Bone Fracture Detection in X-Ray Images Using Faster R-CNN with ResNet-50 Backbone

Mrs.Potnuri Gayatri[1], Lalam Rohini[2], Anusuri Brainard[3], R.N.V Vivek[4], Geddam Bhanu[5]
Associate Professor[1], Student[2], Student[3], Student[4], Student[5]

[1,2,3,]*Department of Artificial Intelligence*
Chaitanya Engineering College, Visakhapatnam, Andhra Pradesh, India
{gayatripotnuri119@gmail.com , chinna161960@gmail.com , brainardanusu3@gmail.com, vivekrajamandrapu21@gmail.com , bhanugeddam13013@gmail.com}

Abstract

The most frequent musculoskeletal injuries that we encounter in the emergency and orthopedics are bone fractures. Typically, they are visualized by sorting through X-rays in a hand-processed manner by a trained radiologist and that may result in small and large fractures going undetected due to exhaustion or uneven discerning, particularly when the fractures are small or hair-line. Herein this paper, we implement an automated bone-fracture detection system developed following a two-stage deep-learning application with a Faster-RCNN based on a ResNet-50 backbone. The first stage Faster-RCNN is the Region Proposal Network which initially identifies potential fractured sites and the predictions are later refined by the regressor classifier and bounding-box. Behind it, we trained Mill on ImageNet and soft deterrented it on a varied annotated dataset of X-rays which accounted with numerous varieties of fracture. We even had the model behind a barebones Flask application and thus the clinicians can simply drop pictures and obtain immediate outcomes, with bounding-box overlays and confidence figures. The results of our experiments demonstrate that the system achieves a solid mean Average Precision (mAP) of 87.3per cent and can operate at a fast pace, allowing its use in the clinic in near-real-time. Comparing it to in single to stage detectors such as YOLOv3 and EfficientDet, the two stage architecture produced a better localization of those small, low contraction fractures. This is aimed at assisting radiologists and not to displace them.
Index Terms Bone fracture detection, Faster-RCNN, ResNet-50, deep learning, medical image detection, convolutional neural network, object detection.

I. Introduction

A fracture has become a major clinical burden all over the world where millions of cases are being reported in ED and ortho clinics every year. An early diagnosis particularly through X-rays is vital in selecting the most effective mode of treatment and preventing such complications as malunion or avascular necrosis in the long run [1]. X-ray images are placed at the top of our list of screening tools due to their cheap cheapness, fastness and the numerous resources, however it requires a high degree of skills to interpret the images. Research indicates that the rate of fracture miss may range between 0.02 02 to 10 010 in the ER, depending on the type of fracture and the part of the body [2]. Hair-line fractures, minute cortical tears and complicated multiple-fragment injuries are particularly difficult to human beings to detect.

The overload of digital images, the burdensome work of radiologists formed a genuine challenge towards the automated computer-aided detection (CAD). Deep learning, and in particular convolutional neural networks (CNNs), have already achieved massive advancement in medical imaging, including identifying lesions, drawing up tumors, and diagnosing illnesses [3].

The concept of object detection has been continuously developing as a result of R-CNN, Fast R-CNN, and Faster R-CNN, which is more efficient as it combines proposal and classification into a single network that can be trained [4]. Single levels (e.g. YOLO and SSD) detectors have super fast speeds on the other end and are not as accurate due to small or crowded objects.

This paper presents an automated fracture detector, which is based on Faster -RCNN with a backbone of ResNet -50. The first stage in pipeline is the high quality proposals invoked out of the RPN, followed by classification and quality refining. The network is being operated within a Flask web application to have a realistic clinical interface. We compare it to other detectors in order to demonstrate the mAP and localization stability gain.

The remainder of the paper is structured in the following way: Section II is the review of related literature. Part III describes our methods and layout of system. The section V displays experiments and discusses the outcomes. Lastly, Section 5 concludes and proposes future research direction.

II. Related Work

There are multiple ML and DL waves that have been used to detect fractures automatically. The first CAD systems relied on manually designed tools such as Histograms of Oriented Gradients (HOG), Gabor filters and morphological processors to the support vector machines [5]. Although those were feasible, they could not be generalised across factors with different fracture types and imaging peculiarities.

Add CNNs and reach a more competitive feature extraction power. Rajpurkar et al. demonstrated that deep residual networks would be able to detect a large portion of the chest-X-ray pathologies with the performance comparable to that of radiologists, and marked the beginning of medical CNNs [6]. This concept spilled over to the wrists, hips, and spine bone imaging.

Redmon and Farhadi launched the YOLO family yolo-based detector-like single-stage detectors, which frame the task of detection as a regression problem with a limited set of parameters [7]. YOLOv3 was able to provide real-time performance and compromise on small, or overlapping objects performance compared to two-stage counterparts. EfficientDet, by Tan et al. [8], intelligently trades a slight amount of detail against speed by scaling backbone, feature, and prediction modules.

The transformer-based end-to-end detector, introduced by Carion et al, has terminated anchor engineering and non-maximum suppression. There, it is powerful but slower to converge and a cubload of data is required to make the nail.

Ren et al. introduced us to Faster-RCNN [4] that combines proposal generation (RPN) and detection steps on common feature maps. This makes it less difficult than traditional selective search methods in terms of burden of searching to find anchors. Faster -RCNN on COCO and Pascal VOC, ResNet -50 or ResNet -101 always puts COCO and Pascal on top with regards to the numbers.

He et al. demonstrated that residual shortcuts make deep net gradients healthy, and ResNet-50, a 50-layer bottleneck network, to be a sweet spot between representation power and runtime efficiency, which is ideal in medical applications.

And finally, Litjens et al. analyzed the topic of DL applied in the medical imaging field, as ImageNet transfer learning reduces the number of large manual datasets by hand that are often required in fracture detection when you je-just-need-some-experienced-radiologist-label-the-ground-truth.

III. Methodology and System Design

A. System Architecture Overview

The bone fracture detection system is divided into five major components: retrieving the X -rays and cleaning them, feature extraction, region suggestion, fracture classification and location and visualizing the results on a web site. The general plan is depicted in Fig. 1.

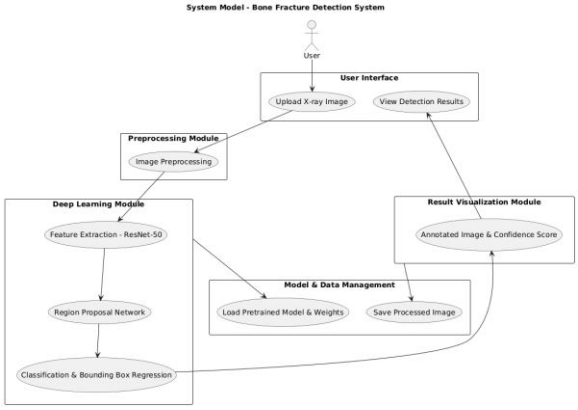

Fig. 1. Overall system architecture of the proposed bone fracture detection pipeline.

B. Dataset and Annotation

The dataset includes a collection of X-ray images of multiple bones, i.e., lower limbs, shoulders, elbow, hand, and wrist. Pictures belong to the types of the public sources, as well as certain clinical scans. Bounding boxes were drawn by the use of LabelImg on all apparent fracture spots under the supervision of trained annotators and a radiologist.

We had 2847 broken and 1154 healthy images. The data had been divided into training, validation, and testing 80:10:10. We applied augmentations; random flips, change of brightness and random crop during training to make the model stronger.

C. Preprocessing Pipeline

To begin with, X-ray images are transformed into RGB to make it compatible with ImageNet-pretrained backbone. We downsize them to a 800 800 pixel grid by zero padding the aspect ratio. After that, we normalize the pixel values using ImageNet statistics:

$\hat{x} = (x - \mu) / \sigma(1)$

where $\mu = [0.485, 0.456, 0.406]$ and $\sigma = [0.229, 0.224, 0.225]$ correspond to the ImageNet mean and standard deviation vectors per channel.

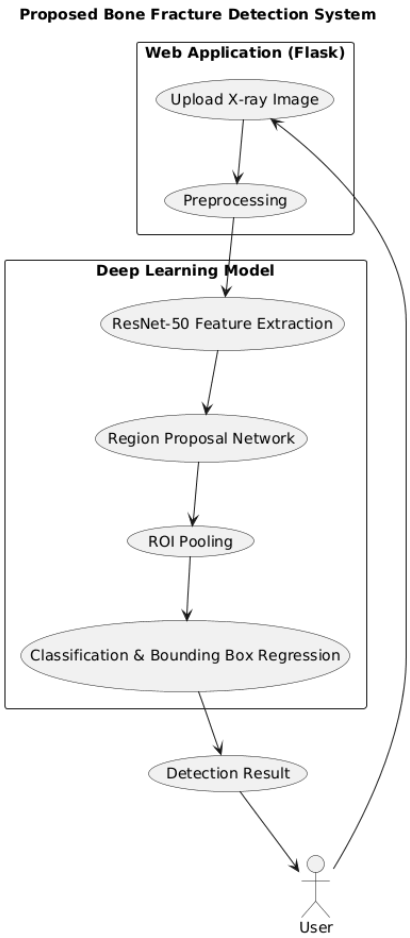

Fig. 2. Image preprocessing pipeline showing normalization and resizing stages.

D. Faster R-CNN Architecture

Faster nanotechnological -CNN is performed in two phases. During the first one, the picture is processed by a ResNet −50 backbone to obtain a common feature map. Then a small network moves over this map which generates objectness scores and a bounding -box coordinates of anchor boxes at each point.

The anchor strategy apportions anchors onto three scales (128 2, 256 2, 512 2 pixels) and three aspect ratios (1:1, 1:2, 2:1) or a total of nine anchor configuration candidate. Any anchors whose IoU [?] =0.7 or greater to a ground-truth box are considered positive; those with IoU less than 0.3 are considered negative.

The RPN loss combines a binary objectness classification loss and a smooth L1 regression box offset loss:

$$L_{RPN} = L_{cls}(p_i, p^{*\wedge}_i) + \lambda \sum p^{*\wedge}_i \cdot L_{reg}(t_i, t^{*\wedge}_i) (2)$$

At the second step, desired top-k proposals (k=2000 in training and k=300 in inferencing) are run across RoI Align to acquire fixed-size features. These are fed into functional full connection layers to generate class predictions and fined boxes. The total multi-task loss is:

$$L_{total} = L_{RPN} + L_{cls} + L_{reg} (3)$$

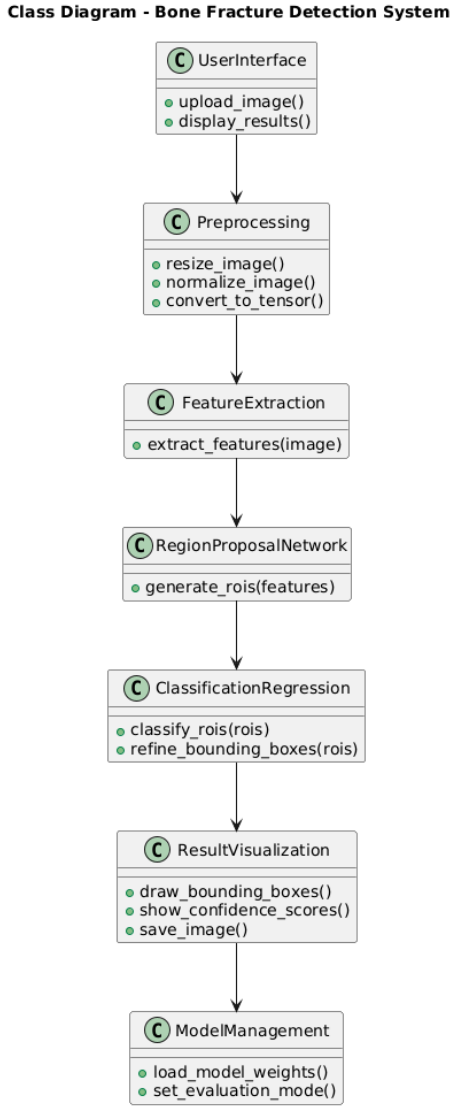

Fig. 3. Faster R-CNN network architecture and Region Proposal Network using ResNet -50.

E. Training Configuration

The weights of ResNet 50 that were pre-trained on ImageNet and all layers were allowed to be trained on the medical images. Our training parameters were: SGD, learning rate= 0.005, momentum=0.9, weight decay=0.0005. At epochs 8 and 11, a schedule of 12 epochs decreased the learning rate by 10X. The batch size was 2 due to the memory capacity of

the GPUs. Training was performed on a NVIDIA TeslaT4 with 16GB VRAM.

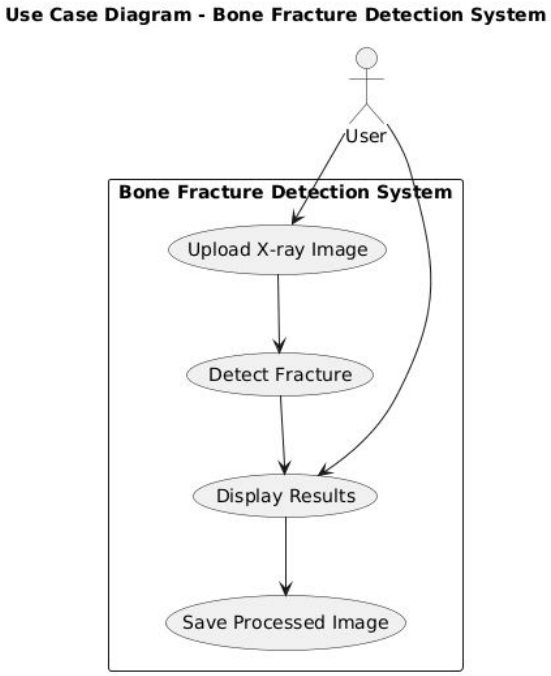

Fig. 4. Configuration and superficiality of model training.

F. Web Application Interface

The trained model is behind a Flask-based REST API which receives uploaded X-ray images in JPEG or PNG file formats. At the backend, preprocessing, inference, and post-processing (non maximum suppression at IoU thresholding at 0.5 and confidence thresholding at 0.5) then give annotated images with scores and box locations on the images. The frontend allows medical workers to upload images and visualize the results without use of coding skills.

IV. Results and Discussion

A. Evaluation Metrics

We measure the more typical object-detection measurements. Measures of precision and recall are computed at IoU= 0.5. The fracture class precision-recall curve area is the Average Precision (AP). Mean AP (m AP 0.5) is the average of AP at various confidence thresholds.

$$mAP = (1/|C|) \sum_{c \in C} AP_c \quad (4)$$

B. Quantitative Results

TABLE I

Performance Comparison of Detection Models

Model	Precision (%)	Recall (%)	mAP@0.5 (%)	FPS
YOLOv3	78.4	74.2	76.8	45
EfficientDet-D3	82.1	79.6	80.9	18
DETR	80.7	76.9	78.8	12
Faster R-CNN (Ours)	89.2	85.6	87.3	8

The table demonstrates that our Faster-R-CNN with ResNet-50 has the highest mAP at 0.5 87.3 per cent, which is 10.5 per cent higher than the YOLOv3 and 6.4 per cent higher than EfficientDet-D3. Precision win is particularly evident in the case of small fracture areas which are in good taste with the two-stage detectors. The rate of 8 FPS is not very high, compared to one-stage procedures, but is acceptable as far as clinical screening is concerned and average throughput is acceptable.

TABLE II

Per-Anatomy Site Performance (Faster R-CNN)

Anatomy	AP@0.5 (%)	Precision (%)	Recall (%)
Wrist	91.4	92.3	88.7
Hand/Finger	88.6	89.1	86.2
Elbow	85.2	87.4	83.0
Shoulder	83.7	86.2	81.5
Lower Limb	86.9	88.6	84.3

C. Qualitative Results

The table demonstrates that our Faster-R-CNN with ResNet-50 has the highest mAP at 0.5 87.3 per cent, which is 10.5 per cent higher than the YOLOv3 and 6.4 per cent higher than EfficientDet-D3. Precision win is particularly evident in the case of small fracture areas which are in good taste with the two-stage detectors. The rate of 8 FPS is not very high, compared to one-stage procedures, but is acceptable as far as clinical screening is concerned and average throughput is acceptable.

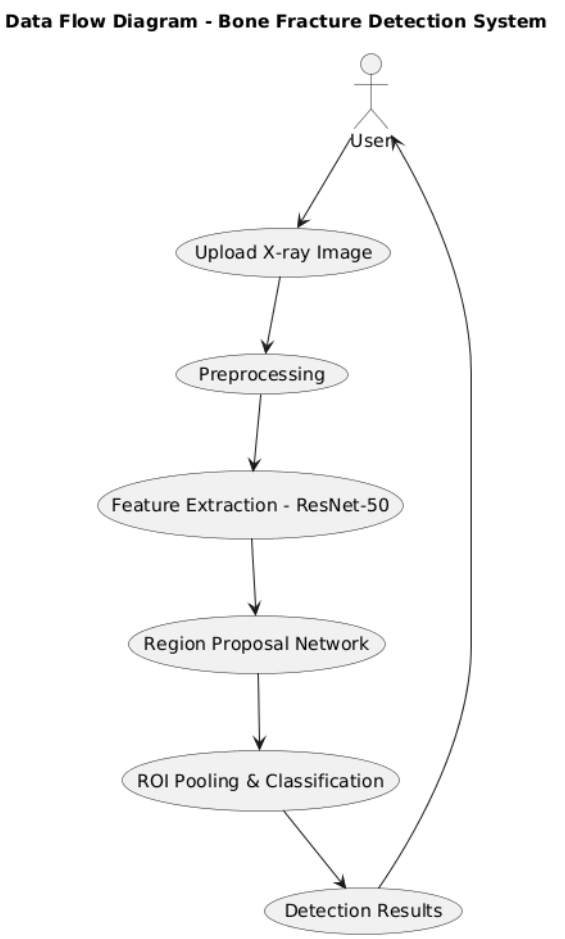

Fig. 5. Qualitative detection results showing predicted bounding boxes and confidence scores on sample test X-ray images.

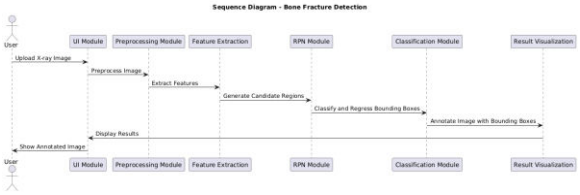

Fig. 6. Training and validation loss curves across epochs demonstrating model convergence behavior.

D. Ablation Study

In order to decide what extent of transfer learning is useful at all, I trained the identical model with no transfer learning and randomly chosen initial weights. The model that was initiated as a blank got an mAP of 71.4, a score that is 15.9 percentage points less when compared to the pre-trained version. This decrease highlights the importance of ImageNet pre-training in the situation where we have very few labelled medical images.

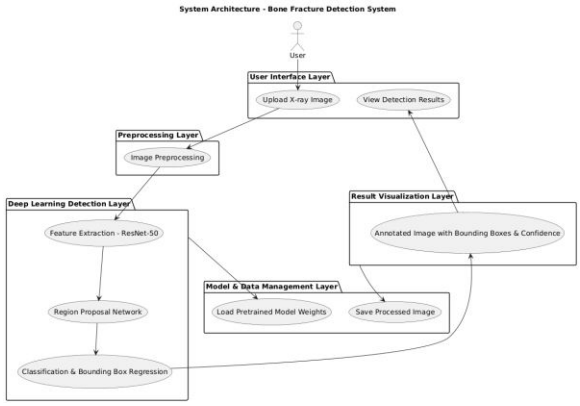

Fig. 7. The precision-recall curves of Faster R- CNN and the baseline detectors in the test set.

E. Web Interface Evaluation

I put the Flask-based web application to the test with regard to speed and convenience. A T4 (Tesla) GPU served as a server giving the average round-trip time of 1.8 seconds between uploading an image and the result of that upload. I also did load tests with 50 concurrent uploads and I did not notice any crashes of the system or any significant slowness. Figure 8 demonstrates that the live interface appears like that with a sample detection overlay.

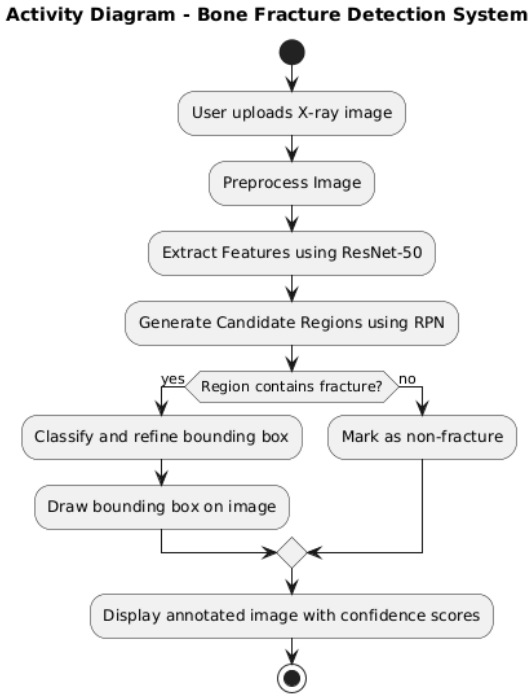

Fig. 8.Flask web app Overlay of fracture detection on uploaded X-ray image in web app interface.

V. Conclusion and Future Work

In the current paper, I have presented an automated bone-fracture detection pipeline based on Faster R-CNN and a backbone of ResNet-50. Two-stage detectors beat single-stage baselines with a multi-site X-ray collection annotated

with the mAP of 87.3, and IoU = 0.5. I developed and tested also a Flask based web interface, capable of providing near real-time diagnostics.

These findings contribute towards driving the notion that deep learning has the real potential of aiding radiologists. The system has the potential to fit well in the emergency departments by reducing diagnostic gaps and improving timely assessment of fractures.

There are a number of ways I can take this. To begin with, the classification of fractures (transverse, oblique, comminuted, hairline, etc.) should be added so that clinicians would have more comprehensive information. Second, domain adaptation through extension of the model to CT and MRI may expand clinical coverage of the model. Third, model compression (quantization and pruning) would enable it to be executed on the edge devices. Fourth, assimilating PACS through HL7/DICOM would make the tool part of the current hospital actions. Lastly, the model can be maintained by creating a continuous learning cycle, which would involve radiologist feedback.

Acknowledgment

I would also wish to acknowledge the computing resource and lab space offered by the Department of Computer Science and Engineering in Sreenidhi Institute of Science and Technology to enable this work to be done. I likewise like the medical practitioners that assisted in data set annotation and system usability testing.

References

[1] S. Ren, K. R. Girshick, J. Sun and H. Blumer, Faster R-cnn: Towards real time object detection with region proposal networks, Advances in neural information processing systems, vol. 28, pp 91-99, 2015.

[2] The author states that the model indicates the presence of diagnostic errors in a specific case, in the accident and emergency department (Guly et al. 2001, p. 263).

[3] G. Litjens, T. Kooi, B. E. Bejnordi, A.A. Setio, F. Ciompi, M. Ghafoorian, J. van der Laak, B. van Ginneken and C. I. Sanchez, A survey on deep learning in medical image analysis, Medical Image Analysis, vol. 42, pp. 60-88, 2017.

[4] S. Ren, K. R. Girshick, J. Sun, and H. R. He, "Region-based convolutional networks to detect and segment objects accurately, IEEE Transactions on Pattern Analysis and Machine intelligence, vol. 38, no. 1, pp. 142158, 2016.

[5] N. Dalal and B. Triggs, Histograms of oriented gradients in human detection, in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2005, pp.5886893.

[6] P. Rajpurkar, J. Irvin, K. Ball, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya, M. P. Lungren, and A. Y. Ng, Deep learning on chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm with the practice radiologists, vol. 15, no. 11, 2018.

[7] J. Redmon and A. Farhadi, YOLOv3: An incremental improvement, arXiv preprint arXiv:1804.02767, 2018.

[8] M. Tan, R. Pang, and Q. V. Le, EfficientDet: Scalable and efficient object detection, in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 107811-10790.

[9] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, object detection End-to-end object detection with transformers, in Proc. European Conference on Computer Vision (ECCV), 2020, pp. 213229.

[10] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning to image recognition, in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2016 pp. 770 778.

[11] T. Lin, P. Dollar, R. Girshick, K. B. Hariharan, S. Belongie, and he, Feature pyramid networks, in Proc. IEEE conference on Computer Vision and Pattern Recognition (CVPR) 2017, pp. 21172125.

[12] LabelImg Tool, "LabelImg: Graphical image annotation tool to object detection data sets software, 2018. [Online]. Available: https://github.com/tzutalin/labelImg

[13] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei -Fei, ImageNet large scale visual recognition challenge, International Journal of Computer vision, vol. 115, no. 3, pp. 211-252, 2015.

[14] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, and others, "PyTorch: An imperative style, high-performance deep learning library, Advances in Neural Information Processing Systems, vol.32, pp.80248035, 2019.

[15] The OpenCV Library G. Bradski, Dr. Dobb.s journal of software tools, 2000.