

# Building a Geological Cyber-infrastructure: Automatically detecting Clasts in Photomicrographs

Ari'El Encarnacion<sup>1</sup>, Ben Katin<sup>1</sup>, Matty Mookerjee<sup>1</sup>, and Gurman Gill<sup>2</sup>

<sup>1</sup>Sonoma State University

<sup>2</sup>Sonoma State University

November 24, 2022

## Abstract

To incentivize the participation and contribution to the growth of an earth-science-based cyberinfrastructure, analytical environments need to be developed that allow automatic analysis and classification of data from connected data repositories. The purpose of this study is to investigate a machine learning technique for automatically detecting shear-sense-indicating clasts (i.e., sigma or delta clasts and mica fish) in photomicrographs, and finding their shear sense (i.e., sinistral (CCW) or dextral (CW) shearing). Previous work employed transfer learning, a technique in which a pre-trained Convolutional Neural Network (CNN) was repurposed, and artificially augmented image datasets to distinguish between CCW and CW shearing. Preprocessing images by denoising, a process in which noise at different scales is removed while preserving edges of an image, improved classification accuracy. However, upon randomizing the denoising parameters, the CNN model didn't converge due to severe lack of data. While the efforts for acquiring more labeled data is ongoing, this work compensated for it by implementing a pre-processing "detection" system that automatically crops images to regions of image containing the clasts. This is done by utilizing YOLOv3, a CNN based image detection system that outputs a bounding box around an object of interest. YOLOv3 was trained using 93 photomicrographs containing bounding boxes of 344 shear-sense-indicating clasts. The retrained detector was tested on two sets: set A with 10 photomicrographs containing clasts and set B with 100 photomicrographs not containing clasts. All but one of the clasts in set A were correctly detected with an average confidence score of 96.6%. On set B, 72% of images correctly did not indicate presence of clasts. On the remaining images, where clasts were incorrectly identified, an average confidence score of 78.3% was observed. By utilizing a threshold on the confidence scores, the system could be made more accurate. Future work involves utilizing the bounding boxes output by the detection system to refine and improve the CNN model for classifying shear sense of clasts in photomicrographs.

# Building a Geological Cyber-infrastructure: Automatically Detecting Clasts in Photomicrographs



**Building a Geological Cyber-infrastructure: Automatically Detecting Clasts in Photomicrographs**

Ari'El Encarnacion\*, Ben Katin\*, Dr. Gurman Gill\*, Dr. Matty Mookerjee\*\*

\* Department of Computer Science, \*\* Department of Geology  
Sonoma State University

**Overview**

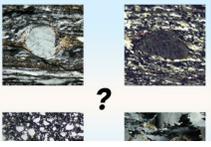
**The Ambition**

To incentivize the participation and contribution to the growth of an earth science cyberinfrastructure, analytical environments need to be developed that allow automatic analysis and classification of data from connected data repositories.

As a test case for this endeavour, we chose to develop a system that will automatically detect and classify shear-sense photomicrographs of sigma clasts into two categories, clockwise (CW) and counter-clockwise (CCW).

OPEN

**Dataset**



OPEN

**Methods: YOLOv3**

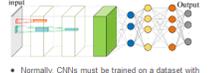


- YOLOv3 (a recent version of YOLO), created by [1], is a CNN based image detection system that outputs bounding boxes around objects of interest
- YOLOv3 specifically used in our data optimization

OPEN

**Methods: CW/CCW Classification**

**Transfer Learning**

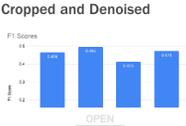


- Normally, CNNs must be trained on a dataset with millions of images to learn the intricate details in

OPEN

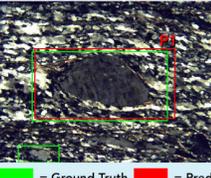
**Results: CW/CCW classification'**

**Cropped and Denoised**



OPEN

**Evaluating YOLOv3 Predictions on 'With' Images: Overlap & Confidence**

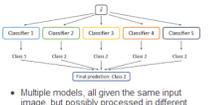


Denoise: alpha 100, Resize: None  
P1: O = 0.82, C = 100%

■ = Ground Truth ■ = Predicted

**Future Work**

**Ensemble Classification**



- Multiple models, all given the same input image, but possibly processed in different

OPEN

Ari'El Encarnacion\*, Ben Katin\*, Dr. Gurman Gill\*, Dr. Matty Mookerjee\*\*

\* Department of Computer Science, \*\* Department of Geology  
Sonoma State University

PRESENTED AT:



**2020 EarthCube Annual Meeting  
Virtual – June 18, 2020**

# OVERVIEW

## The Ambition

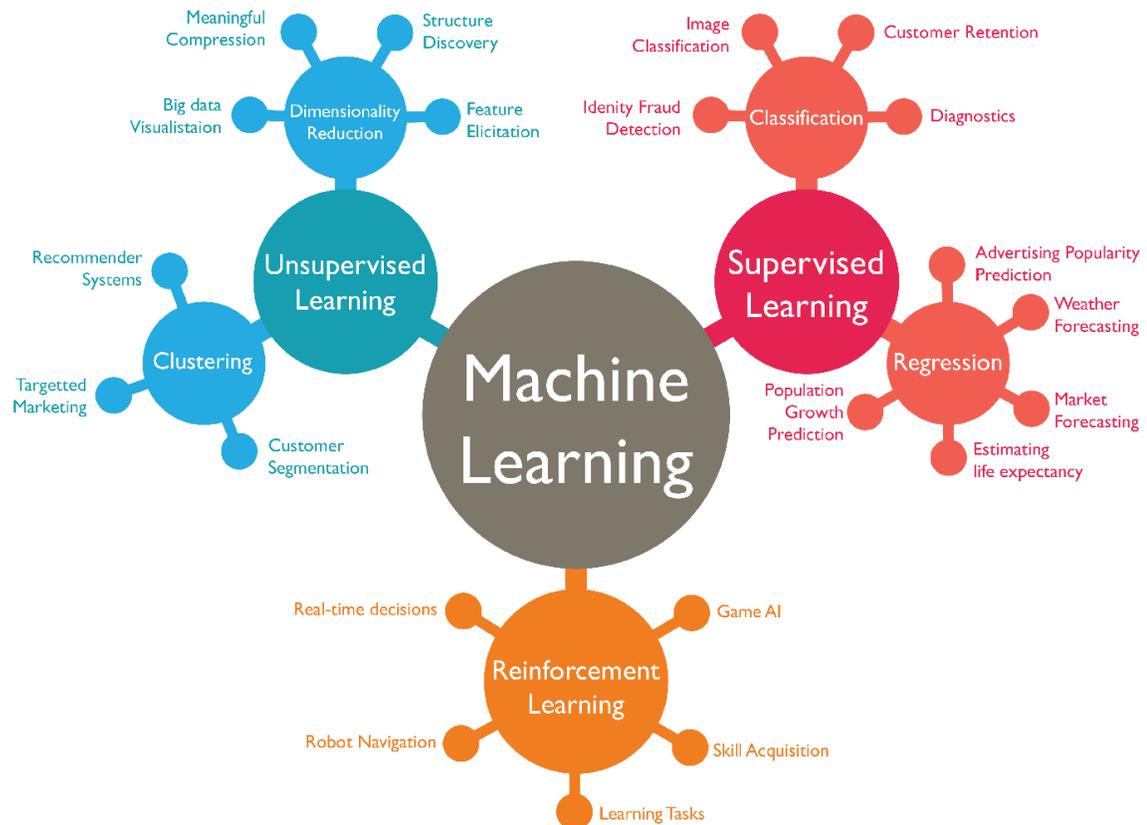
To incentivize the participation and contribution to the growth of an earth science cyberinfrastructure, analytical environments need to be developed that allow automatic analysis and classification of data from connected data repositories.

As a test case for this endeavour, we chose to develop a system that will automatically detect and classify shear-sense photomicrographs of sigma clasts into two categories, clockwise (CW) and counter-clockwise (CCW).

## Poster Overview

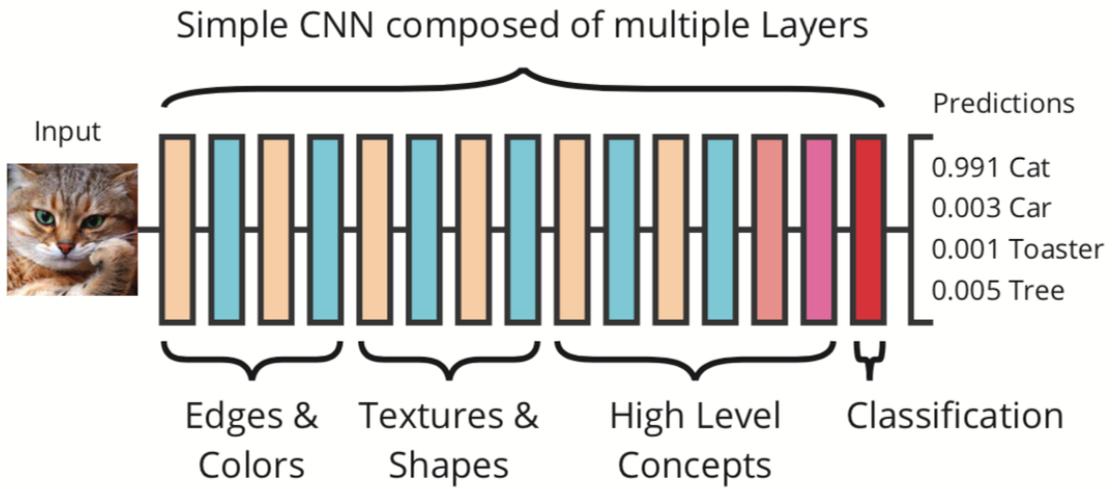
- Machine Learning, specifically Convolutional Neural Networks as an image classification tool
- Transfer Learning and Image Augmentations to compensate for an especially small dataset
- Preprocessing and auto-cropping as data optimization pipelines
- Future work & References

## Machine Learning



- Machine Learning (ML) is a powerful tool that automatically finds patterns in a given dataset
- A trained ML model can then be used to perform automatic analysis on never before seen data

## Convolutional Neural Network (CNN)

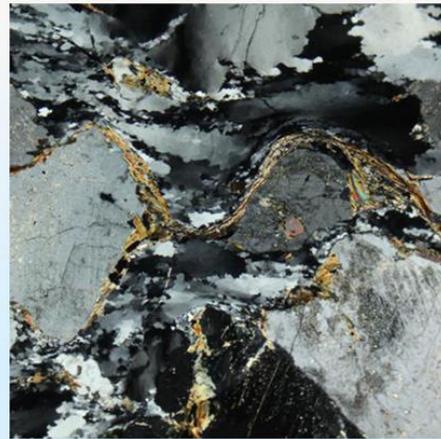
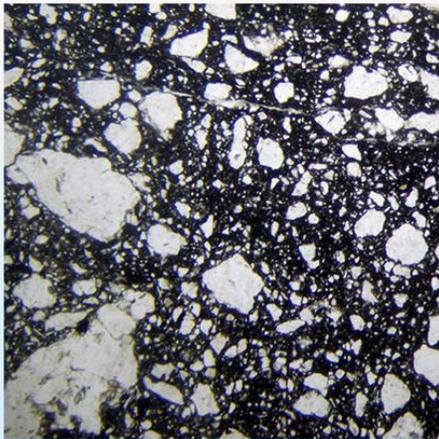
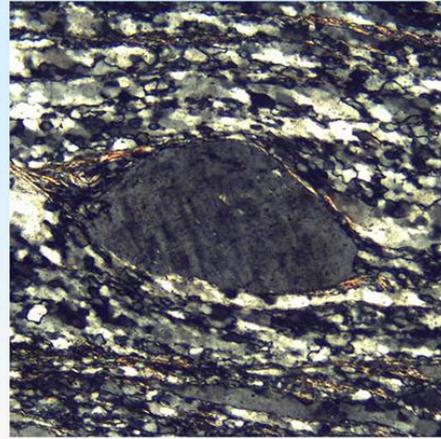
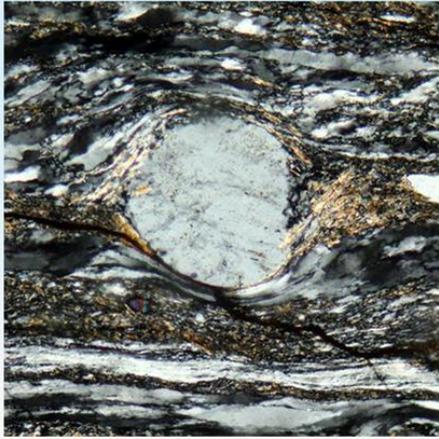


- CNNs are a type of ML network
- Modeled after the human brain
- Often used for image processing
- Two main steps: training and testing.
  - Training is a “setup” phase in which the CNN automatically learns how to recognize significant features of an image
    - Model learns primitive shapes, succeeded by curves and edges, ultimately reaching the ability to recognize complex shapes
  - Testing is the phase in which new data (data that was not used during training) is input into the CNN
    - A prediction is output with an associated confidence level for each trained class

## Technologies Used

- Models: Regression, Classification
- Libraries: TensorFlow, Keras, PyTorch
- Systems: Inception, ResNet, YOLO
- Techniques: Data Preprocessing, Data Augmentation

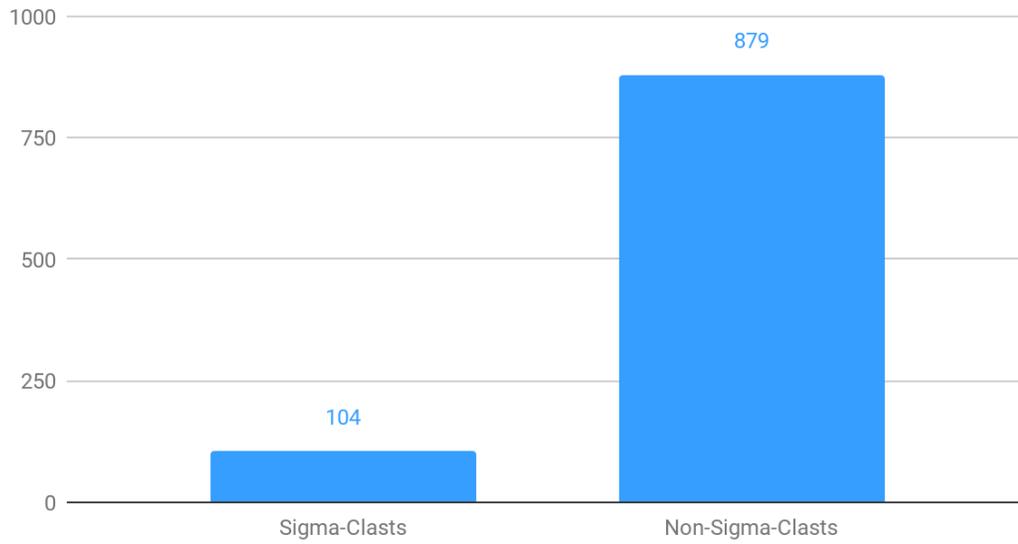
# DATASET



- Our dataset is quite small regarding common image classification standards. Typically CNN classifiers are trained on millions of images.
- 103 "With" images (those containing a clast)
  - 33 CCW clasts
  - 70 CW clasts
- 879 "Without" images (those containing no clast)

## Difference in Class Size

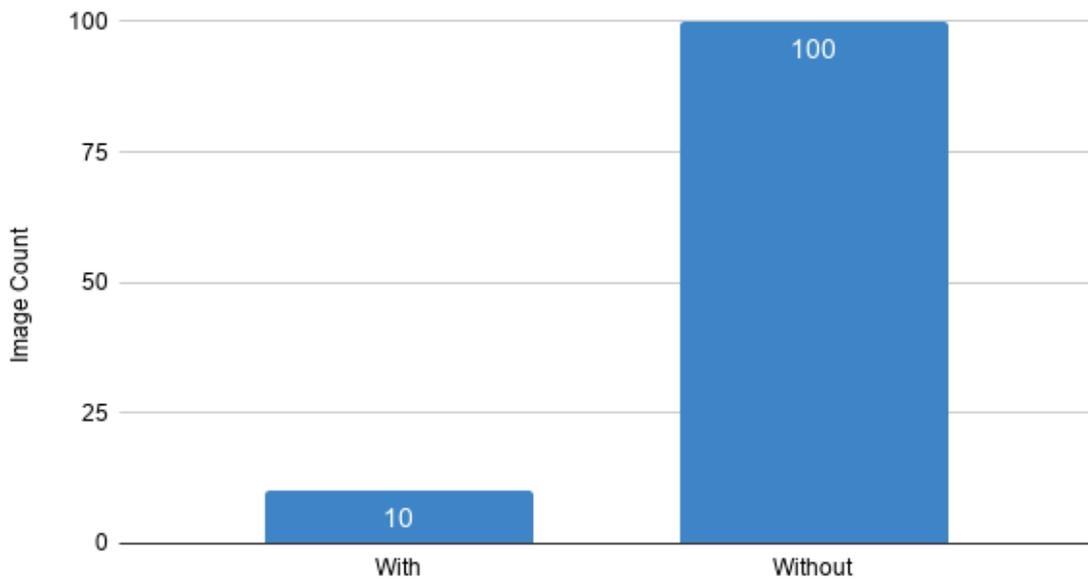
## Data Distribution between Sigma and Non-Sigma Categories



- Large gap between each respective class causes issues when training our model

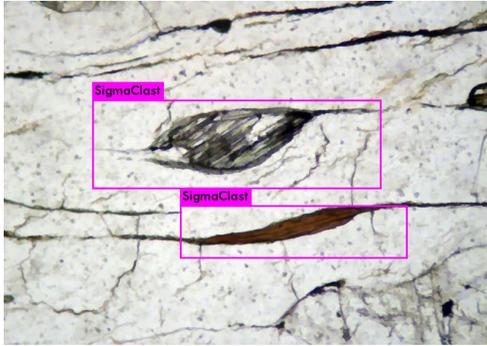
## YOLOv3 on Our Data

### YOLOv3 Test Set Distribution: With vs Without



- YOLOv3 trained on 93 photomicrographs
  - Contains 344 labeled shear-sense-indicating clasts
- Test set includes 110 photomicrographs
  - 10 photomicrographs with shear-sense-indicating clasts ('With' images)
  - 100 photomicrographs containing no clast ('Without' images)

## METHODS: YOLOV3

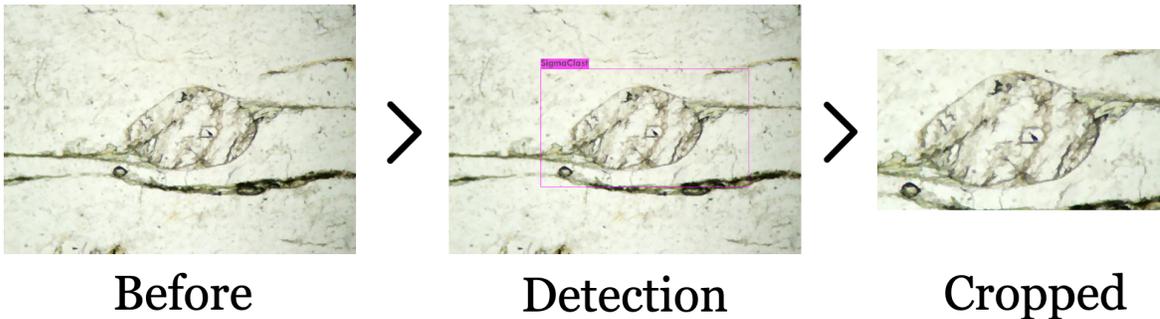


- YOLOv3 (a recent version of YOLO), created by [1], is a CNN based image detection system that outputs bounding boxes around objects of interest
- YOLOv3 specifically used in our data optimization pipeline

### Auto-Cropping

## Data Optimization

Using image detection as a method for auto-cropping



- Automatically cropping an image reduces data clutter
- Cropped to a region smaller than the dimensions of the original photo
- Given buffer large enough to encapsulate all the significant details of the region of interest
- Implemented with YOLOv3
- Utilize bounding box output by YOLOv3

### How Does YOLOv3 Work?

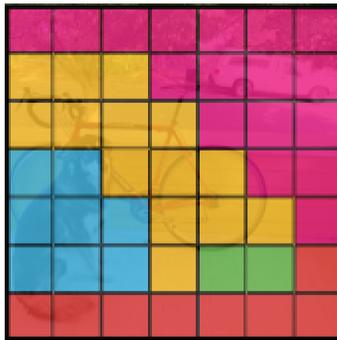
Image classifiers typically operate by scanning a small portion of an image multiple times. On each iteration the detector guesses what's in the current window and gives a confidence level for that guess. Once the entire image has been scanned, the detector selects its most confident guesses as its overall output. [2]



YOLO only scans an image once (hence its name: You Only Look Once). On running detection, YOLO first applies a single neural network to the input image, dividing it into multiple regions. Each region then receives a bounding box prediction. Each bounding box then outputs a confidence score that evaluates the shape of the box, out of context from whatever the box may contain. [2]

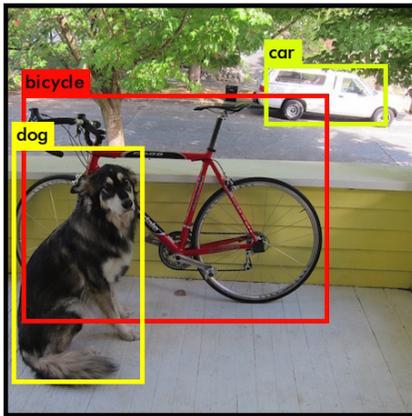


Next, each box is given an associated class prediction, similar to a traditional classifier. This creates a map, returning a highest probable class to each cell.



Class probability map

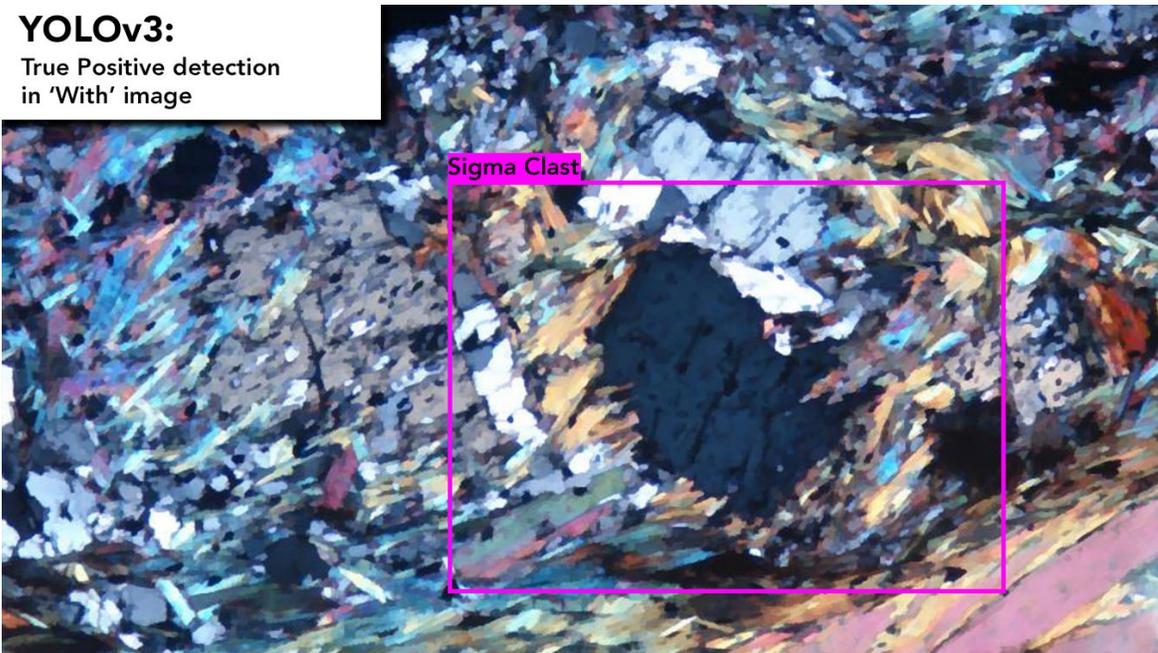
Finally, a bounding box is selected based off this map and a threshold is applied to ignore any box below a certain confidence score. [2]



## Detection Examples

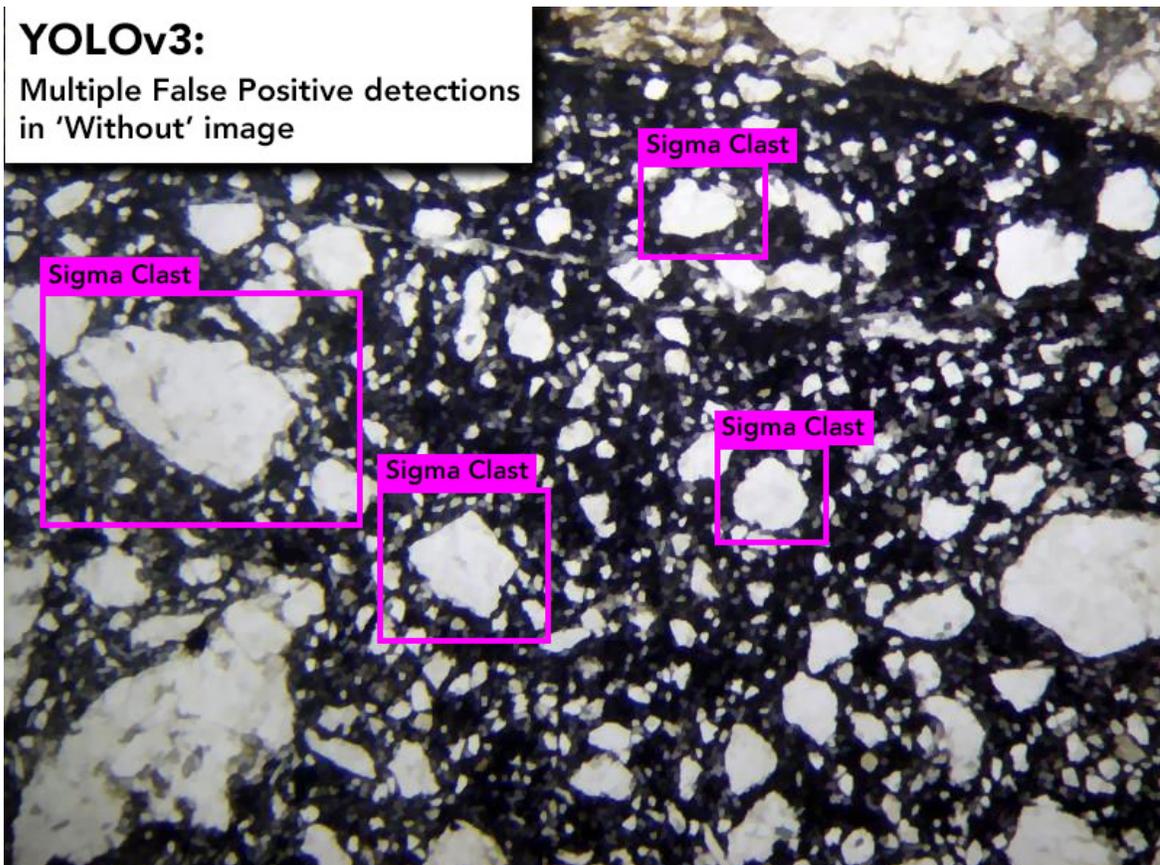
### YOLOv3:

True Positive detection  
in 'With' image



## YOLOv3:

Multiple False Positive detections in 'Without' image



## Evaluation

### Confidence Score (Conf.)

- A metric output by YOLOv3 for each predicted bounding box
- Defines how confident YOLOv3 is on its predictions

### Prediction-Truth Overlap (O)

Let  $O$  be the result of the Prediction-Truth Overlap,  $P \cap G$  the area of intersection between the predicted bounding box and the ground truth bounding box, and  $P \cup G$  the area of union between the predicted and ground truth bounding boxes.

Then,

$$O = \frac{P \cap G}{P \cup G}$$

- Used for evaluating accuracy of YOLOv3 bounding box

### Recall (R) & Precision (P)

Let  $T$  = True Positive detections in 'With' images

Let  $A$  = All detections in 'With' images

Let  $F$  = False Positive detections in 'With' and 'Without' images

Then,  $recall = \frac{T}{A}$  and  $precision = \frac{T}{T+F}$

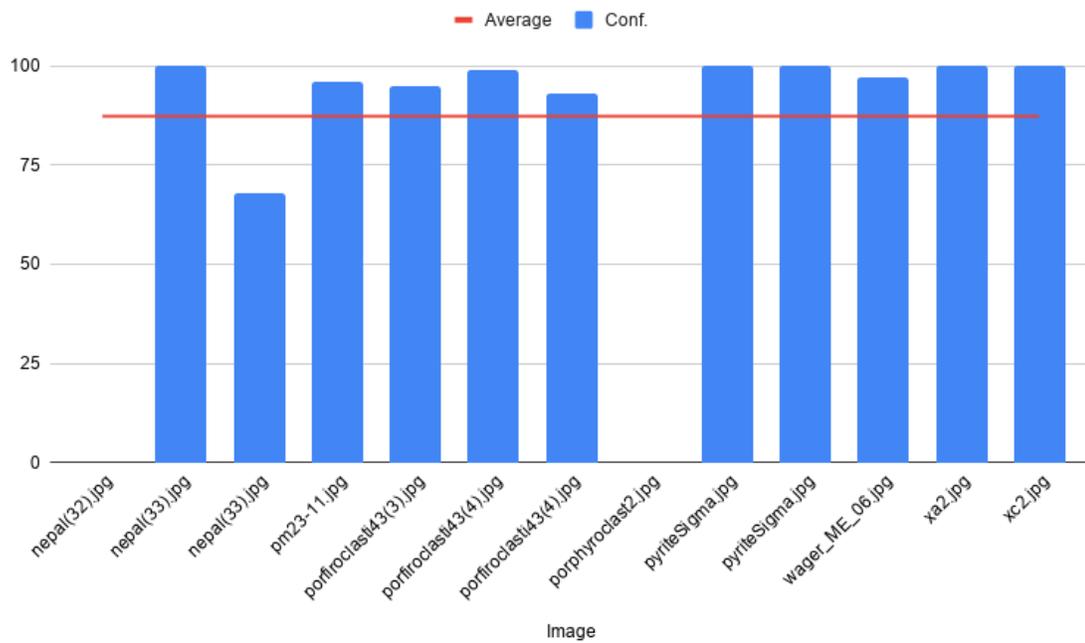
- R is the total amount of True Positive detections output by YOLOv3
- P is the ratio between T and F detections

## YOLOv3 Results

Detections on 'With' Images (alpha 100, full size)

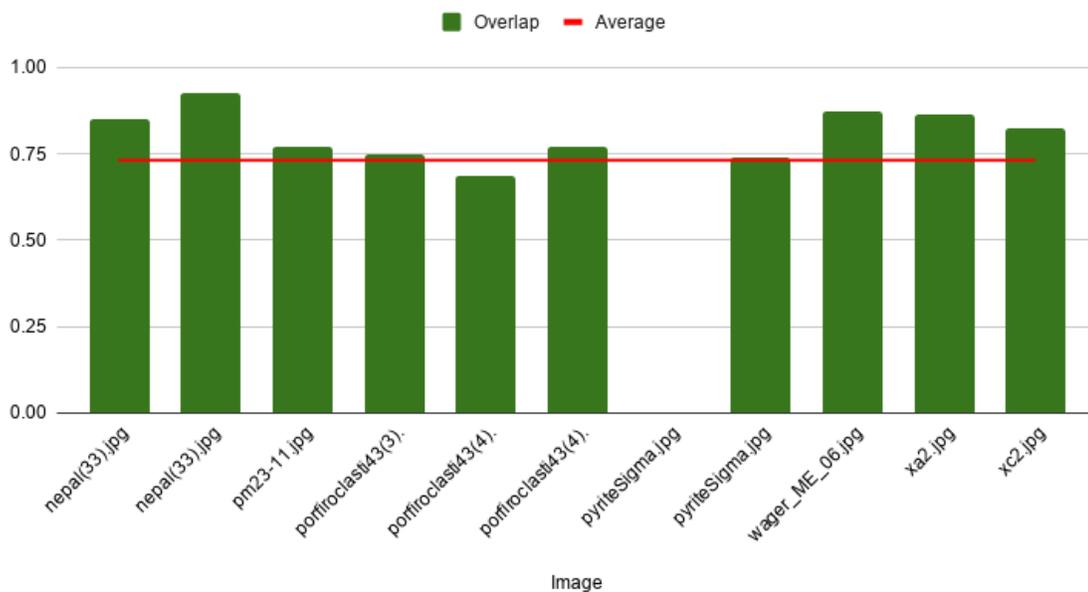
- Of the 10 'With' images, 11 detections occurred
- 10 detections output a Conf. above  $\geq 90\%$
- Average Conf. of 83%

### YOLO Confidence Scores on 'With' Test Set (alpha 100, full size)



- 7 detections had an accuracy above 75%
  - note, 'pyriteSigma.jpg' had 2 detections, one of which had a resulting accuracy of 0
- Average accuracy of 73% was observed

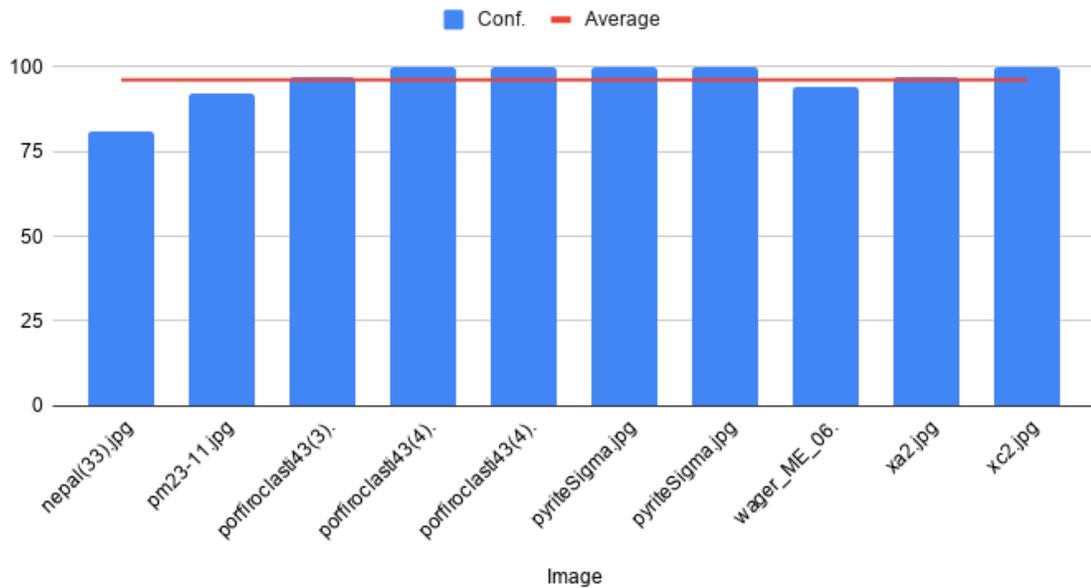
### YOLO Prediction-Truth Overlap on 'With' Test Set (alpha 100, full size)



Detections on 'With' Images (alpha 100, resize 128)

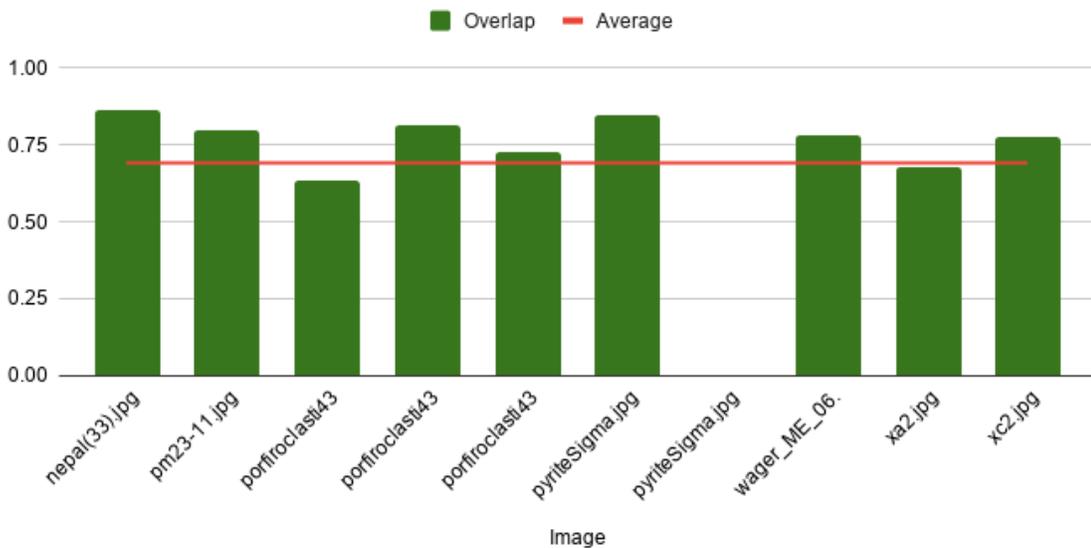
- Of the 10 'With' images, 10 detections occurred
- 9 detections output a Conf. above  $\geq 90\%$
- Average Conf. of 96%

### YOLO Confidence Scores on 'With' Test Set (alpha 100, resize 128)



- 6 detections had an accuracy above 75%
  - note, 'pyriteSigma.jpg' had 2 detections, one of which had a resulting accuracy of 0
- Average accuracy of 69% was observed

### YOLO Prediction-Truth Overlap on 'With' Test Set (alpha 100, resize 128)



### Detections on 'Without' Images (alpha 100, full size)

- 44 false positive detections occurred
  - 22 detections output Conf.  $\geq 90\%$
  - Average Conf. of 75% was observed

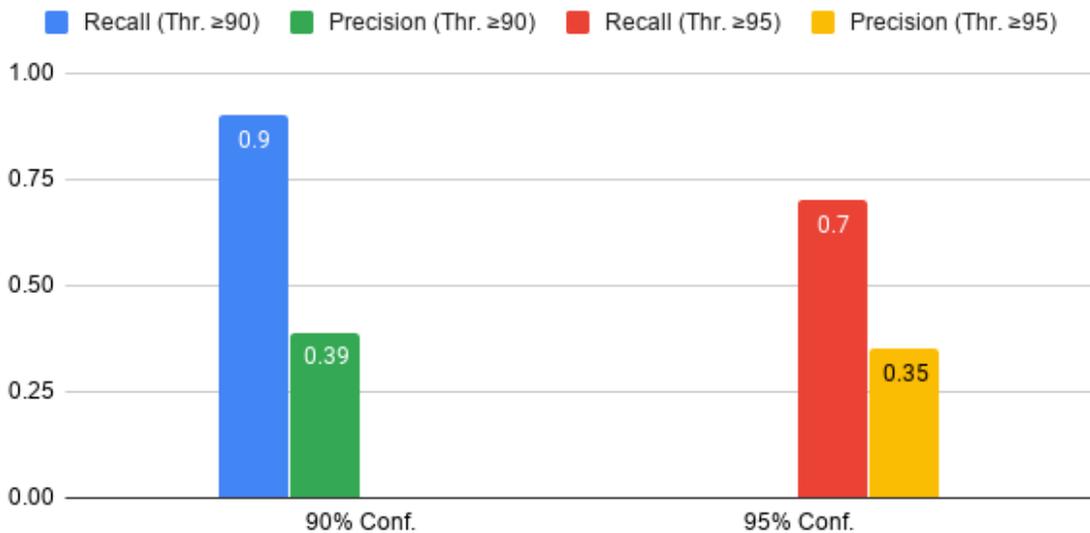


## Recall & Precision: 90% vs 95% Confidence Threshold on 'With' Test Set (alpha 100, full size)



- Recall and Precision for alpha 100, resize 128 test set
  - Calculated at Conf. 90% and 95%

## Recall & Precision: 90% vs 95% Confidence Threshold on 'With' Test Set (alpha 100, resize 128)



### Notable Comparisons of Results

- Let the *alpha 100, full size* test set be *Set A*
- Let the *alpha 100, resize 128* test set be *Set B*

### Prediction Confidence & Accuracy

- Set A average Conf. for predictions was 83%, Set B average was 96%
- Set A average accuracy of predictions was 73%, Set B average was 69%

This comparison shows that resizing may improve the confidence of detections. However, the quality of predictions goes down.

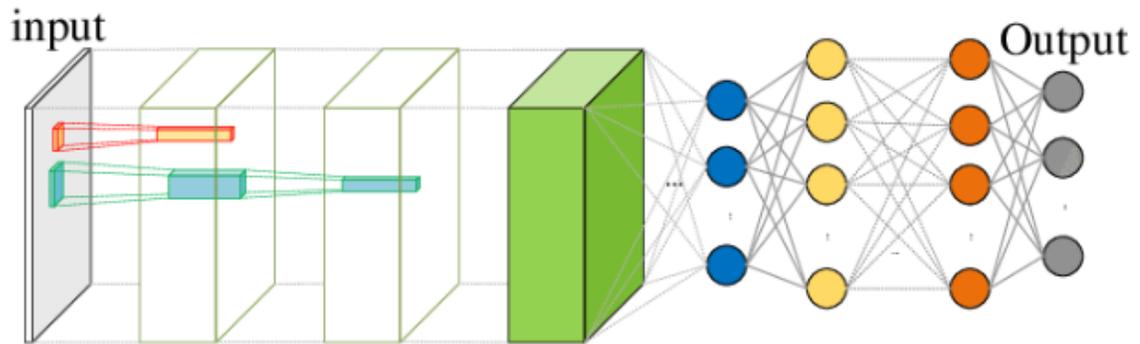
### **Precision & Recall**

- Precision saw marginal improvement from Set A to Set B
- Recall lowered from Set A to Set B
- Set B saw a 7% increase in precision

As Set B performed worse overall, its evident resizing does not help with YOLOv3 detection on denoised images.

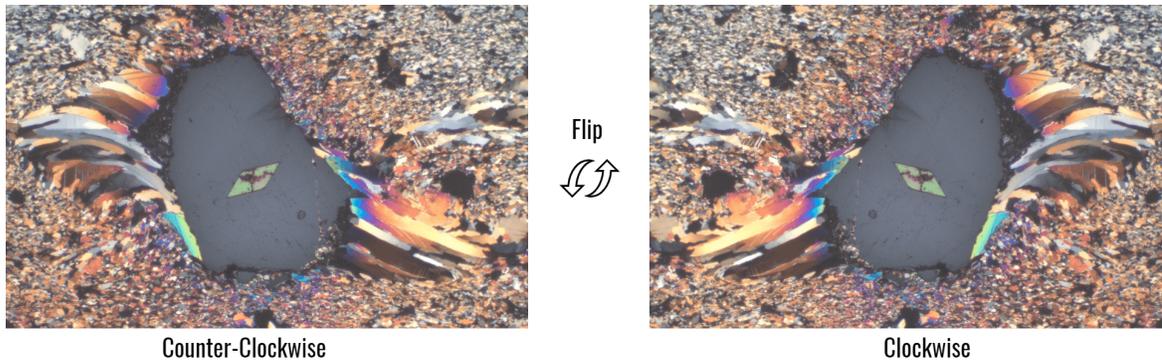
# METHODS: CW/CCW CLASSIFICATION

## Transfer Learning



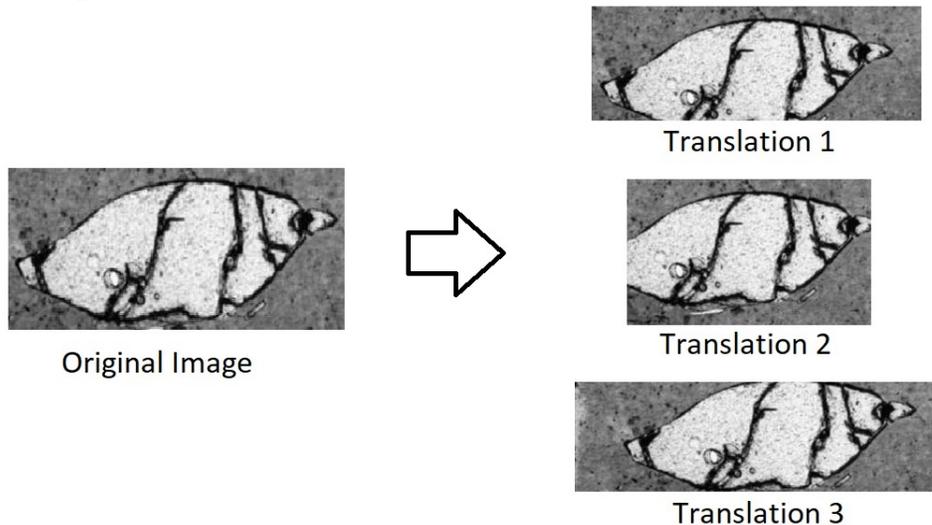
- Normally, CNNs must be trained on a dataset with millions of images to learn the intricate details in the trained images.
- To compensate, we used Transfer Learning:
  - A previously trained CNN as a basis for an entirely new network.
  - The new network is trained on domain specific data—in our case sheer sense clasts—in order to refine the model according to that data.

## Data Augmentation



- Data Augmentation is a technique used to artificially expand a dataset using various image modification techniques
- Transfer Learning still benefits from larger datasets, so we use data augmentation to overcome our lack of data

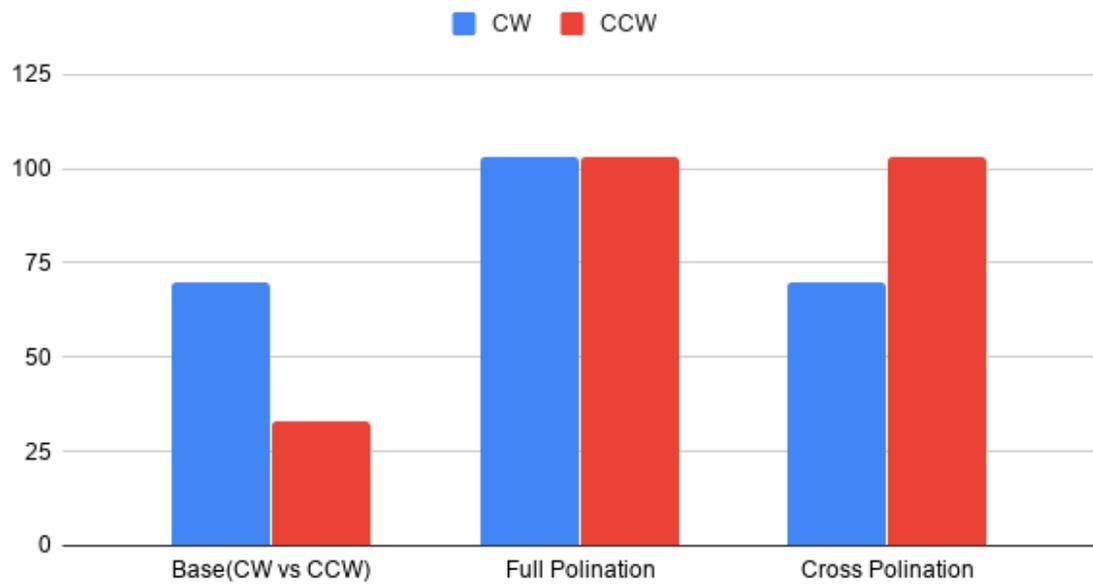
## Data Augmentation: Translation



- Borders of cropped image are randomly varied, each border independently

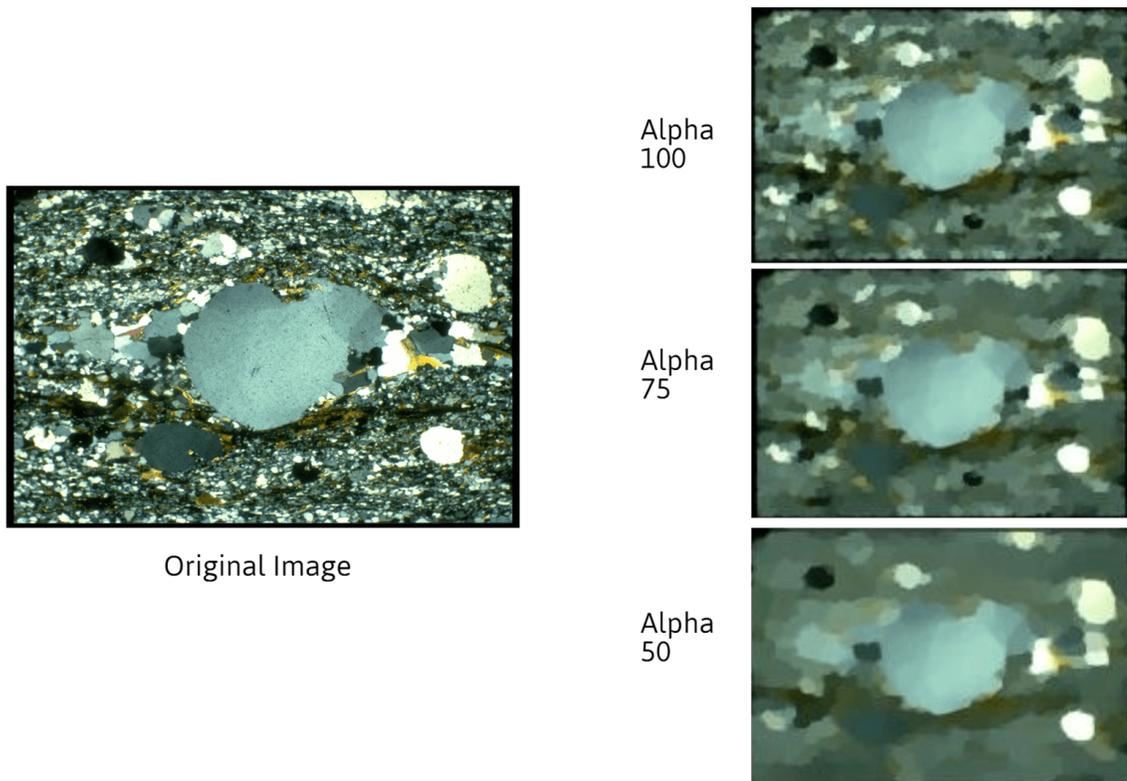
## Data Augmentation: Pollination

### Base(CW vs CCW), Full Polination and Cross Polination



- Because classification is based on direction of rotation, mirroring images changes their classification, allowing same image to be used in both categories
- Full Polination - All Images of both categories are mirrored, than added to other category
- Cross Polination - Only Images of larger category are mirrored and added to smaller category

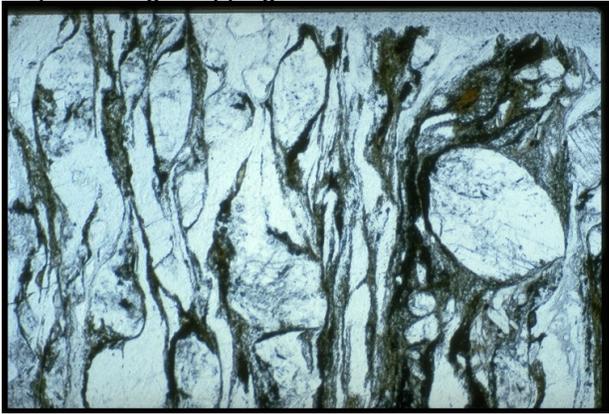
## Preprocessing: Denoising



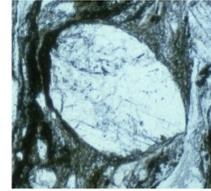
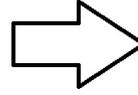
- Element in image under a certain size are removed, leading to a smoother noiseless version

- Used to remove unneeded information from images so CNN's can better learn underlying structures.

### Preprocessing: Cropping



Original



Resized

- Images are cropped to only show the Clast and its features
- Ideally, we would use the output of the YOLOv3 network to produce
- Currently, The ground truth labels of the clast crop for the images is used as an idealized substitute.

## Evaluation

### F1-Score

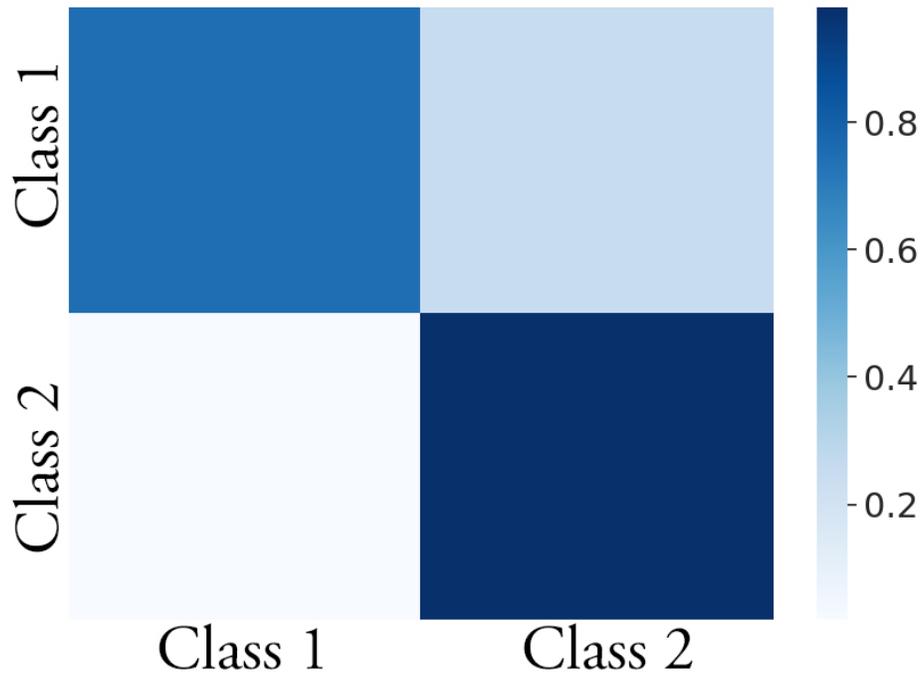
- F1 score is the metric used to evaluate the performance of a classifying CNN

$$F1-Score = 2 \cdot \frac{(P * R)}{(P + R)}$$

- The reason that F1 score is a useful metric is because both Precision and Recall can reveal a different type of error
  - low Precision meaning that the model has a high false positive rate
  - low Recall meaning the model has a high False Negative.

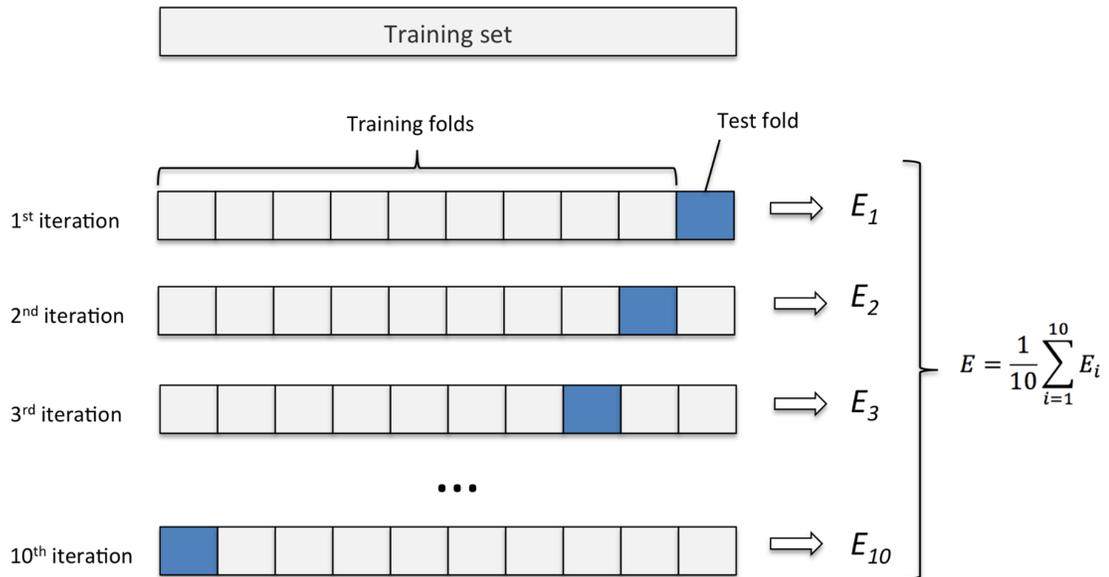
F1 score is preferred over R because a poor performance in classifying any class will result in a low overall F1 score, where as in Accuracy a low precision or recall can be masked by a high score in the other.

### Confusion Matrix



- Visualizes True Positive vs False Positive results for two given classes
- Expected on the left and model prediction on the bottom

### K-Fold Cross-Validation

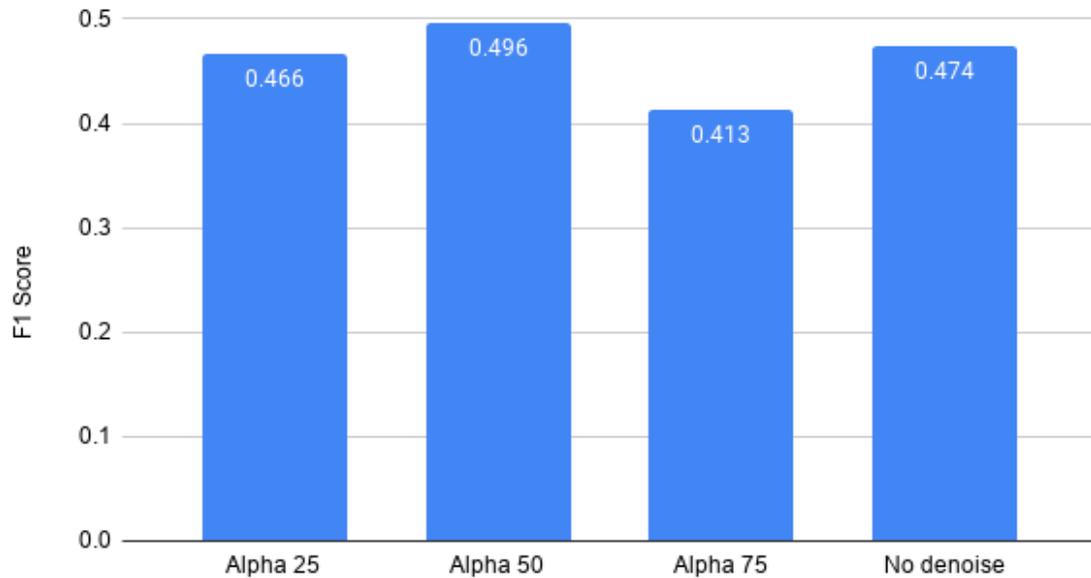


- Allows for automation of training and testing iterations
- Divides the dataset into K groups.
- For each K grouping, the model is trained on K-1 sets, then tested on the remaining set.

# RESULTS: CW/CCW CLASSIFICATION'

## Cropped and Denoised

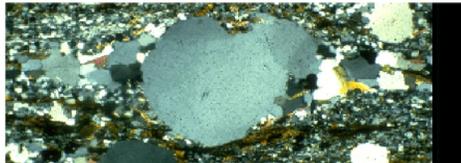
### F1 Scores



### Non Denoised Images Examples

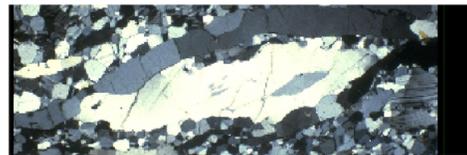
Key:  
Incorrect Prediction  
Correct Prediction

### Predicted CW



Prediction confidence:  
0.603

### Predicted CW



Prediction confidence:  
0.707

### Predicted CW



Prediction confidence:  
0.997

### Predicted CW

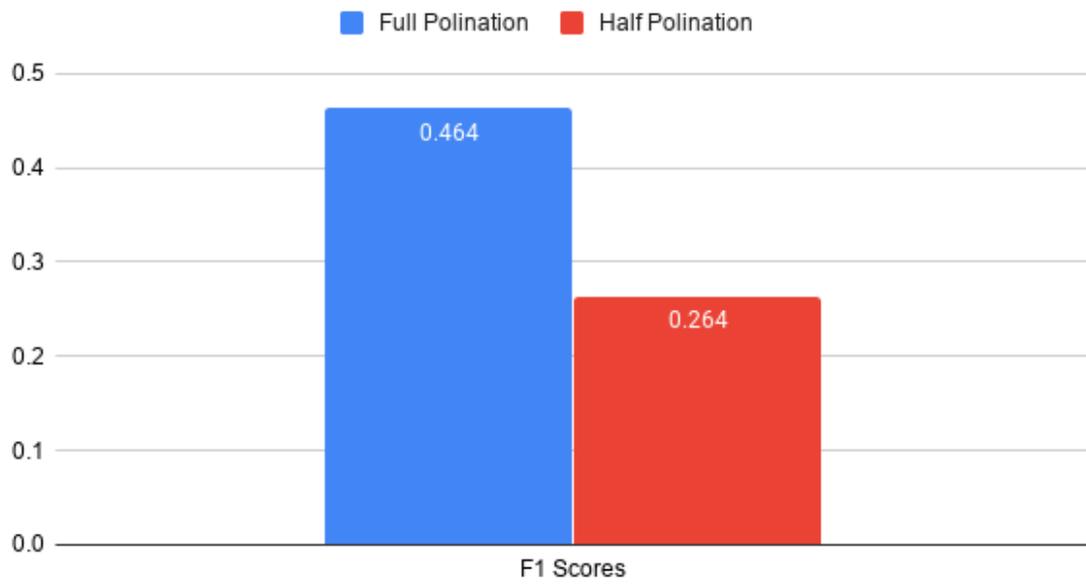


Prediction confidence  
0.984

- Images are first cropped to only include the sigma clasts, and then were denoised to a variety of denoising levels
- Best Denoising outperformed no denoising, but only by a slim margin

## Pollination

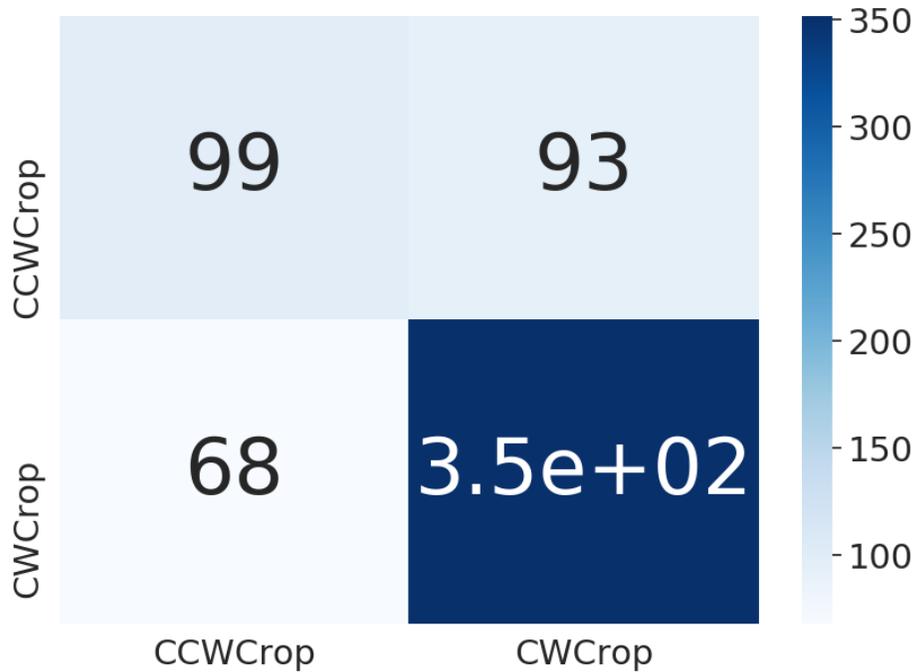
### Full Polination and Half Polination F1 scores



- Using a base of cropped images denoised to Alpha 50
- Overall, no improvement
- Slight decrease in performance for Full Pollination
- Half Pollination performance was poor

## Translated

Final CM



F1 Scores: CCWCrop: 0.531, CWCrop: 0.794,  
F1 Average Score: 0.663

Key:  
Incorrect Prediction  
Correct Prediction

Predicted CCW



Prediction confidence:  
0.570

Predicted CW



Prediction confidence:  
0.543

Predicted CW



Prediction confidence:  
0.866

Predicted CW



Prediction confidence:  
0.974

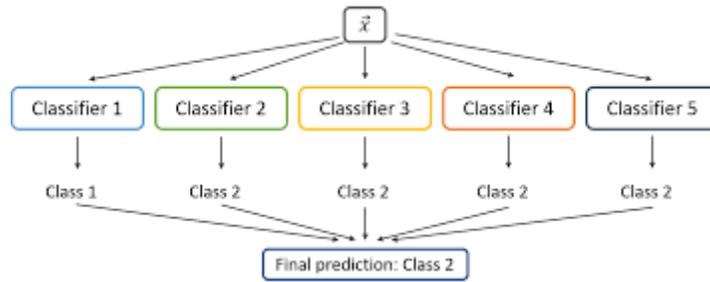
- Starting from baseline of cropped images denoised to an alpha of 50
- Each image had 5 random translations applied to it to get derivative images
- End result is 6 times as many total images

- Possible Bias: Translations were done before data was separated into k-folds, meaning that versions of the same image could have been in both the train and test sets, inflating results.



# FUTURE WORK

## Ensemble Classification



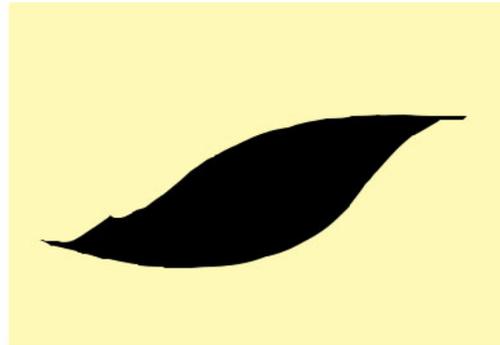
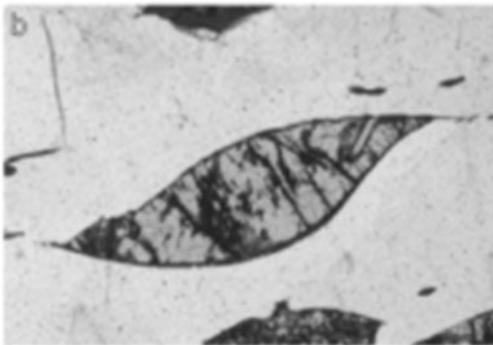
- Multiple models, all given the same input image, but possibly processed in different ways
- Each individual model outputs a prediction, which can then be combined to make an overall prediction.

## Improve YOLOv3 Detection

### Synthetic Images

- Synthetic images, in our domain, is any image that encapsulates the significant details of a shear-sense clast
  - Created by hand or with a script

# Example of a Synthetic Image



- Currently two options to test
  - Augmenting dataset by adding synthetic images
  - Train purely on synthetic images

### Increase Dataset Size

- Increasing dataset size may improve YOLO model accuracy
- Dataset can be buffed using different methods
  - Image Augmentations
  - Adding Synthetic Images
  - Gathering more real-life data

### Tuning Hyperparameters

- A hyperparameter is a configurable parameter external to chosen model
  - Tuned using dedicated analyses to optimize both training and detection
  - Tuned before training

- There are multiple hyperparameters one can tune within YOLOv3
  - Learning rate
  - Momentum
  - Optimizer

## **Integrate YOLOv3 Detection**

- Implement auto-cropping into current pipeline
- Auto-cropping will come before or after preprocessing, and before classification

# ABSTRACT

To incentivize the participation and contribution to the growth of an earth-science-based cyberinfrastructure, analytical environments need to be developed that allow automatic analysis and classification of data from connected data repositories. The purpose of this study is to investigate a machine learning technique for automatically detecting shear-sense-indicating clasts (i.e., sigma or delta clasts and mica fish) in photomicrographs, and finding their shear sense (i.e., sinistral (CCW) or dextral (CW) shearing). Previous work employed transfer learning, a technique in which a pre-trained Convolutional Neural Network (CNN) was repurposed, and artificially augmented image datasets to distinguish between CCW and CW shearing. Preprocessing images by denoising, a process in which noise at different scales is removed while preserving edges of an image, improved classification accuracy. However, upon randomizing the denoising parameters, the CNN model didn't converge due to severe lack of data. While the efforts for acquiring more labeled data is ongoing, this work compensated for it by implementing a pre-processing "detection" system that automatically crops images to regions of image containing the clasts. This is done by utilizing YOLOv3, a CNN based image detection system that outputs a bounding box around an object of interest. YOLOv3 was trained using 93 photomicrographs containing bounding boxes of 344 shear-sense-indicating clasts. The retrained detector was tested on two sets: set A with 10 photomicrographs containing clasts and set B with 100 photomicrographs not containing clasts. All but one of the clasts in set A were correctly detected with an average confidence score of 96.6%. On set B, 72% of images correctly did not indicate presence of clasts. On the remaining images, where clasts were incorrectly identified, an average confidence score of 78.3% was observed. By utilizing a threshold on the confidence scores, the system could be made more accurate. Future work involves utilizing the bounding boxes output by the detection system to refine and improve the CNN model for classifying shear sense of clasts in photomicrographs.

# REFERENCES

[1] YOLOv3 [pjreddie.com/media/files/papers/YOLOv3.pdf](https://pjreddie.com/media/files/papers/YOLOv3.pdf) (<https://pjreddie.com/media/files/papers/YOLOv3.pdf>)

[2] M. Holleman. "Real-time object detection with YOLO". *machine think BLOG*, 20 May 2017. [machinethink.net/blog/object-detection-with-yolo/](https://machinethink.net/blog/object-detection-with-yolo/) (<https://machinethink.net/blog/object-detection-with-yolo/>)