Analyzing X-ray images to find bone fractures.

Venkata Ramana Reddy Duggempudi and **Pranavi Sriya Vajha**
The State University of New York at Buffalo
vduggemp@buffalo.edu, pvajha@buffalo.edu
50539685, 50540897

Abstract

Automated bone fracture detection in X-ray images helps doctors to provide accurate and timely patient care. However, detecting the fractures near joints and hairline fractures is a big challenge due to the change in X-ray images quality and the characteristics of the fractures. Radiologists commonly use manual inspections in traditional methods, a time-consuming process vulnerable to human error. To help medical professionals, we proposed a project called bone fracture detection which uses computer vision and deep learning techniques like YOLO (You Only Look Once), Detection transformers and Faster R-CNN.

1 Introduction

The main objective of the project is to find the fracture location and create bounding box to it. And also predicts the type of bone fracture from the trained classes. The dataset is taken from the roboflow.

The dataset contains approximately 4000 X-ray images which are categorized in six classes. The classes are Elbow Positive, Fingers Positive, Forearm Fracture, Humerus Fracture, Shoulder Fracture, and Wrist Positive.

2 Problem Description

Here we are going to find the fracture in the bones. It is difficult to find the fracture which has thickness of hairline. And it is difficult to find the fracture near the joints.

• The quality of X-ray images contains lot of noise as compared to regular images, It is hard to read and find the fracture which has a thickness of hair line.

3 Data Augmentation

We have used the albumentations module to perform data augmentation in the case of object detection.

The data augmentation techniques we have performed are:

• Random Brightness Contrast: This augmentation is performed to mimic the variations in X-ray exposure levels which help the model to generalize the different lighting conditions.

• Elastic transform: This augmentation applies an elastic deformation to the image. This creates realistic and non-realistic distortions in images which simulate minor variations in the positioning of patients or the soft tissue around bones.

• Gaussian noise: This noise is added because it helps the model to perform better as the X-ray images have varying levels of noise due to different X-ray machine qualities.

• CLACHE (Contrast Limited Adaptive Histogram Equalization): This augmentation is used to enhance X-ray images by improving the visibility of subtle fractures and by enhancing local contrast without amplifying the noise.

• Sharpen: This augmentation is performed to enhance the edges and details of X-ray image which makes the features easier to perceive. This is useful to identify hairline fractures.

Here is the sample image before and after doing Data Albumentation

4 Models

4.1 YOLO v4

YOLO v4(You Only Look Once) is an object detection algorithm. It is mainly used in processing real time applications by using a single forward pass through the Neural Network. The architecture of YOLOv4 includes CSPDarknet53 as the backbone, PANet as the neck, and YOLOv3 as the detection head. The YOLO algorithm divides the input image into a grid of cells, and for each cell, it predicts the probability of the presence of an object and the bounding box coordinates of the object. It also predicts the class of the object. In the given task,

Figure 1: X-ray image before Data Augumentation

Figure 2: X-ray image after Data Augumentation

Image need to be classified and if the fracture is detected it is able to create bounding box for that.

4.2 Faster R-CNN

Faster R-CNN is a deep learning model that detects objects in images by using a convolutional neural network (CNN). It works by first identifying regions of interest (ROIs) in an image. The ROIs are then passed to a second network, which classifies the objects in each ROI and predicts their bounding boxes.

The fully convolutional network that simultaneously predicts object bounds and objectness scores at each position. The RPN (Region Proposal Network) is trained end-to-end to generate high-quality region proposals, which are used by Fast R-CNN for detection. RPN and Fast R-CNN are merged into a single network by sharing their convolutional features: the RPN component tells the unified network where to look.

4.3 Detection Transformer

Detection Transformers are a class of deep learning models designed for object detection tasks in computer vision. They extend the transformer architecture. Unlike traditional convolutional neural networks (CNNs), which process images in a hierarchical manner, Detection Transformers treat the image as a sequence of patches and process them in parallel.

These models excel at capturing global context and long-range dependencies within images, making them suitable for tasks like object detection, segmentation, and even image generation. By leveraging self-attention mechanisms, Detection Transformers can attend to relevant patches across the entire image simultaneously, enabling more efficient and accurate feature extraction.

5 Experimental Setup

5.1 Dataset

Dataset is downloaded from the roboflow. It consists train, validation and test data with it's bounding boxes.

5.2 Data Augmentation

The images are more noisy. By using data augmentation techniques, the model is able to generalize the images with different exposure levels, lightning conditions and sharper images.

5.3 Selection of model for training

We selected YOLO v4 as our primary model for the training. Generally, it is trained based on YOLO v8. The main reason for selecting YOLO v4 is balance between speed and accuracy. YOLO v4 is computationally less expensive as compared to YOLO v8.

Again we tried with Faster R-CNN and Detection Transformers. Both models are mainly used for object detection and Classification tasks in Medical Imaging. It is not performed well as compared to YOLO v4.

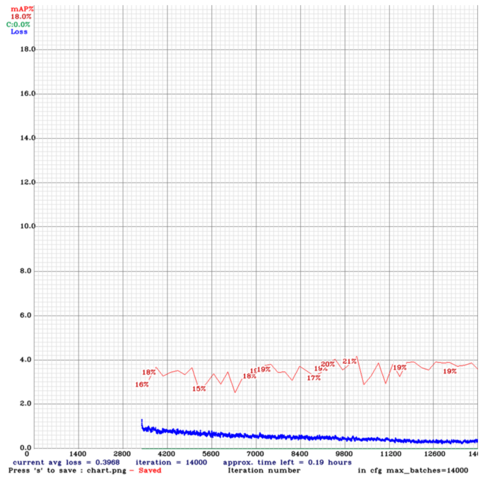

Figure 3: mAP score and loss values for YOLO v4 during training

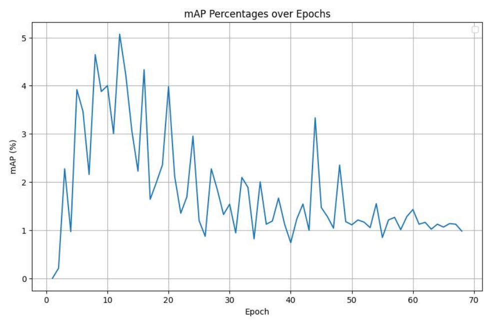

Figure 4: mAP score for Faster R-CNN during training

From the graphs: We trained YOLO v4 for 14000 epochs and recorded the maximum mAP as 20.08%. Whereas, we trained Faster R-CNN for 70 epochs and got the mAP as 5%.

Finally as mentioned in milestone 2. We tried using Detection transformers. After training the model with train and validation dataset, we got the loss values for train, validation and test dataset as follows.

Validate metric	DataLoader 0
validation/loss	0.47947242856025696
validation_cardinality_error	0.3640870153903961
validation_loss_bbox	0.02483379654586315
validation_loss_ce	0.05082133039832115
validation_loss_giou	0.15224115550518036

Figure 5: Loss values for train dataset in Detection Transformers

5.4 Validating and testing

Validated the model with the training weights and done hyper-parameter tuning based on the mAP values. We experimented with different batch_size and with the batch_size of 64 and sub divison of 8.

Validate metric	DataLoader 0
validation/loss	1.4056905508041382
validation_cardinality_error	0.4568965435028076
validation_loss_bbox	0.08251330256462097
validation_loss_ce	0.2321612536907196
validation_loss_giou	0.3804813623428345

Figure 6: Loss values for validation dataset in Detection Transformers

Validate metric	DataLoader 0
validation/loss	1.5155158042907715
validation_cardinality_error	0.48520711064338684
validation_loss_bbox	0.08811526745557785
validation_loss_ce	0.2341357171535492
validation_loss_giou	0.42040181159973145

Figure 7: Loss values for test dataset in Detection Transformers

5.5 Findings

During training the mAP of 20.75% which is more as compared to the roboflow mAP which is 20.6. The average loss is 0.68. It is because of Data Augmentation which is done in a prepossessing step.

Here from Table 2, by comparing the loss values. It concludes that the YOLOv4 is performing better as compared to Detection Transformers. But when it comes to predictions, Detection transformers is predicting well with the test dataset.

It is confusing that YOLOv4 is creating better bounding boxes as compared to Detection transformer. It may be due to Data Augumentation technique which is used in YOLOv4.

6 Results

For evaluation we used specified metrics called mAP (Mean Average Precision).

mAP is calculated as follows:

$$\mathrm{mAP} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{AP}_i$$

where n is the number of classes, and AP_i is the average precision of class i.

Precision is calculated as follows:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (1)$$

Recall is calculated as follows:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (2)$$

For Faster R-CNN the model is trained for 68 epochs and mAP values are

Model	mAP	Precision	Recall
roboflow	20.6%	37.7%	21.5%
YOLO v4	18.8%	51%	24%
Faster R-CNN	5%	1.38%	18%

Table 1: Performance comparison of models.

Model	loss
Detection Transformer	0.4794
YOLO v4	0.3967

Table 2: Performance comparison of models by using loss values.

From the predicted images it finds the fracture location and created a bounding box with mentioning the types of class. It is able to find those fractures by including its confidence score. All the YOLOv4 results are from validation dataset. It is unable to predict the fracutre in test dataset.

7 Conclusion

By experimenting with these three models, Detection transformer is working well to classify the image type and create bounding boxes as compared to remaining two models. The main reason behind this is it's attention mechanism and number of parameters. The number of epochs used to train the models are YOLOv4 (14000 epochs), Faster R-CNN (70 epochs) and for Detection Transformers (75 epochs).

From the three models only Detection transformer is able to predict the Bone fracture class and creating bounding boxes.

8 Challenges

The major challenge is with the computational resource to train the models. Especially with Detection transformers. Because it consists of 41.5 Millon parameters.

Due to the computational resource limitation we trained detection transformer for 75 epochs. By using A100 gpu in colab pro it took 32 hours to train detection transformer itself.

References

1. YOLOv4: Optimal Speed and Accuracy of Object Detection
2. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks
3. End-to-End Object Detection with Transformers
4. Roboflow
5. Latex for report

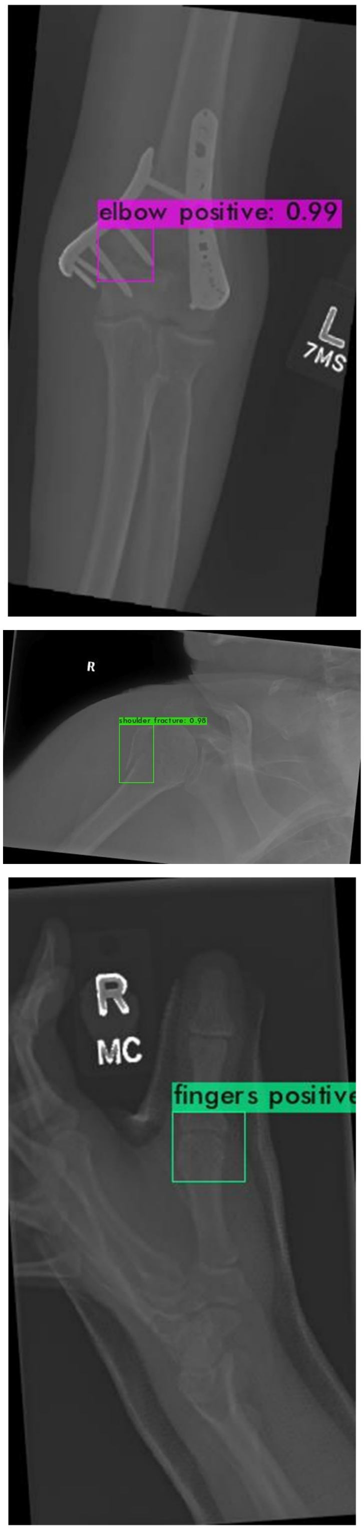

Figure 8: First set of prediction Images with YOLO v4 Validation dataset

Figure 9: Second set of prediction Images with YOLO v4 validation dataset

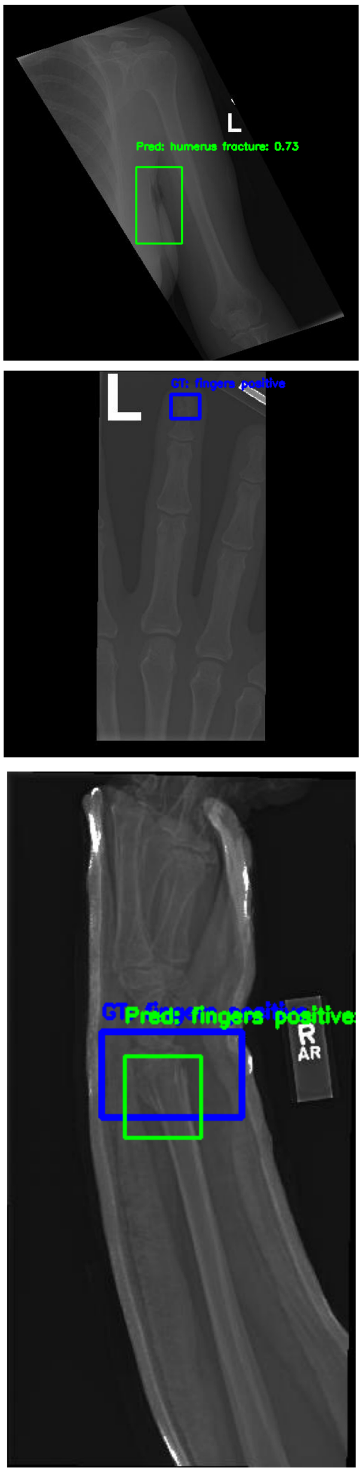

Figure 10: Sample prediction with Detection Transformer

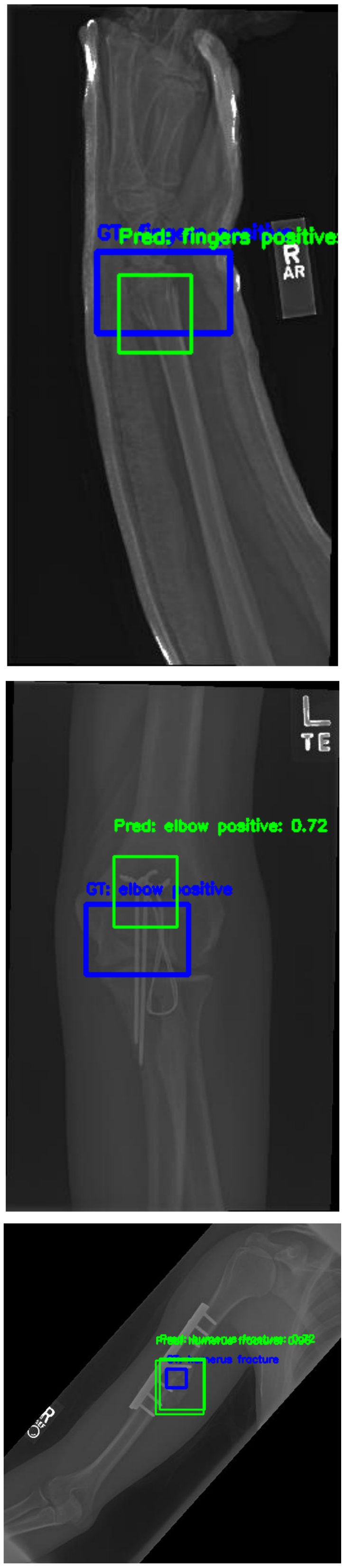

Figure 11: Sample prediction with Detection Transformer

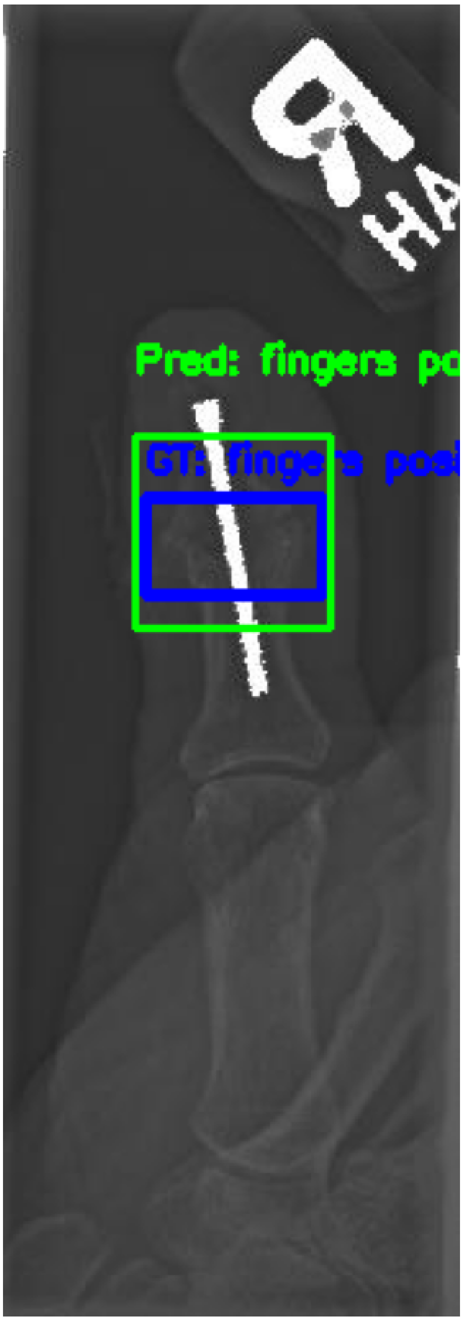

Figure 12: Sample prediction with Detection Transformer