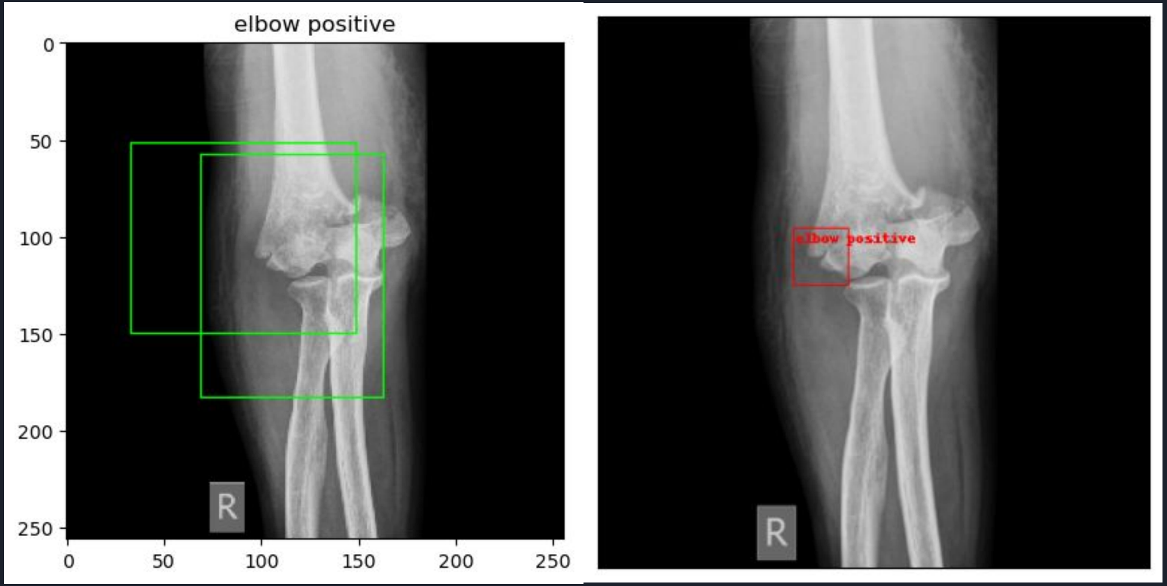
A COMPUTER VISION MODEL FOR DETECTING BONE FRACTURES IN THE UPPER EXTREMITIES

PROBLEM STATEMENT:

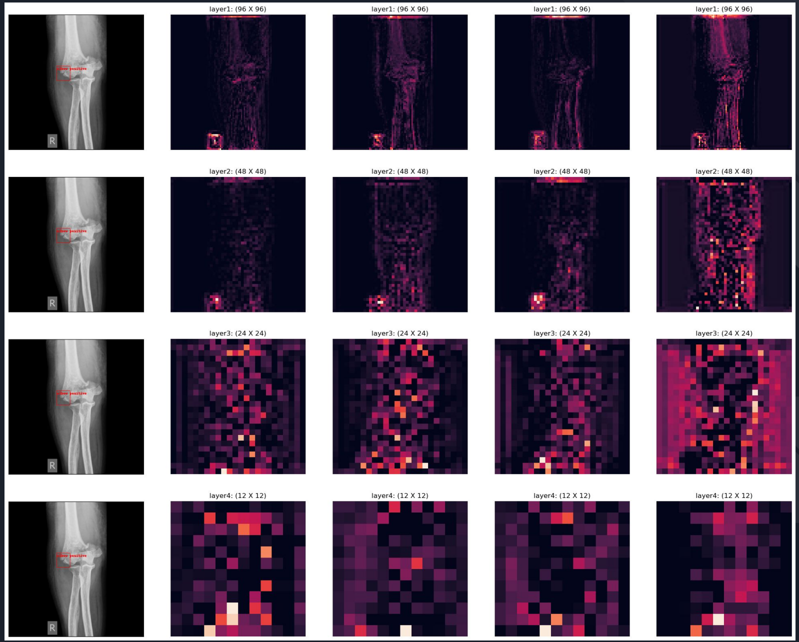
Build an AI over the course of a week that detects bone fractures in X-rays images at at least 85% accuracy to aid medical personnel in diagnosing fractures in the upper extremities and thus expediting treatment.

DATA WRANGLING

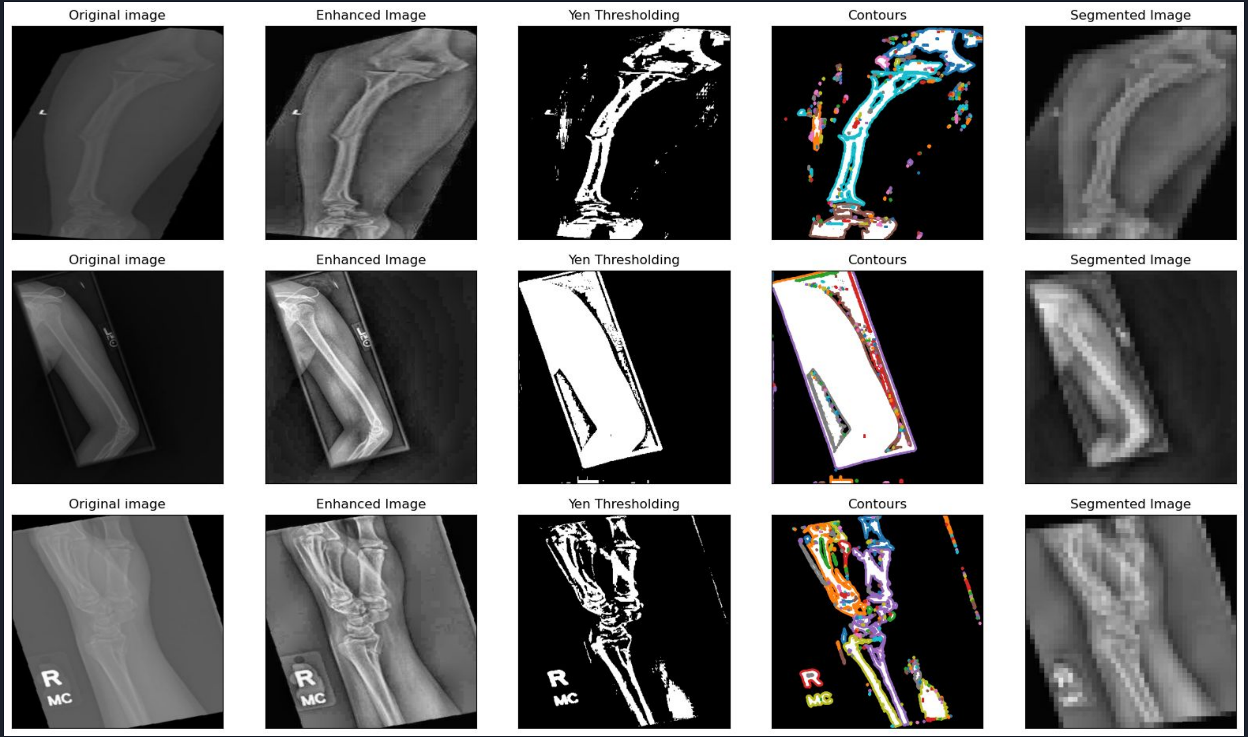


A before and after of the data wrangling process: the first image on the left is an initial representation of image and labeling data and the right image is of the final image with labeling data used for training a model.

EXPLORATORY DATA ANALYSIS/FEATURE ENGINEERING



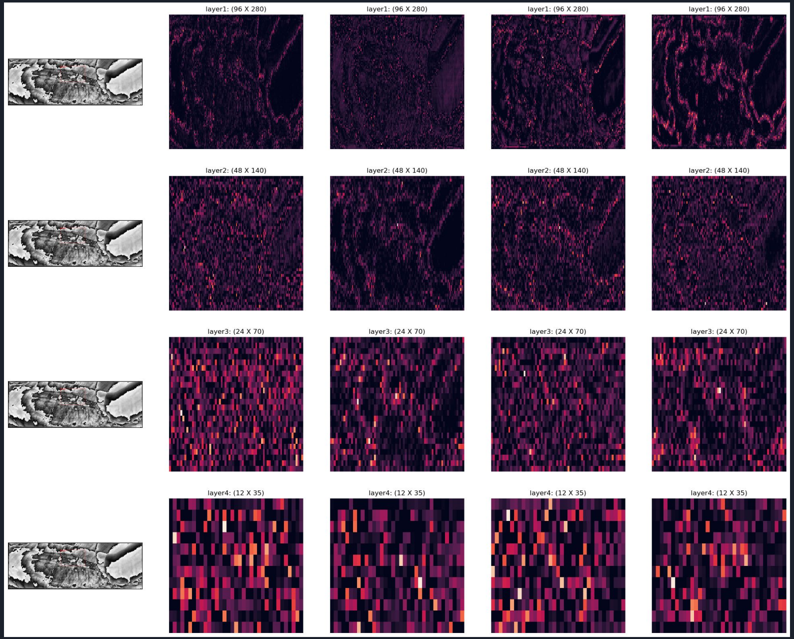
This image shows the analysis process of a test ResNet Convolutional Neural Network with a SoftMax Activation layer. (RCNN) Effectively as the model moves through it's hidden layers, the resolution of the image decreases allowing the model to find from finer to larger features in order to formulate a classification.



A quick cycle of images and their test transforms clearly indicating that there is no clean repeatable way to transform these images into a compatible Visualization that a Machine Learning model would effectively be able to interpret.

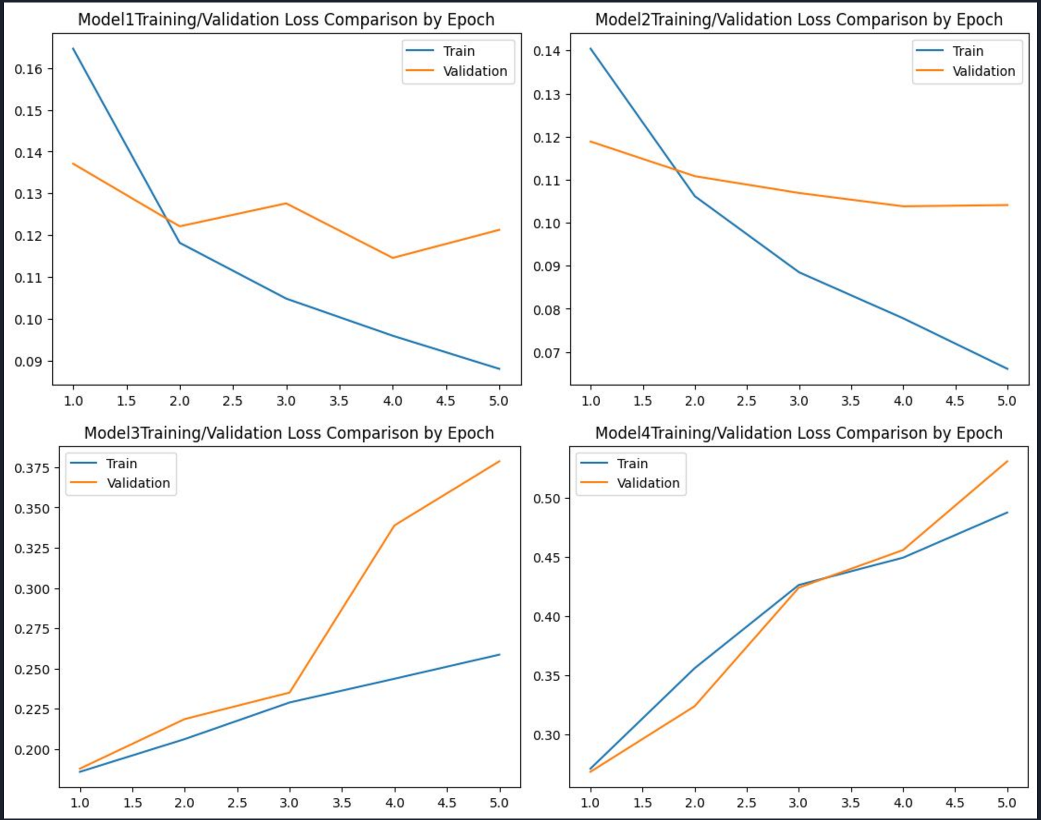
WHY A DEEP LEARNING SOLUTION RATHER THAN A MACHINE LEARNING MODEL

MODEL PRE-PROCESSING



- transforms.Resize(224)
- transform.ToTensor()
- transforms.RandomAutocontrast(0.1)
- transforms.RandomAdjustSharpness(0.1)
- transforms.RandomErasing(0.1)
- transforms.RandomHorizontalFlip(0.1)
- transforms.RandomInvert(0.1)

One of the issues I volved with this data set was it was highly idealized. Therefore the above methods were used to randomly transform %10 of per transform (so some may have been transformed twice!) in order to better simulate corrupt data in a real world setting.



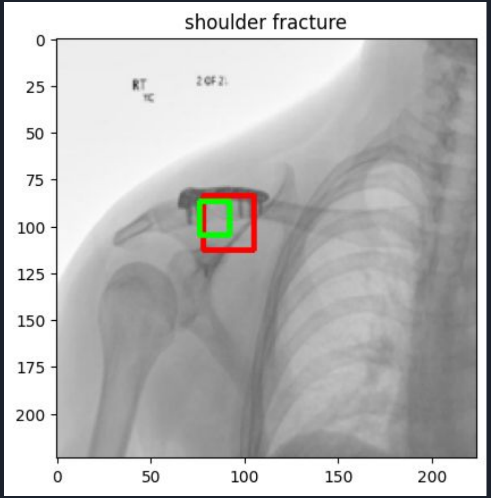
Here are training graphs from 4 different models.

- X is number of training epochs
- Y is loss.
- Loss is defined as the mean sum of the SoftMax error and Intersection on Union (IoU). Less is better.

The only two models worth considering are Model 1 and Model 2. Model 2 is slightly more accurate but substantially less performant.

MODELING: FOUR FASTER RCNN NETWORKS WITH DIFFERENT STRENGTHS

FINAL THOUGHTS



The FasterRCNNPredictor model utilizes the summation of two loss functions, CrossEntropy for multiclassifier, and L1 for the bounding boxes. The Training/validation cycles both took the sum of the averages of these scores to determine loss per epoch. As to be expected for a loss function, less is better, set on a scale from 0 to 1 so it can be considered as a percentage of loss. in this case, our best loss was .114 or %11.4 error in both classification, and Intersection over Union from the bounding boxes.

Next steps, would involve a finetuning process which per the PyTorch documentation is a COCO function which would require a refactoring of code to put 0 as the background or "No Fracture" classification as well as a new module installation and import. In a word it would require all but re-writing this code from scratch which we will be sure to do as an addendum to this project, however, at this time we will unreasonably be able to access the the specific IoU scores for the bounding boxes and in order to optimize it for False positives (Predict there is a fracture where there is none.) Pending that We would again rewrite code to accomodate the substantiation of backbones for the RCNN network as well as modification and adustment of layers and finetune again. but for the sake of this project we are already hitting desired outcomes, and will continue work in fine-tuning this model for production at a later date.