Bone Fracture Detection

Giuglan Cătălin-Ionuț
Faculty of Automatic Control and Computer Science
UNSTPB
Bucharest, Romania
catalingiuglan@yahoo.com

Brutaru-Mihăilișca Bogdan-Alexandru
Faculty of Automatic Control and Computer Science
UNSTPB
Bucharest, Romania
brutaru.m.bogdan.a@gmail.com

I. Dataset Description

We use, for this project, a publicly available medical imaging dataset originally designed for multi-class fracture classification. This contains X-ray images labeled according to specific fracture types, such as wrist fractures, ankle fractures, and shoulder fractures, and images without fractures. The initial purpose of this dataset was to support fine-grained fracture recognition in clinical decision-support systems.

A. Composition of Dataset

The dataset collected contains approximately **3600 images**, covering multiple anatomical regions. Each image is provided in standard radiographic format, grayscale, high-resolution, accompanied by its corresponding annotation stored in YAML files. The initial YAML annotations contain detailed class labels describing the type or location of the fracture.

B. Modification of the Original Labels

Since the goal of our system is binary fracture detection, an input image will either have or not have a fracture; the original multi-class label schema was simplified. All fracture-related classes were combined and placed under a common class labeled ”fracture”.

This change was made directly in the YAML annotation files to ensure compatibility with the training pipeline and to reduce the complexity of the classification. Such conversion of this dataset into a binary task is in line with the project goal of developing an initial detection system, not a fine-grained diagnostic tool.

C. Data Quality and Preprocessing

The data has generally good quality with clear radiographic images suitable for computer-vision processing. However, some preprocessing was still necessary:

- Removal of corrupted or unreadable images
- Normalization and resizing to a standard input dimension for the baseline models
- Verification and correction of labels after merging classes
- Optional contrast enhancement for some low-visibility samples

D. Dataset Rationale

It provides an adequate number of real medical images with a variety of anatomical regions and a relatively balanced distribution of fractured versus non-fractured cases; thus, after merging, it is suitable for the development and evaluation of baseline models in performing binary fracture detection.

II. Baseline Description

As a baseline, we implemented **YOLOv8** to detect whether an X-ray image contains a fracture. YOLOv8 was chosen due to its strong performance in object detection tasks and its ability to detect features in medical images with high spatial resolution.

The model was trained on the preprocessed dataset using the following configuration:

- Task: detect
- Model: yolov8n.pt
- Epochs: 150
- Batch size: 16
- Image size: 256x256
- Optimizer: auto
- Pretrained: true
- Other standard YOLOv8 defaults (cosine LR, augmentation off, single class false)

III. Initial Results

The baseline YOLOv8 model was evaluated on the training, test and valid sets. The results are summarized in Table I.

Metric	Train Set	Test Set	Valid Set
mAP@0.5	0.5013	0.1854	0.2501
mAP@0.5:0.95	0.2220	0.0625	0.0813
Precision	0.6349	0.2367	0.4555
Recall	0.4516	0.1979	0.2549

TABLE I: Baseline YOLOv8 performance on binary fracture detection.

IV. Result Analysis

The results indicate that the baseline YOLOv8 model performs reasonably well on the training set but shows a significant drop in performance on the test set, with the valid set performing somewhere in between. This suggests that the model is overfitting and may not generalize well to unseen data.

Additionally, the clear difference between training and testing performance suggests that the model is learning dataset-specific patterns rather than general radiographic features. This is typical for small datasets or datasets with high intra-class variation, both of which apply in our case. X-ray images vary significantly in quality, anatomy, exposure, and patient positioning, making generalization a difficult task even for advanced deep-learning models.

Another important aspect is that fracture detection is inherently challenging because fractures can be extremely subtle, sometimes nearly invisible without domain expertise. Thus, even high-performing models may require more specialized training procedures to approach clinically relevant performance.

V. Challenges and Limitations

During the development of the baseline model, we observed that the obtained results, although functional, are significantly below the expected performance of modern medical imaging systems. Despite this, the results reflect the best configuration we were able to achieve after experimenting with multiple setups.

We tested various combinations of parameters, including:

- different learning rates and optimizers,
- multiple input resolutions (from 256 to 512 pixels),
- several YOLO variants (YOLOv8n, YOLOv8s),
- different augmentation techniques,
- changing batch sizes and training schedules.

Each tested approach led to either overfitting or underfitting, and the baseline configuration presented in this document represents the most stable and reproducible result we could obtain given our constraints.

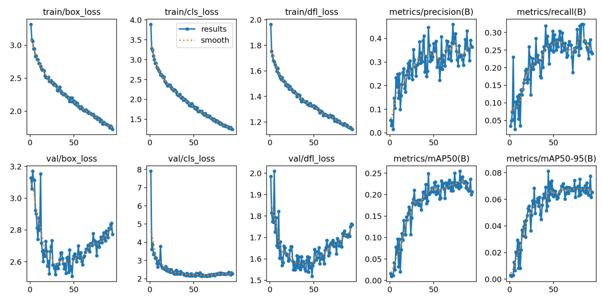
A major limitation throughout the process was the hardware available for training. Both laptops used in the project have limited GPU capability (6GB), and the RAM available is also limited (16 GB). This significantly increased training time and limited our ability to experiment with larger models or more complex architectures.

Although we also used Google Colab to supplement our local hardware, the free version offers only modest GPU resources and strict time limits. As a result, the performance improvement was not substantial, and the limited session duration restricted our ability to run longer or more computationally demanding training experiments.

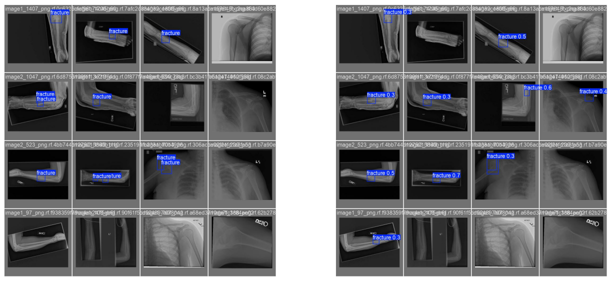
Despite these limitations, the current results establish a functional baseline. In the next project milestone, we aim to refine the model further, addressing overfitting and improving robustness while optimizing under the same hardware constraints.

VI. Visual Results

Figure 1 summarizes the performance of our baseline YOLOv8 model. The first subfigure shows the training metrics, including loss and mAP over epochs. The second subfigure compares the expected outputs (ground truth) with the model's predictions on a batch of X-ray images, highlighting how well the model detects fractures.



(a) Training metrics: loss and mAP over epochs.



(b) Comparison of expected outputs (left) vs. YOLOv8 predictions (right).

Fig. 1: Visual illustration of the model's performance.

VII. Future Work

For the next stage of the project, we plan to improve the performance of our baseline YOLOv8 model through fine-tuning and additional optimizations. This will include:

- Hyperparameter tuning to better adapt the model to the binary fracture detection task.
- Applying data augmentation techniques to increase the diversity of the training set and reduce overfitting.
- Experimenting with different backbone architectures or ensemble approaches to capture more subtle features in X-ray images.
- Evaluating additional loss functions and training strategies to improve precision and recall, particularly on challenging cases.

Beyond these improvements, we also plan to explore additional tweaks to the current model, such as small adjustments to the training process or minor variations in the data preprocessing pipeline. These changes are aimed at improving stability and overall performance without fundamentally altering the model architecture.

Finally, we intend to strengthen our validation approach by refining the dataset splits and monitoring the model's performance more carefully on unseen data. This will help ensure that any improvements observed are consistent and reliable, while staying within the scope of the current experimental setup.

References

[1] Justin Bolivar, “Bone Fracture Detection Using YOLOv8”, Kaggle, 2024. https://www.kaggle.com/code/justinbolivar/bone-fracture-detection-using-yolov8/notebook