Fracture Detection

1 Description

A fracture is a medical condition in which there is a partial or complete break in the continuity of the bone. Fractures are among the most common reasons for patients to visit emergency departments (EDs), and X-ray imaging is the primary diagnostic tool used by clinicians to assess patients for fractures. Missing a fracture in a radiograph often has severe consequences for patients, resulting in delayed treatment and poor recovery of function. Nevertheless, sometimes the size of fractures is not significant and detecting fractures is time-consuming for clinicians. Therefore, to assist clinicians in diagnosis, effective and intelligent systems that automatically detect fractures should be designed.

In recent years, with the development of deep learning, great advances have been made in the field of computer vision [1, 2, 3, 4, 5, 6, 7]. In this project, your challenge is to build an algorithm to detect fractures in chest x-ray images (Figure 1). You'll develop your solution using a dataset with 500 images and you can use any method.

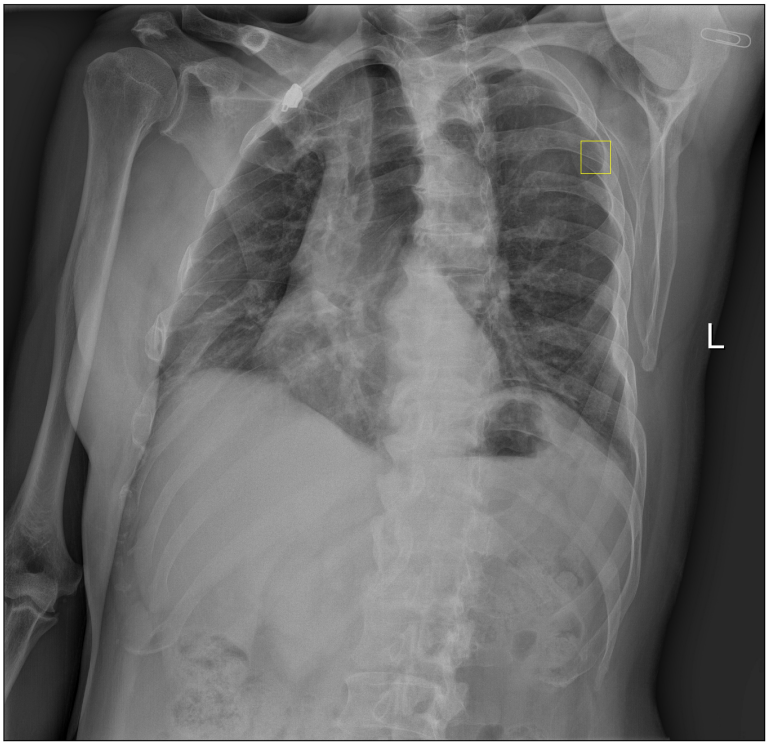

Figure 1: A chest x-ray image. The yellow box indicates a fracture.

2 Dataset

The dataset is divided into a training set and a validation set, which consists of 450 and 50 images, respectively. The annotations have the same format as COCO[8], you can find the explanation of the format at [9]

You can download the dataset at

`https://disk.pku.edu.cn:443/link/1B5659EC21540E6B1285A77F643F6FD0`

(password: xLxb)

3 Evaluation

Results are evaluated by computing AP_{50}. You can find the definition of AP_{50} at [10, 11]. In addition, we will test your submitted model on a test set which is not accessible to you. You should use cocoapi[12] to evaluate the results.

4 Submission

You should submit the followings:

- Report. The performance on validation set should be included in your report.
- Model. You should submit a single model for inference.
- Code. The code for training and inference should be included in your submitted codes. Pytorch[13] is recommended and you can refer to Detectron2[14]. One restriction to note is that the code for inference should be run like this:

```
python test.py --data_dir /path/to/fracture/test \
               --anno_path /path/to/anno_test.json \
               --output_path /path/to/output/results.json \
               --model_path /path/to/model/file
```

  and output a JSON file to `output_path`, which has the same format as COCO, just like[15].

The submission deadline will be notified later. You can use any natural-image dataset, but if you want to use external data about fractures, you must receive permission from *TA-Shishuang Zhao*.

References

[1] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[2] Ross Girshick. Fast R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015.

[3] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015.

[4] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells III, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*, volume 9351 of *Lecture Notes in Computer Science*, pages 234–241. Springer, 2015.

[5] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, volume 9905 of *Lecture Notes in Computer Science*, pages 21–37. Springer, 2016.

[6] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[7] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[8] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014.

[9] `http://cocodataset.org/#format-data`.

[10] `http://cocodataset.org/#detection-eval`.

[11] `https://github.com/rafaelpadilla/Object-Detection-Metrics`.

[12] `https://github.com/cocodataset/cocoapi/blob/master/PythonAPI/pycocoEvalDemo.ipynb`.

[13] `https://github.com/pytorch/pytorch`.

[14] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[15] `https://github.com/cocodataset/cocoapi/blob/master/results/instances_val2014_fakebbox100_results.json`.