ISSN 2224-087X. Електроніка та інформаційні технології. 2024. Випуск 28. С. 14–21
Electronics and information technologies. 2024. Issue 28. P. 14–21

УДК 004.93 DOI: https://doi.org/10.30970/eli.28.2

ANALYSIS OF EFFECTIVE IMAGE PROCESSING METRICS ON RASPBERRY PI AND NVIDIA JETSON NANO

R. Mysiuk

Ivan Franko National University of Lviv,
50 Drahomanova St., UA-79005 Lviv, Ukraine
mysyukr@ukr.net

Image processing and object recognition technologies for detecting defects on the surfaces of a variety of materials is critical to ensuring the safety and durability of infrastructure. That is why this topic and problems of the field of defect detection were chosen in the study. Therefore, the development of effective image processing methods for defect identification, especially in low-light conditions and hard-to-reach places, is of great relevance. The research is a comparison of classical image processing methods with modern deep learning algorithms such as CNN (Convolutional Neural Networks) and YOLO (You Only Look Once). The study analyzes the effectiveness of these methods under specific defect inspection conditions, including diffuse lighting and device mobility. An important aspect is the use of microcomputers such as Raspberry Pi and Nvidia Jetson Nano, which ensures the mobility and autonomy of the system. The practical value of the research lies in the implementation of effective image processing methods for detecting defects on the surfaces of engineering structures. This makes it possible to significantly improve the accuracy of surface defect identification, which is confirmed by the IoU (Intersection over Union) and Dice metrics. In particular, using CNNs for surface defect identification showed 35% better results compared to existing implementations of similar networks and 12% more efficient compared to YOLO. On the other hand, YOLO proved to be more productive in terms of processing frames per second on microcomputers, which is important for real-time monitoring.

Keywords: computer vision, deep learning, image processing, surface defect detection, Raspberry PI, Nvidia Jetson Nano, performance metrics.

Overview.

The quality and accuracy of the analysis of objects in the images depends on a number of different conditions and features of the studied objects. For example, when processing images to monitor the condition of surface defects, diffuse illumination is an important aspect when processing images in dark and hard-to-reach areas. Therefore, taking into account these features, the efficiency analysis is carried out using single-board computers to ensure the mobility of devices with an attached camera.

The complexity of developing and improving image processing methods in the field of flaw detection lies in the variety of defects, materials, and lighting conditions. Furthermore, it is essential to consider the large amount of training data and the selection of optimal algorithms for evaluating efficiency and reliability.

Object recognition is performed based on various algorithms and data analysis models. In general, these methods can be divided into two main categories: classical image processing

methods and the latest deep learning methods that apply neural networks. A combination of both groups of image processing methods is often used to solve defects analysis problems.

The classic methods include the following options [1]:

- Image filtering to highlight distinctive features with consideration of blurring to remove noise.
- Morphological operations for highlighting the shape of objects.
- Algorithms for identifying contours of objects for dividing regions in images.
- Thresholding of pixel values above or below a given threshold to highlight objects by certain colors.

Research results obtained in works [2–6] showed intensive use of modern deep learning methods: Convolutional Neural Networks (CNN), YOLO (You Only Look Once), Region-based CNN (R-CNN), Mask R-CNN, and Generative Adversarial Networks (GAN). They have different architectures and purposes in the field of image processing and computer vision. The main difference between them:

- CNNs are used to classify and extract features in images.
- YOLO provides fast detection of objects in real time.
- R-CNN identifies objects and their boundaries with high accuracy, but at a lower speed.
- Mask R-CNN adds object segmentation capability to R-CNN.
- GANs generate new, synthetic, real-world-like data, improving training datasets.

Taking into account the above-described features of the application of such algorithms, CNN and YOLO models have been implemented to compare the efficiency and accuracy in the tasks of recognizing defects on the surfaces of the investigated objects on the image.

During the implementation of CNN in the recognition tasks, UNet deep learning architecture is used for image segmentation. Such an architecture is particularly effective for tasks where it is necessary to determine the boundaries and shape of elements in the defect recognition image. A feature of this architecture is that the structure is divided into two parts: encoder and decoder. The first stage serves to reduce the size of the image with a certain amount of information, the final one - increases the size while restoring the original information. At the same time, segmentation of the image of surface defects of objects takes place based on the used series of convolutional and connecting layers. The structure of the neural network and its application features are described, the results of defect identification are considered in previous works [7–9]. However, the selection of model parameters, the number of neural network layers, and other preparatory stages should be performed experimentally for each data set.

After that, it is worth evaluating the effectiveness of image processing using machine learning methods on defect images using quality metrics and data selection. Meanwhile, the optimal values of the model parameters can be obtained experimentally based on the number of layers, image size, number of filters for each layer, activation function, estimation of the loss function, the result of data augmentation, as well as hyperparameter settings.

The first stage of implementation is the preparation of data for training. All images are augmented by applying random rotations of +/- 5% and then split into training, validation, and test datasets of 70, 20, and 10%, respectively. Images are resized according to convolutional neural network (CNN) architecture requirements.

Among the existing software implementations [10] of defect segmentation with CNN, a dataset was used for training. Therefore, the work used 11200 images with a standardized size of 448 by 448.

The datasets include images of crack-type defects with noise (dirt, seams, and other objects) on the surfaces. For comparison, the existing data set was supplemented with images of defects such as cracks on cylindrical objects with augmentation associated with image rotations.

In each image, the expected polygonal areas (annotation of images) of defects on the surfaces in the Roboflow environment are selected to calculate the accuracy [11].

A CNN model was created based on the UNet architecture, trained using the PyTorch library and converted to the ONNX format. Considering the volume of the model, the ONNX Runtime and OpenCV libraries were used, which help to perform a faster assessment of damage to the surfaces of engineering structures in real time [12].

Visual Geometry Group 16 (VGG16) is used as a deep learning architecture. The structure of this network: 16 layers, consisting of 13 convolutional and 3 fully connected layers. VGG16 is simpler in terms of architecture [10], compared to ResNet.

The main steps of the operational scheme of the study are shown in Fig. 1, including the collection and pre-processing of data on engineering objects with crack-type defects. Then, two neural network architectures are chosen - CNN and YOLO, which are trained on more powerful computers with better graphics processor characteristics and on extended data sets. After that, the performance of the models is evaluated using the accuracy segmentation metric, and the frame-per-second processing efficiency is analyzed on Raspberry Pi and Nvidia Jetson Nano single-board computers.

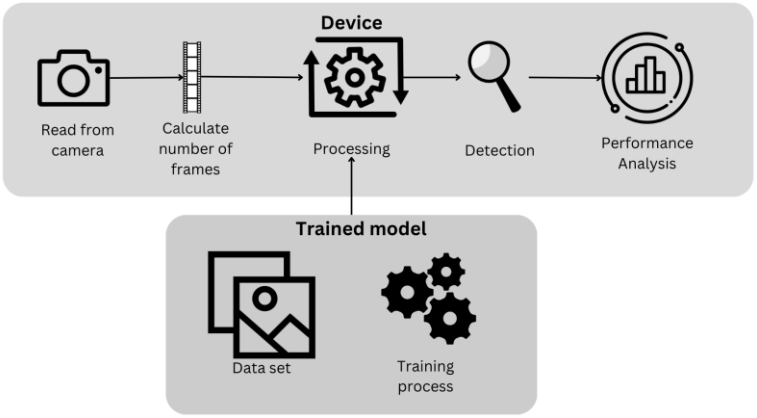

Fig. 1. Scheme of operation of the main steps

Another implementation of deep learning using the YOLO library was performed to classify objects in real time. Based on the grid, areas and confidence of similarity of objects are determined. Python version 3.10.12 was chosen as the main programming language because it works well enough with libraries designed for machine learning. The YOLO model was trained using Ultralytics YOLOv8.0.196 [13] and PyTorch version 2.1.0 [14] with Compute Unified Device Architecture (CUDA) version 12.2 support. For comparison, the Google Colab Research environment was used for the corresponding implementation on the default free NVIDIA Tesla T4 graphics accelerator with 15 GB of memory.

After this, the training time analysis was performed, which showed that the training process took approximately 1 minute for 25 epochs for a model with 168 layers. The total processing time of one image consists of the time of preprocessing (on average 3.5 ms), determining the location of defects (on average 18.2 ms) and adding visual labels to the image (on average 28.3 ms).

After training, the model is ready for use on single-board devices, where its effectiveness has been evaluated. As for platforms, there are ready-made robotic platforms JetRacer AI Kit and PiRacer Pro AI Kit, which allow recognition of objects in motion based on running trained models.

Performance evaluation for detection.

The characteristics of single-board computers affect the efficiency and effectiveness of identifying defects on the surfaces of engineering structures. Due to limited resources (RAM, processor power and storage space), it is important to ensure the speed of processing a large number of video frames.

One of the indicators for evaluating the efficiency (speed) of the method on single-board devices is the calculation of the processing time of the number of frames per second (FPS).

$$FPS = \frac{C}{T_{s2} - T_{S1}}$$

where T_{S1} - start time, T_{S2}- finish time, C - number of frames per video time.

When processing high-resolution images on low-power devices, there may be some delays. Neural network model optimization can be done with the TensorRT engine [15], which is supported in Nvidia Jetson Nano and is used to work with trained models based on Caffe, TensorFlow, PyTorch and ONNX. This engine allows memory and GPU bandwidth optimization based on layer data and tensor fusion processes. Thus, it reduces the overhead of reading and writing the tensor data for each layer. Such optimization based on TensorRT allows identifying defects on surfaces ten times faster and with a shorter response time. The created model is converted to ONNX format using PyTorch. After that, it is converted to a file with the model.trt extension in tensorrt. A small amount of processing FPS may be due to the complexity of the models.

YOLO models are heavy enough, so to perform image processing with object identification on Jetson Nano, it is needs to be investigated in further works with simplified models, for example, different versions from tiny as it is described in [16]. This is due to the fact that images with a high resolution reduce the efficiency of processing FPS. Therefore, you need to resize the image in the datasets to a smaller size.

Based on the results of measurements of the load on the graphic processor during the corresponding calculations for the purpose of identifying objects in the image, models with smaller images can be considered more effective, but the accuracy of identification is not high enough. And measuring the efficiency of image processing on Nvidia Jetson Nano with different resolutions showed the effect of image size on the results of calculating the number of FPS. Therefore, all images in the datasets are standardized to an average size of 448 by 448. Processing of video fragments of different lengths was performed using CNN [10], CNN with the ONNX Runtime library, and YOLO on Raspberry Pi and Nvidia Jetson Nano microcomputers. During the processing of each video frame, masks (red highlights) with identified defects are superimposed on the input image. Processing images and displaying this process takes some

time, which causes a delay (Table 1), which may depend on the complexity of the neural network and the capabilities of technical means.

Table 1. Processing time of a video fragment on pre-trained models

ML methods	Average number of FPS based on video fragments of different lengths	
	Nvidia Jetson Nano (2 GB)	Raspberry Pi 4 (4 GB)
CNN [10]	24	21
CNN з ONNX Runtime	27	24
YOLO	29	26

As mentioned earlier, the training of deep neural networks was performed in the Google Colab environment. This choice of training environments is related to the limited computing resources of single-board devices. The resulting model file (pre-trained model) is loaded onto the device for its execution. Thus, the conducted distributed training allows to speed up the training time and take into account the limited characteristics of devices such as CPU or GPU, the amount of RAM.

Therefore, the processing delay can be affected by large video sizes and high bitrates, the amount of neural network calculations (reduction of encoders and decoders), too many objects to track, the presence of artifacts or connection problems. However, with the optimization of the neural network, changes in the accuracy of identification are possible.

In the Table 1 results may be due to GPU and video processing optimizations, hardware architecture characteristics of Nvidia Jetson Nano (2 GB) and Raspberry Pi 4 (4 GB). The advantage of Nvidia Jetson Nano is the NVIDIA Maxwell graphics processor, which supports CUDA and accelerates the processing of large amounts of data. In contrast, the Raspberry Pi 4 does not have this level of graphics acceleration and optimizations.

Evaluation of the accuracy of defect detection.

Intersection over Union (IoU) and Dice are metrics for evaluating the accuracy of segmentation (delineation) of regions in images. In Table 2 these performance parameters are used to evaluate the accuracy of the models.

IoU of the union area (the area containing both the original and the projected area) to the area of intersection of these areas and is calculated

$$IoU = \frac{TP}{TP + FP + FN}$$

where TP (True Positive) is the number of pixels that the model correctly identified as an object region; FP (False Positive) - the number of pixels that the model mistakenly identified as an object region; FN (False Negative) - the number of pixels that the model missed.

In the context of image segmentation, the Dice coefficient is also used as a measure of similarity between the predicted and true masks and is calculated by the formula

$$Dice = \frac{2 \times TP}{2 \times TP + FP + FN}$$

Table 2. Indicators of segmentation quality during defect identification

Metric	CNN [10]	Advanced CNN with cylindrical objects in images	YOLO
Intersection of Union	0.47	0.63	0.56
Dice	0.6	0.72	0.68

Fig. 2. Examples of image-based defect identification with/without cracks inside a part of a pipe

As shown in Fig. 2, the input and result of image processing are shown with the corroded part of the pipe without cracks and with cracks. Usually, the data processing method is usually chosen based on the amount and quality of data available. The increase in volume and improvement in the quality of data sets has contributed to the improvement of the identification of defects, in particular cracks, in cylindrical objects such as pipelines. This allows more accurate and reliable detection of problem areas and prevention of possible accidents.

Conclusion.

The study compared the efficiency and accuracy of CNN and YOLO neural network architectures for various engineering objects. The results of the analysis were used for an extended set of training and test data. Metrics for evaluating the quality of segmentation procedures, such as IoU and Dice, have shown that CNN provides better results when identifying defects. In particular, the performance of CNN was found to be 35% higher compared to an existing implementation of a similar CNN network and 12% better compared to a YOLO-based implementation.

On the other hand, when analyzing the number of frames per second processed, it is found that YOLO is more effective for identifying defects on single-board computers such as Raspberry Pi and Nvidia Jetson Nano. In particular, the efficiency of YOLO on these devices is 20.2% and 20.7%, respectively. These results can be explained by the peculiarities of the implementation of the techniques: CNN has a greater depth of the neural network, which allows

better recognition of complex structures, while the simpler architecture of YOLO provides higher image processing speed, but lower segmentation accuracy.

Also, the use of TensorRT and ONNX Runtime mechanisms allows you to speed up the process of identifying defects on microcomputers. In addition, the features of defect identification, such as corrosion processes and cracks in structural elements, were considered and analyzed in a number of well-known scientific works. These studies emphasize the importance of the relationship between the results of defect diagnosis and the methods of transmission of processed images.

References

[1] *S. Vani, H. Singh*. The Performance Analysis of Edge Detection Algorithms for Image Processing Based on Improved Canny Operator. *International Journal of Computer Trends & Technology*. 2020. Vol. 68, No. 10. P. 29–34. https://doi.org/10.14445/22312803/IJCTT-V68I10P106

[2] *A. Saberironaghi, J. Ren, M. El-Gindy*. Defect Detection Methods for Industrial Products Using Deep Learning Techniques: A Review. *Algorithms*. 2023. Vol.16. P. 95. https://doi.org/10.3390/a16020095

[3] *Z. Ren et al*. State of the Art in Defect Detection Based on Machine Vision. *International Journal of Precision Engineering and Manufacturing-Green Technology*. 2022. Vol. 9, P. 661–691. https://doi.org/10.1007/s40684-021-00343-6

[4] *J. Lu, S.-H. Lee*. Real-Time Defect Detection Model in Industrial Environment Based on Lightweight Deep Learning Network. *Electronics*. 2023. Vol. 12. No. 21. P.4388. https://doi.org/10.3390/electronics12214388

[5] *Zhen Yu*. YOLO V5s-based Deep Learning Approach for Concrete Cracks Detection. *SHS Web Conf.* 2022. Vol. 144. P. 03015. https://doi.org/10.1051/shsconf/202214403015.

[6] *Riya Kumari et al*. A Review of Image Detection, Recognition and Classification with the Help of Machine Learning and Artificial Intelligence. *SSRN Electronic Journal*. 2020. https://doi.org/10.2139/ssrn.3611339

[7] *R. Mysiuk et al*. Detection of Surface Defects Inside Concrete Pipelines Using Trained Model on JetRacer Kit. *2023 IEEE 13th International Conference on Electronics and Information Technologies*, 2023. P. 21–24. DOI: 10.1109/ELIT61488.2023.10310691

[8] *R. Mysiuk et al*. Video-based Concrete Road Damage Assessment Using JetRacer Kit. *2023 17th International Conference on the Experience of Designing and Application of CAD Systems*. 2023, P. 1–4, doi: http://doi.org/10.1109/CADSM58174.2023.10076528.

[9] *R. Mysiuk, V. Yuzevych*. IoT-based solution for detection defects in infrastructure objects using Raspberry PI. *Electronics and Information Technologies*. 2023. No. 21. P. 45–56. http://doi.org/10.30970/eli.21.5

[10] GitHub - khanhha/crack_segmentation: This repository contains code and dataset for the task crack segmentation using two architectures UNet_VGG16, UNet_Resnet and DenseNet-Tiramusu. GitHub. URL: https://github.com/khanhha/crack_segmentation

[11] Roboflow: Give your software the power to see objects in images and video. Roboflow: Give your software the power to see objects in images and video. URL: https://roboflow.com/

[12] ONNX Runtime. Inference. ONNX Runtime. Home. URL: https://onnxruntime.ai/inference

[13] Ultralytics. Home. Home – Ultralytics YOLOv8 Docs. URL: https://docs.ultralytics.com/

[14] PyTorch v2.1.0. Previous PyTorch Versions. URL: https://pytorch.org/get-started/previous-versions/#v210
[15] A Guide to using TensorRT on the Nvidia Jetson Nano. Donkey Car. URL: https://docs.donkeycar.com/guide/robot_sbc/tensorrt_jetson_nano/
[16] *D. Adami, M. O. Ojo and S. Giordano.* Design, Development and Evaluation of an Intelligent Animal Repelling System for Crop Protection Based on Embedded Edge-AI. in IEEE Access. 2021. Vol. 9, P. 132125–132139. DOI: 10.1109/ACCESS.2021.3114503

АНАЛІЗ ПОКАЗНИКІВ ЕФЕКТИВНОЇ ОБРОБКИ ЗОБРАЖЕНЬ НА RASPBERRY PI ТА NVIDIA JETSON NANO

Роман Мисюк

Львівський національний університет імені Івана Франка,
вул.Драгоманова, 50, м.Львів, 79005, Україна
mysyukr@ukr.net

У сучасному світі технології обробки зображень та розпізнавання об'єктів відіграють важливу роль у багатьох галузях, зокрема у дефектоскопії інженерних конструкцій. Виявлення дефектів на поверхнях різноманітних матеріалів є критично важливим для забезпечення безпеки та довговічності інфраструктури. Тому удосконалення швидкодії методів обробки зображень для ідентифікації дефектів, особливо в умовах слабкого освітлення та важкодоступних місць, має велику актуальність. Дослідження полягає у порівнянні показників ефективності та точності сегментації класичних методів обробки зображень з сучасними алгоритмами глибокого навчання, такими як CNN (Convolutional Neural Networks) та YOLO (You Only Look Once). Для порівняння ефективності здійснено аналіз цих показників з наявних прикладів реалізації та створеними моделями, які містить збільшений наборів зображень дефектів у циліндричних об'єктах, виконану аугментацію та повторному навчанню у середовищі Google Colab Research.

Дослідження аналізує ефективність цих методів з урахуванням специфічних умов дефектоскопії на одноплатних пристроях. Оскільки такі пристрої мають обмежені обчислювальні характеристики, то варто використовувати оптимізацію моделі існуючими інструментами для покращення ефективності. Для цієї задачі обрано Raspberry Pi з обємом оперативної пам’яті 4 Гб та Nvidia Jetson Nano - 2 Гб, що забезпечує мобільність та автономність системи ідентифікації дефектів на поверхнях. Практична цінність дослідження полягає у впровадженні ефективних методів обробки зображень для виявлення дефектів на поверхнях інженерних конструкцій. Це дозволяє суттєво покращити точність ідентифікації дефектів, що підтверджено метриками IoU (Intersection over Union) та Dice. Інтегровані графічні прискорювачі TensorRT у Nvidia Jetson Nano дозволяє обробити більшу кадрів в секунду, у порівнянні з Raspberry Pi.

Зокрема, використання CNN для ідентифікації дефектів показало на 35% кращі результати порівняно з існуючими реалізаціями схожих мереж та на 12% ефективніше порівняно з YOLO. З іншого боку, YOLO виявилася більш продуктивною з точки зору обробки кадрів в секунду на мікрокомп'ютерах, що є важливим для реального часу моніторингу.

Ключові слова: комп'ютерний зір, машинне навчання, обробка зображень, виявлення поверхневих дефектів, Raspberry PI, Nvidia Jetson Nano, показники ефективності.

The article was received by the editorial office on 18.10.2024.
Accepted for publication on 01.11.2024.