CSCI 370 Final Report

Corevintiv

Eadyn Thompson
Conor Hanna
Aiden Jenkins
Revised May 15, 2024

CSCI 370 Summer 2024

Prof. Kelly

Table 1: Revision history

Revision	Date	Comments
New	05-15-2024	Created initial requirements template and completed the following sections: I. Introduction II. Functional Requirements III. Non-functional Requirements IV. Risks V. Definition of Done VI. Team Profile References Appendix A - Key Terms
Added System Architecture	05-23-2024	
Added Software Quality	05-30-2024	
Finished initial draft	06-09-2024	IX. Project Completion Status X. Future Work XI. Lessons Learned
Finished revisions and finalized paper	06-16-2024	

Table of Contents

I. Introduction

This project aimed to collate and label image data of rock cores collected by 17 labs to extract useful geologic information such as mineral composition or the presence of natural fractures in the rock. The initial stage of this project involved reconstructing full rock cores from depth-labeled images of small subsections through the use of a mask-based recurrent convolutional neural network (MASK RCNN). After that, we used more in-depth annotations to extract useful features from the data.

This project was done in conjunction with Ovintiv, a petroleum company based out of Canada and the United States, and Dr. Zane Jobe, a research professor with the geology department at Colorado School of Mines. Data was collected and provided by Ovintiv through several subsidiary laboratories across the United States. This project was meant to simplify the prospecting of new oil wells, a process that currently relies heavily on extensive geologist assessment. This project further aids geologists in site planning with higher accuracy and less human labor involved.

This project follows previous research done by Dr. Jobe in the paper *Centimeter-Scale Lithology and Facies Prediction in Cored Wells Using Machine Learning*. This project uses existing open-source image recognition technologies like MASK RCNN, released by Matterport Inc. Some updates to this software were needed in the course of this project, as this library is no longer supported.

Rock core images are photos of cross-sections of rock extracted from the Earth at varying depths, which provide information about things like mineral composition and density. A mask recurrent convolutional neural network is a type of neural network in which bounding boxes are created around pre-specified objects in images using recurrency applied to a convolutional neural network (CNN). A CNN is a network that identifies features in data by aggregating subsets of the data using matrix convolutions. A recurrent network is a network in which previous inputs can affect the evaluated result for subsequent inputs.

II. Functional Requirements

Our project has the following functional requirements:

- Have the ability to “stitch” photos of the same core together to reconstruct the full core spire
- Have the ability to identify features of the core samples (natural cracks, mineral components, etc.) utilizing machine learning
- Differentiate between man-made and natural fractures in the core sample

III. Non-Functional Requirements

In addition to our functional requirements, our project also has the following non-functional requirements:

- Utilize new (<2 years old) machine-learning techniques
- Build upon existing research (in particular, the CoreBreakout paper)
- Have a legible and concise report encapsulating the project and its scope for all well-sites

IV. Risks

Our project could have been delayed or made more difficult as a result of the following:

- Software Risks
    - The code base given to us is lacking in some regard.
        - Likelihood: Likely
        - Impact: Minor
        - Mitigation: To counteract this event, we needed to develop our own code
    - The model we develop requires more images to train with than would be reasonable for the size of our dataset
        - Likelihood: Unlikely

        - Impact: Major
        - Mitigation: To avoid this, we needed to carefully design the architecture of our model to perform better with a smaller training dataset with a cost in overall performance
- Experience Risks
    - The members of our team have varying levels of experience with machine learning.
        - Likelihood: Very Likely
        - Impact: Moderate
        - Mitigation: Those with less experience can learn on their own or be guided by the other members of the group

V. Definition of Done

The project was completed when we had a machine learning model that could accurately take in images of separated rock cores and output a single image. The model needed to identify the segments of the core in the original image and stack them according to the depth they were originally at. The project also required the identification of any cracks present. The output images were a recreation of the full length of the cores, alongside data tables indicating where cracks were present.

The product will be delivered through a hard drive to be handed to the client at a scheduled lunch meeting.

VI. System Architecture

Our assignment started with unprocessed data from 17 locations, with each location having between 1 and 25 cores, with each core consisting of many images of rock in sequence. This data initially came in many formats and had to be standardized for use in our project. This input data, used for training the Mask-RCNN model, was annotated using the open-source Python package Labelme. Lableme was used to manually create ground truth masks for each image, polygons highlighting parts of the image the model is meant to extract (Fig 1). These masks were used in the training, validation, and testing of the model. Labelme exports the annotations in a JSON format, with each vertex of the label polygons saved and used to create the mask for the corresponding image. The JSON and original image files were saved in a directory named after the company that took the image to simplify file management.

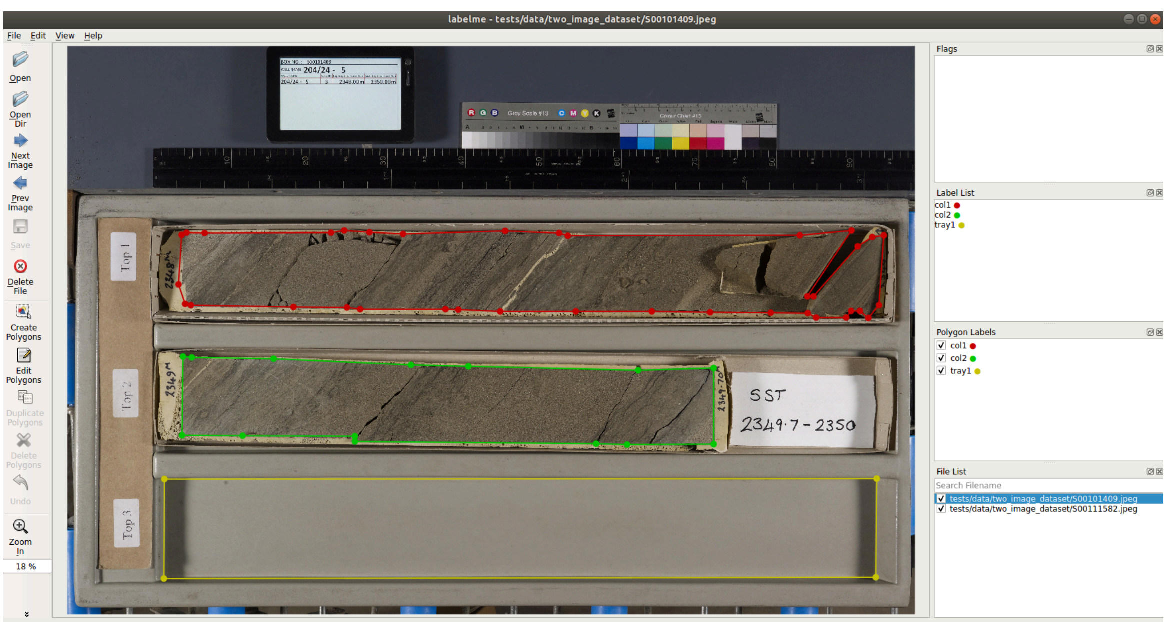

Fig 1: Example Annotations using Labelme Software

The architecture of the Mask-RCNN model used in this project was designed to perform object detection and segmentation. It is responsible for differentiating between the tray and the rock core samples. It combined a ResNet backbone for feature extraction with a region proposal network (RPN) and additional layers for mask prediction.

Model evaluation was done by comparing ground truth (GT) masks to masks predicted by the Mask-RCNN. The loss function attempted to balance accurately, including parts of the image that are part of the mask, while not including parts of the image that are not part of the mask. This was done by defining an overall loss term, as the per-pixel sum of the loss terms for $L = L_{cls} + L_{box} + L_{mask}$, with L_{cls} as a function of the true class of the pixel, used to encourage correctly assigning a mask to a class, L_{box} as a function of the distance between the predicted and ground truth masks, and L_{mask}, which evaluates how accurately the class that each pixel belongs to is predicted by the model. These individual loss functions were optimized to robustly ensure that our model could accurately segment images as needed. It was especially important to ensure that the model consistently identified all parts of the rock sample to avoid losing information. This was impossible to avoid to some extent, but minimizing this was a major part of our core goals.

To date, many versions of the Mask-RCNN model have been trained and tested with the aim of applying the best possible model to our unseen image data. There were not any major issues in this process.

Additionally, as a stretch goal, Image light levels were explored using OpenCV to isolate color samples, alongside tools like Matplotlib for visualization of data collected.

VIII. Technical Design

The below image showcases the flow of data through our model, from the starting state data was received in, to the use of the model for constructing core images. This process was completed for the more than 3000 images received in this project.

The initial stage, in which libraries are used to standardize image formats, was primarily done in Python. Tools like PIL and Sci-Kit Image, popular Python libraries for image manipulation, were used to process, read, and write images. This is in addition to tools like FFMPEG, an open-source image and video manipulation tool. Pyexcel was used to extract images from Excel sheets when needed.

Images were then annotated using Labelme, a popular open-source Python library that simplified the process of creating masks in an easily usable format. An example of a mask is shown in blue on the diagram.

From there, we modified code from the previous research project to load our data before using it to train a MASK-RCNN model. Finally, the model was used to reconstruct the core images from all the image data received. An example of the model reconstructing a core is shown at the bottom of the diagram.

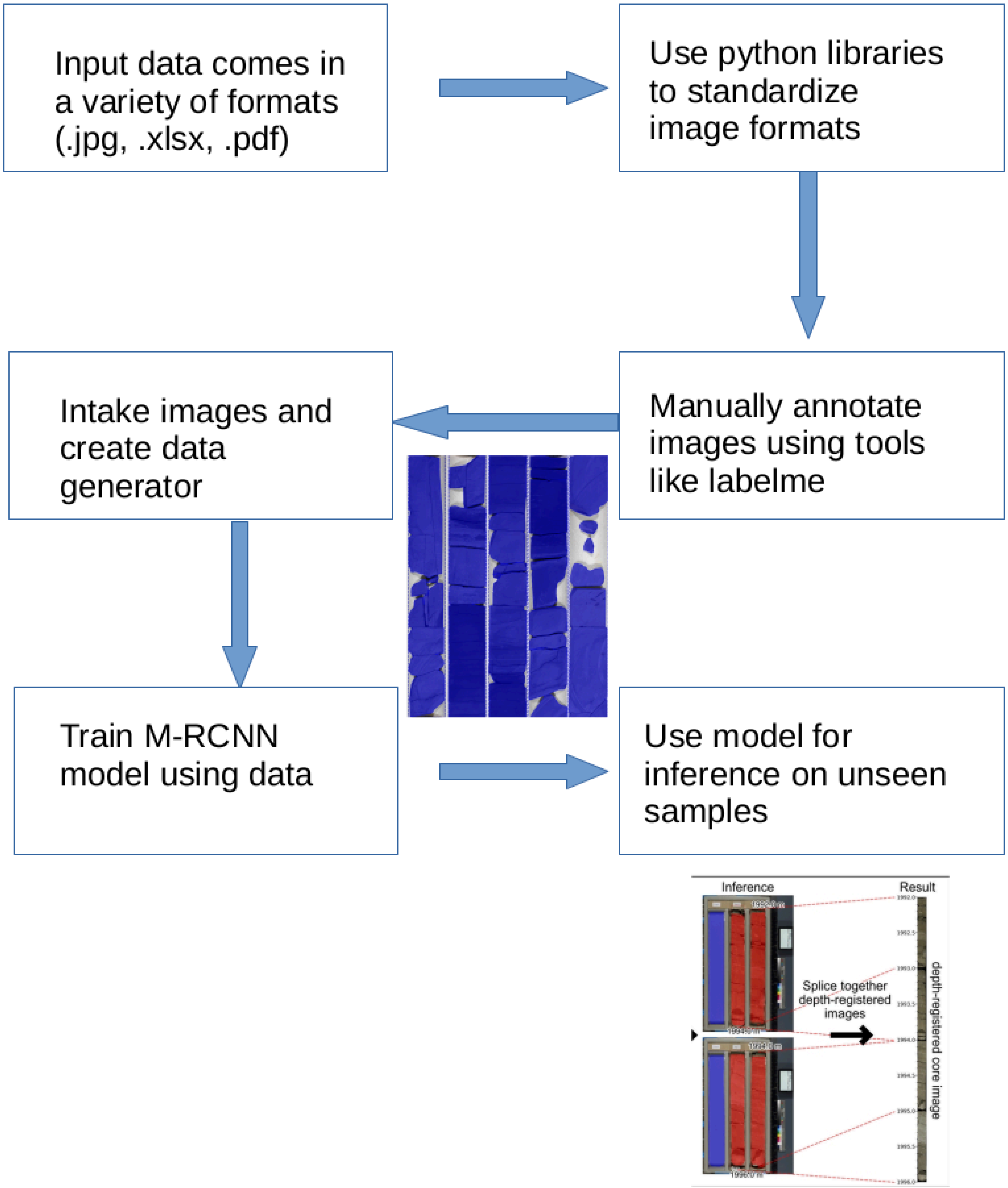

Fig 2: Flow Diagram for Model Processing of Data

The Mask-RCNN model, which is based on a ResNet model architecture, enables this data flow process. ResNet stands for residual network, a type of network that incorporates a residual 'skip' connection. This means that some of the layers making up the model receive the inputs passed to previous layers and the outputs of those previous layers. This significantly speeds up training and improves accuracy.

The below image also shows the breakdown of each stage of the model. Each stage includes convolutional layers, which learn progressively higher-level features in the input data, batch normalization, which acts to ensure inputs received have similar distributions, rectified linear unit (ReLu) layers, which introduce non-linearity by acting differently depending on if the input is positive or negative, and a pooling layer which aggregates previous outputs.

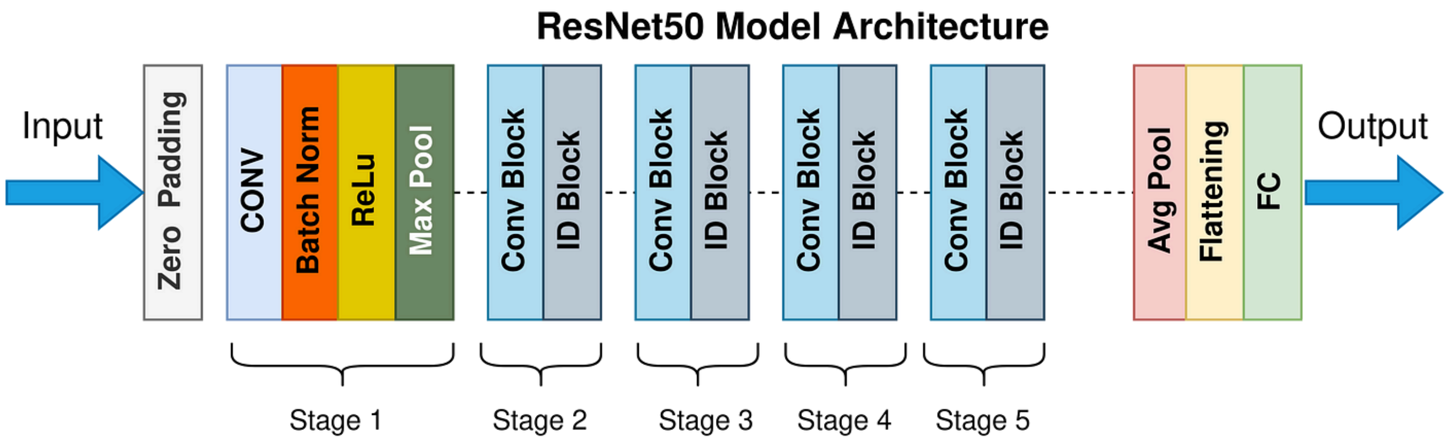

Fig 3: ResNet Model Architecture

This model architecture allows our Mask-RCNN model to effectively learn to segment images as needed for our project.

VIII. Software Testing and Quality

As our project involved machine learning for bounding box detection, relatively few objective measures could be used as assessments. We had access to test metrics, which allowed us to do things like compare the predicted bounding box for part of an image to the human-assessed bounding box for that image. We carried out this comparison at the end of each training epoch, using 15 manually assessed images representing a large spectrum of possible images. This included images that span from large amounts of highly visible fracturing to images with nearly none and images with various backgrounds and layouts. This meant that model assessment on these test images should accurately predict average model performance on entirely unseen samples.

In addition to the model validation code, we have also created a model evaluation rubric, which should make it possible to compare model performance on several subjective measures. This rubric was used while manually looking at the performance of model predictions on unseen hold-out images.

Model Evaluation Rubric					
Model	Does the model include the correct parts of the image?	Does the model avoid including incorrect parts of the image?	Does the model propose all relevant columns?	Does the model correctly classify columns?	Total
“TrainingNumber-ChangeDescription”	1-5	1-5	1-5	1-5	4-20
Example: “4-lowConfThreshold”	5	2	5	4	16

In addition to the testing needed to ensure that the Mask-RCNN model is effective and reliable, testing must also be done on the depth extraction code used to collate core images. This testing came in two phases - one round of checks while extracting depths using Optical Character Recognition (OCR) and another when validating already extracted depths. For the first, human checks were used when the OCR code failed to extract a depth range with high confidence. If a clear depth range is not found, an image of the OCR bounding box used is shown to the user, who can note down the correct range for later use (Fig 4). After generating guesses of depth ranges for each image in a dataset, a second script is used to check that the depth labels fully span the range dictated by the overall depth across the entire core.

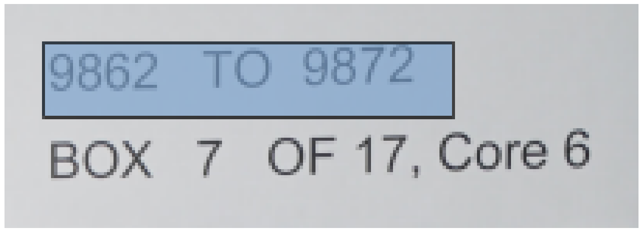

Fig 4: Example of Text-OCR Box

To test our program, our client will manually check the results of our model (Core Outlines, Depths, and Cracks) and the model's performance metrics. We can assess if the validation model's bounding box loss is lower than 0.8 (The predicted bounding box is 80% similar to the correct/man-made bounding box) and if the model class' predictions are lower than 0.025 (The ordering of segments in the same image is wrong only 2.5% of the time). To test the crack detection algorithm, our client will manually check the images, as no performance metric can be created for the algorithm since there is no "correct" example to compare to, and the current student team lacks the geological knowledge necessary to adequately perform this check.

Human inspection of pre-transform and post-transform data verified image light level data.

VIII. Project Ethical Considerations

Our program does not physically interact with humans and is not the final step in any decision process, as a result, the ethical considerations for the product were centered around its quality, communication with the client, and proper data handling. Most importantly, we attempted to adhere to ACM principles 1.03, 3.01, 3.02, 3.10, 3.14, and 5.0 while paying special attention to 2.05 and 5.05. In summary, these principles cover realistic deadlines, keeping confidential information private and secure, proper goal setting, adequate testing, client policy adherence, data integrity maintenance, ensuring quality software, and wishing to do good with said software. Outside of ACM principles, we also ensured the passing of some key Michael Davis tests: we believe our product does less harm than any alternative, adheres to the laws and policies of both our client and the United States Government, and would be more than proud to look at ourselves in the mirror following completion.

IX. Project Completion Status

This project aimed to create a program to reliably compile diverse images of rock core samples into full core images, which has been completed (Fig 5). All central project goals have been met, and the data has been prepared for delivery to the client. This has involved the creation of almost 100 full images, with the longest of these representing more than 1500 feet of rock. This data is accurately depth-labeled and has been verified by looking at its relationship to other measurements. Our final model significantly exceeded the minimums established for validation metrics in part VIII and was rated highly on our model rubric.

Fig 5: One of our completed images, with lines split up to save space

In addition to project requirements, some stretch goals were completed as well. One goal was to identify light levels in images, with the aim of adjusting for them when doing further analysis. We were able to accomplish this, with the below image (Fig 6) showcasing this. On the diagonal are the two images with the lowest and highest brightness, while the off-diagonal shows what the images would look like if they were normalized to have the same light level as their opposite.

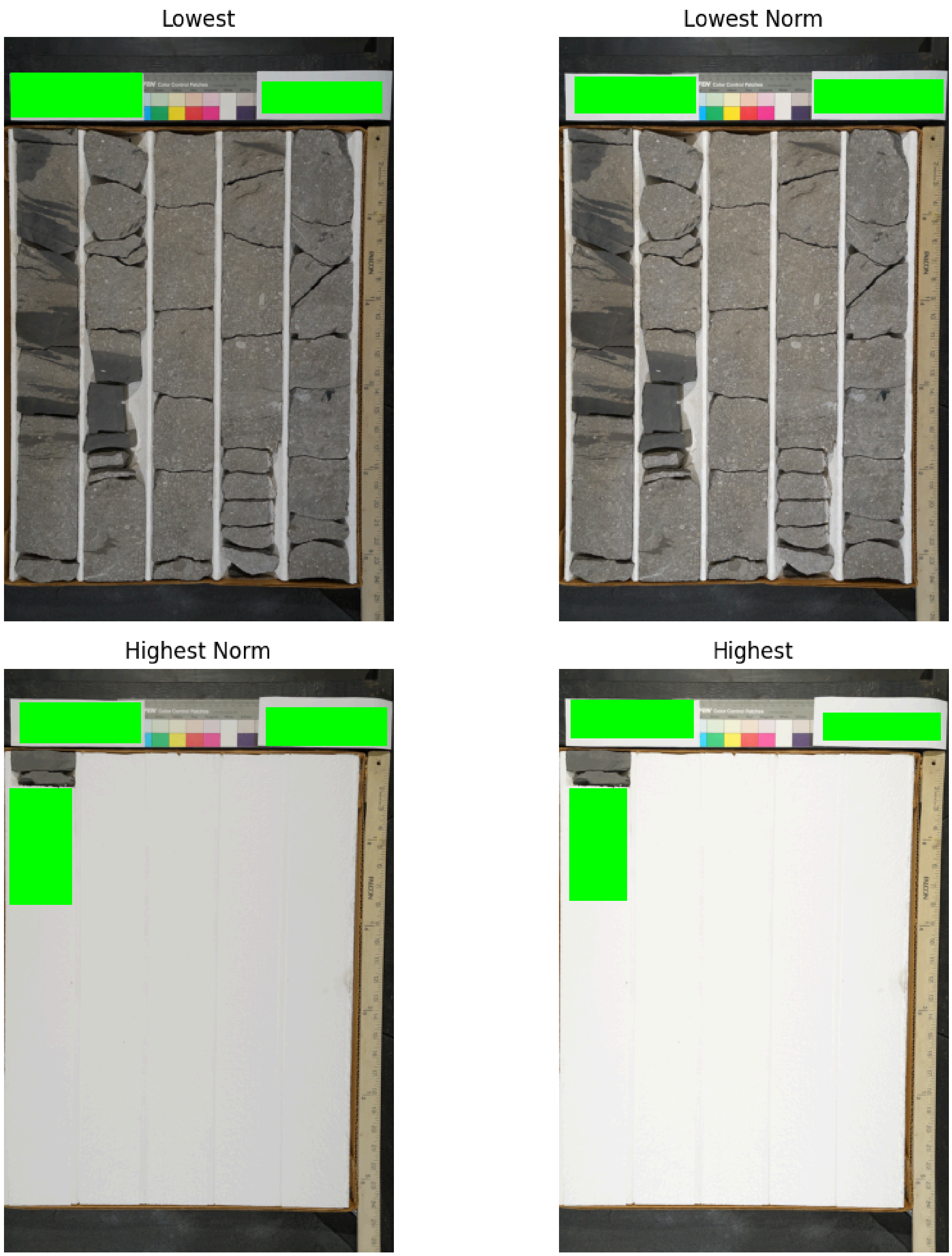

Fig 6: Light Level Comparisons with Location Data Emitted.

Some stretch goals have not been completed, and our recommendations have been compiled in the 'Future Work' section.

X. Future Work

As our team was unable to reach the stretch goal of identifying the core segments' mineral composition, any future work should include adding this last stretch goal to the product's functionality. Beyond this and any more functionality the client will want down the line, future work on the product should include enhancing the model's accuracy. For example, the product currently has a desired bounding box loss of at least 0.8. Future work should lower this threshold so the product has fewer errors in the images it creates. Our recommendation to increase accuracy is to continue with the current model architecture but to use a greater number of annotations. The hyperparameters chosen for the model are also likely imperfect and might be improved through a grid search or similar approach.

There's also more to be done in exploring the effects of lighting conditions on images, which the client introduced as a potential stretch goal later in the project. We've been able to complete preliminary work on this, but a more thorough evaluation would be ideal. If we had more time, it would also be very exciting to see the results if we had adjusted light levels before model training and inference.

Our current approach is also unable to differentiate between natural and artificial breakage in the rock, which was a stretch goal that we hoped to complete. We are still determining if this can be accomplished using a clustering algorithm or if a more complex supervised model is needed. We ran out of time while investigating whether the clustering algorithm, known as DBSCAN, could be used to implement this functionality. We believe it can and recommend looking into DBSCAN more to achieve this functionality. DBSCAN's ability to identify anomalies, like fractures, should allow this approach to work.

XI. Lessons Learned

1. Trying to prevent the loss of crucial information while also needing to reduce the dimensionality or simplify the data is very difficult and sometimes impossible.
2. Going back and refining early work is often critical. The first images segmented went much worse than the final ones, and if we didn't redo those initial samples, we would be delivering a significantly sub-par product.
3. Sanity check every stage possible when doing complex programming. We ran into an issue where image masks were not loading right for a subsection of images, and we spent a long time exploring potential failures in high-level code before discovering that the issue occurred when some image flags were ignored at the beginning of the data-loading process.
4. When looking at an established code base, the first step should always be to try to understand and catalog all parts of the code. We struggled to implement existing or partially supported features at multiple points in this process. For example, we tried using data augmentation to improve model accuracy using custom code before later discovering this feature was already present.
5. Whenever feasible, use existing code and libraries. When trying to identify image light levels, for instance, we used OpenCV to convert the image from RGB to HSV, isolate the color control patches, and apply the transformation used to change light levels.

XII. Acknowledgments

We would like to thank our client, Ovintiv, who has graciously provided us with an opportunity to work on a complex and challenging real-world project that otherwise would have been impossible. We also deeply appreciate the help and support provided by Chris Christofferson, who provided the data that made this project possible and who was always available for help and guidance when needed. We are also grateful for the help and assistance given to us by Dr. Zane Jobe of the Colorado School of Mines Geology department, who continued to provide help and support even as he was physically half the world away. Lastly, we would like to thank our advisor, Professor Kathleen Kelly, for her consistent and

dedicated support and for the supportive learning environment she provided. Each of these people helped us go so much further than we could have otherwise, and we are so grateful for all they've done for us.

XIII. Team Profile

Aiden Jenkins

Sophomore

Computer Scientist

Hometown: Austin, Texas

Work Experience: Jr Data Analyst, Lifeguard, Cashier

Interests: Archery, Basketball, Video Games

Conor Hanna

Sophomore

Computer Scientist

Hometown: Houston, Texas

Work Experience: Host, Lifeguard, Assistant

Interests: Video Games, Foreign Policy/International Affairs

Eadyn Thompson

Sophomore

Computer Science (Data Science Specialty)

Hometown: Philadelphia, Pennsylvania

Work Experience: Health Data Analytics, CRM Software

Interests: Sci-Fi Novels, Hiking

References

K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," arXiv.org, Mar. 20, 2017. https://arxiv.org/abs/1703.06870

K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv.org*, Dec. 10, 2015. https://arxiv.org/abs/1512.03385

Martin T, Meyer R and Jobe Z (2021) Centimeter-Scale Lithology and Facies Prediction in Cored Wells Using Machine Learning. *Front. Earth Sci.* 9:659611. doi: 10.3389/feart.2021.659611

Oriol Falivene, Neal C. Auchter, Rafael Pires de Lima, Luuk Kleipool, John G. Solum, Pedram Zarian, Rachel W. Clark, Irene Espejo; Lithofacies identification in cores using deep learning segmentation and the role of geoscientists: Turbidite deposits (Gulf of Mexico and North Sea). *AAPG Bulletin* 2022;; 106 (7): 1357–1372. doi: https://doi.org/10.1306/03112221015

Appendix A – Key Terms

Include descriptions of technical terms, abbreviations and acronyms

Term	Definition

MASK RCNN	*Mask recurrent convolutional neural network. A type of neural network in which bounding boxes are created around pre-specified objects in images. A convolutional network is one that identifies features in data by aggregating subsets of the data. A recurrent network is a network in which previous inputs can affect the evaluated result for following inputs.*
Rock Core	*Rock core images are photos of cross-sections of rock extracted from the Earth at high depths, which provide information about things like mineral composition and density.*
ResNet	*Short for residual network. A type of model architecture common to image recognition in which a skip (residual) connection is used to compute the results of additional functions of the data which are passed directly into later layers.*
DBSCAN	*A clustering algorithm that determines if a vertex is part of a cluster by looking at if it is close enough to other vertices that are already a part of a cluster.*
OCR	*Optical Character Recognition*

Appendix B – Relevant Ethical Principles

1.03. Approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy or harm the environment. The ultimate effect of the work should be to the public good.

2.05. Keep private any confidential information gained in their professional work, where such confidentiality is consistent with the public interest and consistent with the law.

3.01. Strive for high quality, acceptable cost and a reasonable schedule, ensuring significant tradeoffs are clear to and accepted by the employer and the client, and are available for consideration by the user and the public.

3.02. Ensure proper and achievable goals and objectives for any project on which they work or propose.

3.10. Ensure adequate testing, debugging, and review of software and related documents on which they work.

3.14. Maintain the integrity of data, being sensitive to outdated or flawed occurrences.

5.03. Ensure that software engineers know the employer's policies and procedures for protecting passwords, files and information that is confidential to the employer or confidential to others.

5.05. Ensure realistic quantitative estimates of cost, scheduling, personnel, quality and outcomes on any project on which they work or propose to work, and provide an uncertainty assessment of these estimates.