Image Detection Method Based on CNN and Pre-Trained Models

Oleg Pursky[1], Tetyana Filimonova[1], Anna Selivanova[1], Tatiana Dubovyk[1], Victor Kovtunenko[2] and Iryna Buchatska[1]

[1]*State University of Trade and Economics, Kioto Str. 19, 02156 Kyiv, Ukraine*
[2]*Cherkasy State Technological University, Shevchenka Blvd. 460, 18006 Cherkasy, Ukraine*
pursky_o@ukr.net, t.filimonova@knute.edu.ua, a.selivanova@knute.edu.ua, t.dubovyk@knute.edu.ua, v.kovtunenko@chdtu.edu.ua, i.buchatska@knute.edu.ua

Keywords: Image Detection, CNN, Transfer Learning, Deep Learning, Computer Vision, Neural Networks.

Abstract: This work presents a comprehensive study of the image detection method using CNN and pre-trained models. An object detection method has been developed based on a combination of modern convolutional neural network (CNN) architectures and a transfer learning strategy. Special attention in the developed method is paid to the algorithm for geometric transformation of the input image, which performs scaling while preserving the proportions (aspect ratio) and adding the minimum necessary fields (padding). To improve the interpretation of the neural network output data, a visualization module has been developed that imposes bounding boxes and class labels on the output image. The proposed approach uses the weights of pre-trained models. The results demonstrate an increase in network convergence speed and detection accuracy compared to training models "from scratch", implemented in Python using the TensorFlow/Keras, OpenCV, and NumPy libraries. The practical value of the work lies in creating a full cycle (pipeline) of developing a detection system: from data preparation to a ready-made inference module. The developed toolkit is universal and can be quickly adapted to solve applied tasks (video surveillance, quality control, traffic monitoring) by replacing the dataset without changing the software architecture.

1 INTRODUCTION

Computer vision and image detection technologies play a key role in many areas, such as robotics, video surveillance, medicine, and unmanned transportation. The use of convolutional neural networks (CNN) is the de facto standard for solving these problems, but training such networks "from scratch" requires enormous computational resources. The use of pre-trained models allows you to significantly increase the efficiency of development, reduce training time, and achieve high accuracy even on limited data sets. The development of detection methods based on this approach is relevant, as it provides a balance between speed and accuracy, which is critical for real-time systems and devices with limited computing power.

The aim of the research is to develop an image detection method using CNN and pre-trained models to create an effective object recognition system, optimize the training process, and increase detection accuracy by adapting existing architectures.

The work uses Deep Learning methods to build and configure convolutional neural networks, Transfer Learning to adapt pre-trained models, digital image processing methods to prepare input data, and statistical analysis methods to assess the metrics of model performance (accuracy, completeness, mAP).

The novelty of the results lies in the study of the effectiveness of using different strategies for adapting pre-trained CNN models to solve the problem of object detection in specific conditions. A comparative analysis of architectures was conducted and optimal pre-training parameters were determined, which allow achieving high accuracy of object classification and localization while minimizing computational costs compared to training networks without prior initialization of weights. The developed methodology and software implementation can be used to create applied computer vision systems where fast and accurate object detection is required.

2 LITERATURE REVIEW

A significant number of works are devoted to the study of computer vision and pattern recognition

issues [1]-[11]. The principles of the modern paradigm of machine vision, according to which high detection efficiency can be achieved by using convolutional neural networks (CNN) and ensuring hierarchical feature extraction, were substantiated by J. Lecoun [12].

J. Redmon [11] improved the detection methodology by proposing the "You Only Look Once" (YOLO) approach, which considers detection as a single regression problem, significantly accelerating the processing process.

In the studies of K. He [10], the prospects for the development of deep networks based on residual learning (ResNet), which allowed solving the problem of vanishing gradient when training ultra-deep models.

It should be noted that when considering the functions of computer vision systems, the authors of modern reviews [13] focus on the creation of complex intelligent systems, that is, they analyze the process of processing visual information as the formation of multi-level abstractions.

The scientific and methodological principles of using pre-trained models (Transfer Learning), which, taking into account the peculiarities of learning on limited samples, characterize the possibility of transferring knowledge between domains, are discussed in depth in the scientific works of S. Rader [14]. In [15], the theoretical and methodological principles of adapting CNN architectures for specific detection tasks are defined.

Currently, the main goal of the development of computer vision technologies is to bring the capabilities of artificial systems closer to human perception. It is no coincidence that in the strategies for the development of the digital economy of many countries tasks are defined that are aimed at creating autonomous navigation, monitoring and situation analysis systems, and ultimately - at increasing the safety and efficiency of human activity. The implementation of the tasks is facilitated by using as the goals of the development of the detection method, first of all, such as increasing the accuracy of recognition in difficult weather conditions, reducing the number of false positives, optimizing work in conditions of partial overlap of objects.

In order to study and understand the essence of visual images in computer vision, various interdependent specific research methods are used, the totality of which forms the methodology of digital image processing (Digital Image Processing) [16].

There are several approaches to building effective object detectors [17], [18]. One of them (Two-Stage Detectors, e.g., R-CNN) consists in generating candidate regions (Region Proposals) that potentially contain objects, with the subsequent classification of each region. The disadvantage of this approach is the division of the process into stages and, as a result, the low speed of operation, which makes it impossible to use in real time. Another approach (One-Stage Detectors, e.g., YOLO, SSD) consists in simultaneously predicting the coordinates of frames and object classes through a single neural network. A variation of this approach is the use of anchor frames (Anchors), relative to which displacements are predicted. In this case, the values of the network outputs correspond to the coordinates and probabilities in the image grid. The disadvantages of early versions of this approach include difficulties with detecting small objects, as well as the fact that among the large number of predicted frames, many are false (the problem of imbalance of background and object classes).

Fine-tuning mechanisms of pre-trained models depend on the nature of the new data, the similarity of domains, the required accuracy and speed, and both full unfreezing of weights and training only the last layers can be used. To assess the quality of the detection method, it is necessary to determine its validation metrics - the degree of reliability of the obtained predictions. The basis of the theory of assessing the quality of models is the analysis of Precision-Recall curves [19]. Also, quite often in modern research, data augmentation methods are used. Augmentation is usually carried out by geometric transformations (rotations, scaling) or changing color characteristics.

To determine the quality of detection in situations that are difficult to characterize with a single number (for example, visual quality of segmentation, work in fog or rain), methods of visual inspection of results on test video sequences are used [20]. This approach, although subjective and does not provide an accurate mathematical estimate, allows you to identify specific model errors that may be critical for practical application.

3 METHODOLOGY

This study uses the PASCAL Visual Object Classes (VOC) 2012 dataset [21]. The dataset contains a variety of natural images with many objects of different classes, differing in scale, position, illumination, and degree of occlusion. The dataset has been modified to ensure compatibility with the YOLO family of models. For this purpose, the original annotations in XML format (VOC) were

converted, and the object markup was converted to YOLO format using official scripts provided by the Ultralytics library. As a result, each image corresponds to a text file with bounding box coordinates and class identifiers normalized to the image size. The dataset contains 23.1 thousand files, of which 11.5 thousand are images in **.jpg* format, 11.5 thousand are annotation files in **.txt* format, and one configuration file in **.yaml*. The dataset is distributed under the Creative Commons Attribution-ShareAlike 4.0 (CC BY-SA 4.0) license.

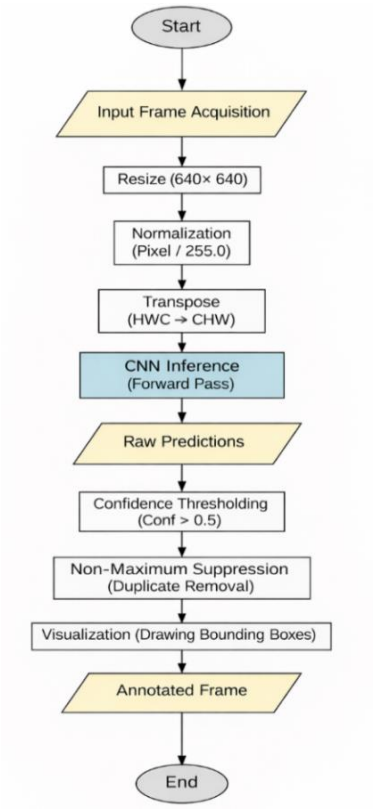

Figure 1: General structure of the algorithm for processing one frame of a video stream or a static image.

A general algorithm for the detection system has been developed. The logic of the system is decomposed into three sequential stages: preprocessing of input data (Preprocessing), inference of the neural network (Inference) and post-analysis of the results (Post-processing). At the Input Frame Acquisition stage, the input image or video frame is read, which is then transmitted to the preprocessing block.

Special attention in the developed method is paid to the algorithm for geometric transformation of the input image (Fig. 1). Most modern CNNs, including YOLO, have a fixed input tensor size (for example, 640×640). Simple bilinear scaling (resize) of images with an aspect ratio other than 1:1 (for example, 1920×1080) leads to geometric distortions of objects, which negatively affects the accuracy of detection. To solve this problem, the Letterbox Resizing algorithm is used, which performs scaling while preserving the proportions (aspect ratio) and adding the minimum necessary margins (padding).

The software complex was developed in Python 3.10 using the Google Colab cloud computing environment, which provides access to graphics accelerators (NVIDIA Tesla T4 GPU). The dataset split includes 5,717 training images and 5,823 validation images, resulting in 11,540 images in total, which corresponds to approximately 11.5 thousand images mentioned above. These numbers refer to images, not to annotated objects. The class distribution is characterized by the expected dominance of the person class, as well as a high representation of the dog, chair, and car classes. The class diagram is presented in Figure 2.

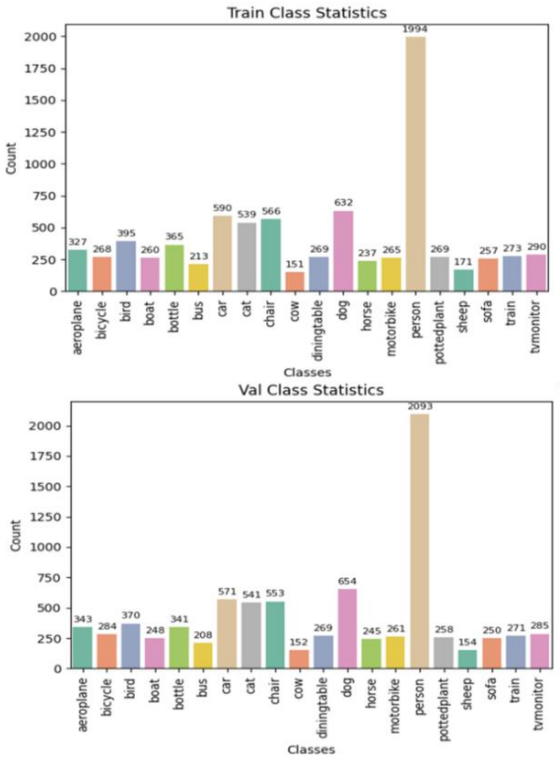

Figure 2: Class diagram of the PASCAL VOC 2012 dataset.

To start training, a *data.yaml* configuration file was programmatically generated, containing all the necessary information about the data set and the training process. The process of initialization and starting training was implemented through the

Ultralytics library API. Note that the training was carried out on the full set of training data, the number of classes used for training was 20. YOLOv8n with pre-trained weights was used as the basic architecture. The model was trained for 50 epochs.

The image before being fed to the model was read using OpenCV and scaled to a fixed size of 640×640 pixels. After resizing, its basic properties after resizing are determined, including width, height, number of channels (1 for grayscale or 3 for RGB), and pixel data type.

After training, the model was exported to ONNX format, which ensures compatibility with various deployment environments and the possibility of further use in autonomous inference systems.

To test the system on arbitrary images (Inference), the *show_prediction* function was developed. Thus, the software implementation represents a full cycle of ML system development: from data preparation and format conversion to model training and creation of tools for user testing. All intermediate results and weights of the trained model are automatically stored in the Google Colab cloud storage.

4 EXPERIMENTS AND RESULTS

A series of experiments were conducted to verify the effectiveness of the developed method. The dynamics of minimizing the loss function (Loss) and increasing accuracy (mAP) are presented in Figure 3. Based on the analysis of the learning dynamics, it was found that the model demonstrates stable and balanced convergence. The values of localization losses (box loss), classification (cls loss) and probability distribution (DFL loss) consistently decrease both on the training and validation samples. At the same time, there is an increase in the precision, recall indicators, as well as the mAP@0.5 and mAP@0.5:0.95 metrics, which confirms the gradual improvement of the detection quality throughout all training epochs. Quantitative indicators of the model quality were obtained by evaluation on the PASCAL VOC validation subset. The final values are given in Table 1.

As a result of testing the developed system, the mAP@0.5 value was 0.7152, while the mAP@0.5:0.95 value was 0.5210. The Precision value of 0.7563 is higher than the Recall value of 0.6458, which indicates that the model tends to reduce the number of false-positive detections. At the same time, the lower Recall value shows that some objects may still be missed in complex scenes. The lower value of mAP@0.5:0.95 compared with mAP@0.5 indicates that detection quality decreases under stricter IoU thresholds. Therefore, the obtained results should be interpreted as a practical trade-off between detection accuracy and real-time inference speed for a lightweight YOLOv8n-based model, rather than as state-of-the-art performance.

Table 1: Summary model performance metrics.

Metrics	Value	Description
Precision	0.7563	Detection accuracy (correct detection rate)
Recall	0.6458	Completeness (fraction of objects found)
mAP@0.5	0.7152	Average accuracy at 50% overlap threshold
mAP@0.5:0.95	0.5210	Integral accuracy metric (COCO metric)

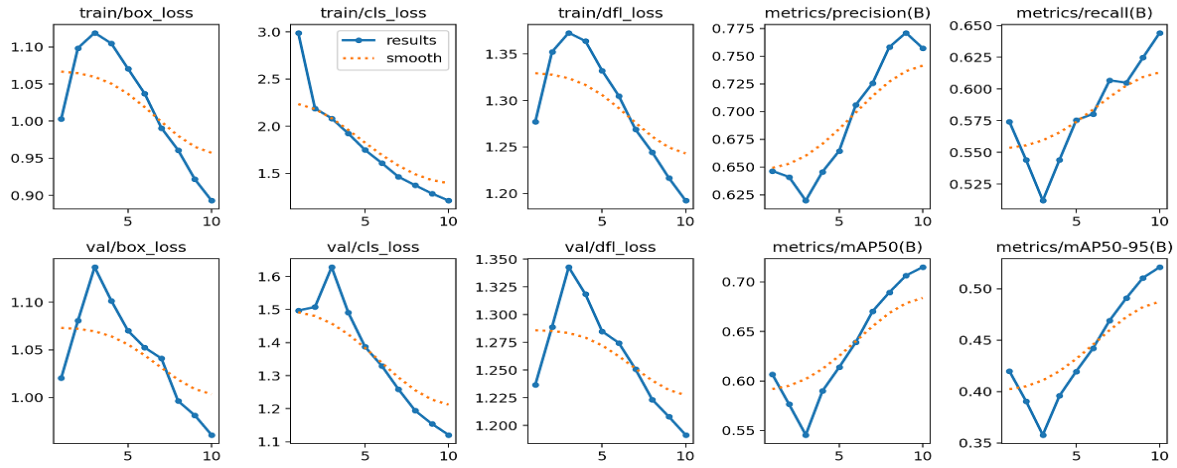

Figure 3: Graphs of changes in metrics (Precision, Recall, mAP) and loss functions during training.

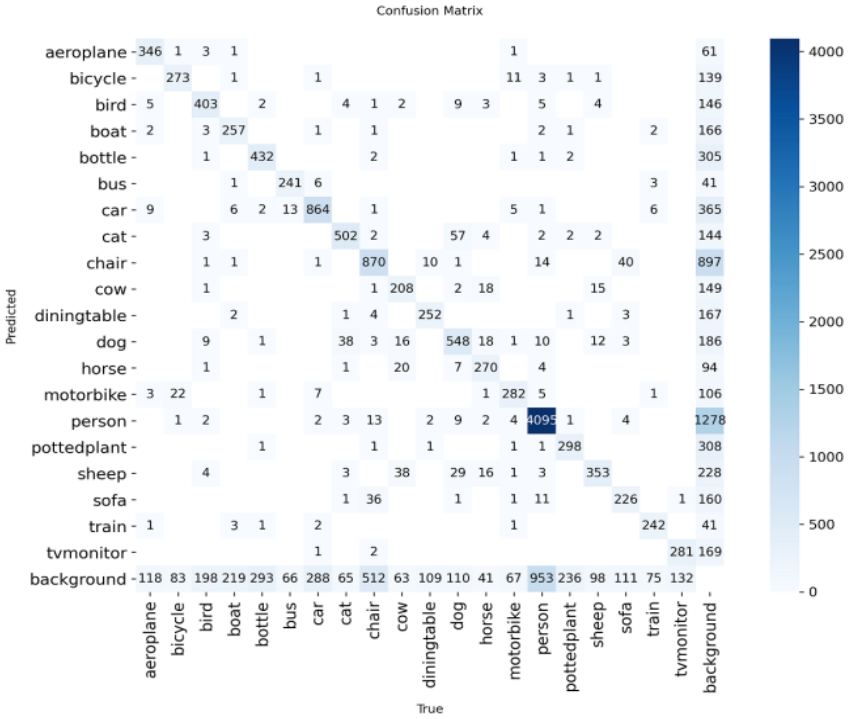

Figure 4: Confusion matrix of detection results on the validation sample.

For a detailed analysis of classification errors, a confusion matrix was constructed, shown in Figure 4.

Inference time was measured for different input image resolutions. The performance analysis was performed on the hardware platform of the Google Colab cloud service (GPU NVIDIA Tesla T4).

Table 2: Dependence of working speed on image size.

Input image size (px)	Inference time (ms)	FPS (frames/ sec)	Note
320 x 320	6.84	146	High speed, reduced accuracy
640 x 640	10.43	96	Optimal balance
1280 x 1280	18.83	53	High precision, low speed

As can be seen from Table 2, at the standard resolution of 640×640 pixels, the model provides an inference time of 10.43 ms, which corresponds to 96 FPS. This value significantly exceeds the real-time processing threshold of 24-30 FPS. When the resolution is increased to 1280×1280 pixels, the inference time increases to 18.83 ms, while the processing speed decreases to 53 FPS. Therefore, the 640×640 configuration can be considered the most balanced mode in terms of speed and detection quality.

To visually assess the adequacy of the system, testing was carried out on real images that did not participate in training (Fig. 5a, b and 6 a, b).

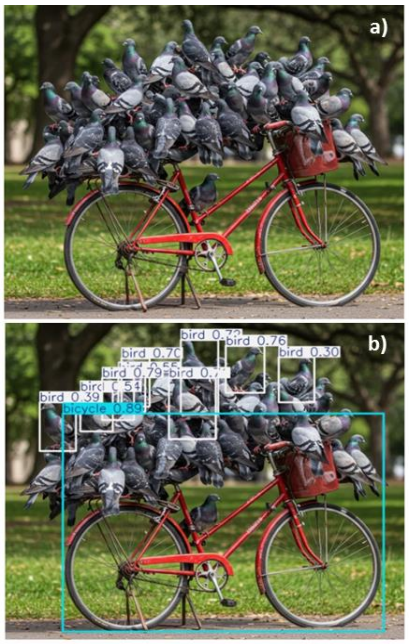

Figure 5: Example of object detection results: original image a), annotated image with detected objects and confidence scores; b) obtained using the YOLOv8n model.

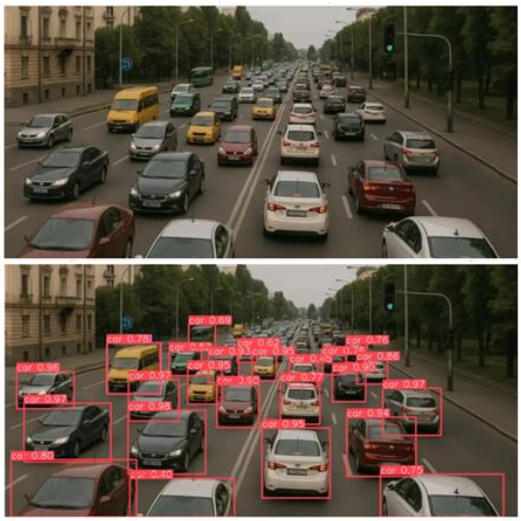

Figure 6: Original photo: a) Vehicle detection in the evening; b) demonstration of low-light resistance.

It should be noted that to improve the interpretation of the neural network output data, a visualization module was developed that imposes bounding boxes and class labels on the output image. The key problem in the visualization of multi-class detection is ensuring perceptual distinguishable objects. The use of random color generation is undesirable, since during video stream processing the color of the frame of the same object may change from frame to frame, which leads to the appearance of visual noise and a flickering effect.

To solve this problem, a deterministic color generation algorithm based on class name hashing is implemented. For each class i, the hash value H is calculated from the class name according to (1):

$$H = Hash(Name_i). \quad (1)$$

The RGB color components are then obtained from the hash value using (2)-(4):

$$R = (H >> 16) \ \& \ 0xFF. \quad (2)$$

$$G = (H >> 8) \ \& \ 0xFF. \quad (3)$$

$$B = H \ \& \ 0xFF. \quad (4)$$

This approach ensures that all objects of the same class (e.g., Person) will always be marked with the same unique color regardless of the system startup time or the order of frame processing.

5 CONCLUSIONS

This paper addresses the problem of improving the efficiency of computer vision systems by proposing and evaluating an image object detection method using convolutional neural networks (CNN) and pre-trained models. The application of Transfer Learning methodology allows reducing the network training time and achieving model convergence in just 10-50 epochs even on limited datasets. A model adaptation method has been developed, which includes the use of pre-trained weights (on the COCO dataset) and a fine-tuning strategy for the specifics of new classes.

For practical implementation, the YOLOv8n (Nano) architecture was selected, as it provides the best balance between inference speed and detection accuracy, making it suitable for real-time applications. The PASCAL VOC 2012 reference dataset was used as an experimental base. A statistical analysis of the class distribution was performed. The speed of image processing allows the system to be used in real time.

The software package was implemented in the Google Colab cloud environment using Python with GPU support. It includes modules for automated data loading, dataset format conversion, model training using the Ultralytics library, evaluation of detection results, inference speed measurement, and visualization of the obtained detections. The effectiveness of the proposed approach was experimentally evaluated on the PASCAL VOC validation subset using standard object detection metrics.

The model achieved Precision = 0.7563, Recall = 0.6458, mAP@0.5 = 0.7152, and mAP@0.5:0.95 = 0.5210. The higher Precision value compared with Recall indicates that the model tends to reduce the number of false-positive detections, although some objects may still be missed in complex scenes. The obtained mAP@0.5 value should not be interpreted as state-of-the-art performance, but rather as a practical trade-off between detection quality and real-time inference speed for a lightweight model, which achieved 95.9 FPS at an input resolution of 640 × 640 pixels.

The practical value of the work lies in creating a full cycle (pipeline) of developing a detection system: from data preparation to a ready-made inference module. The developed toolkit is universal and can be quickly adapted to solve applied tasks (video surveillance, quality control, traffic monitoring) by replacing the dataset without changing the software architecture.

Prospects for further research are seen in optimizing the model for work on mobile devices (Edge Devices) using quantization methods, as well as in expanding the functionality for tracking objects in the video stream.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, pp. 436-444, 2015, [Online]. Available: https://doi.org/10.1038/nature14539.

[2] R. Szeliski, Computer Vision: Algorithms and Applications, Cham: Springer, 947 p., 2023, [Online]. Available: https://doi.org/10.1007/978-3-030-34372-9.

[3] S. K. Rao, Artificial Neural Networks: Advanced Applications, New Delhi: Discovery Publishing House PVT LTD, 248 p., 2024.

[4] A. B. Lozynskyy, I. M. Romanyshyn, and B. P. Rusyn, "Intensity estimation of noise-like signal in presence of uncorrelated pulse interferences," Radioelectronics and Communications Systems, vol. 62, no. 5, pp. 214-222, 2019, [Online]. Available: https://doi.org/10.3103/S0735272719050030.

[5] C. Wöhler, 3D Computer Vision: Efficient Methods and Applications, London: Springer, 400 p., 2014.

[6] O. Alonso and R. Baeza-Yates, Information Retrieval: Advanced Topics and Techniques, New York: Association for Computing Machinery, 836 p., 2024, [Online]. Available: https://dl.acm.org/doi/book/10.1145/3674127.

[7] T. Filimonova, O. Pursky, V. Babenko, A. Nechepourenko, V. Shvets, and V. Gamaliy, "Text Sentiment Analysis using Different Types of Recurrent Neural Networks," in 2024 5th International Conference on Image Processing and Capsule Networks (ICIPCN), Jul. 2024, [Online]. Available: https://doi.org/10.1109/ICIPCN63822.2024.00068.

[8] J. F. Peters, Foundations of Computer Vision: Computational Geometry, Visual Image Structures and Object Shape Detection, Cham: Springer, 744 p., 2017, [Online]. Available: https://doi.org/10.1007/978-3-319-52483-2.

[9] Y. P. Putyatin, Image Processing in Computer Vision Systems, Kharkiv: Prom-Art Publishing, 288 p., 2019.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2016, [Online]. Available: https://doi.org/10.1109/CVPR.2016.90.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2016, [Online]. Available: https://doi.org/10.1109/CVPR.2016.91.

[12] J. Fan, S. Deng, X. Song, J. Liu, and Y. Sun, "A gradient-based lightweight network automated design method for facial expression recognition," Expert Systems with Applications, vol. 296, p. 129130, 2026, [Online]. Available: https://doi.org/10.1016/j.eswa.2025.129130.

[13] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," International Journal of Computer Vision, vol. 128, pp. 261-318, 2020, [Online]. Available: https://doi.org/10.1007/s11263-019-01247-4.

[14] S. Ruder, "An overview of multi-task learning in deep neural networks," arXiv, 2017, [Online]. Available: https://doi.org/10.48550/arXiv.1706.05098.

[15] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2020, [Online]. Available: https://doi.org/10.1109/CVPR42600.2020.01079.

[16] R. C. Gonzalez and R. E. Woods, Digital Image Processing, London: Pearson, 1168 p., 2018.

[17] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2014, [Online]. Available: https://doi.org/10.1109/CVPR.2014.81.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 2017, [Online]. Available: https://doi.org/10.1109/TPAMI.2016.2577031.

[19] M. Everingham, S. M. Eslami, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes Challenge: A Retrospective," International Journal of Computer Vision, vol. 111, no. 1, pp. 98-136, 2015, [Online]. Available: https://doi.org/10.1007/s11263-014-0733-5.

[20] 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, [Online]. Available: https://doi.org/10.1007/s11263-015-0816-y.

[21] "PASCAL VOC 2012. A Benchmark Dataset for Multi-Class Object Detection," [Online]. Available: https://www.kaggle.com/datasets/banuprasadb/pascal-voc-2012, [Accessed: Jan. 14, 2026].