Work

- Make the process of choosing which pages to extract tables from more seamless – can even change the model so it accepts a PDF
- Train the model on more pdfs, and continue to fine tune model based on discoveries from this
- Classifications of the output: how pressure changes with the well tests over time

Flowcharts


INPUT
MODEL
OUTPUT
ANALYSIS
PDF
OCR
PDF file
Images
Select images with desired tables
Using box training and tesseract to develop trained model
Machine readable tables
Data analysis


Random Well Test Report: Phases A-Z Between Bot Hole Pressure Points
PRESSURE (PSI)
TIME OF DAY
BOT HOLE PRESSURE (PSI)
Δ PRESSURE (PSI)
Δ TIME (MIN)
Theoretical Perfect Well Test Report
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z

OCR Language Training Process Using ML and Computer Vision