Method of Dataset Filling and Recognition of Moving Objects in Video Sequences based on YOLO

Mariia Nazarkevych[1], Vasyl Lytvyn[1], Maryna Kostiak[1], Nazar Oleksiv[1], and Nazar Naconechnyi[1]

[1] *Lviv Polytechnic National University, 12 S. Bandera str., Lviv, 79000, Ukraine*

Abstract

The method of filling the dataset of mobile military objects was studied. In the future, objects captured by a video camera mounted on a moving object will be investigated. The software used is YOLO v8, which allows you to track moving objects that fall into the video from the video camera. We use artificial intelligence methods to recognize military equipment. Improvements in object recognition can be achieved by contour analysis, pattern comparison, and point-by-point comparison. In future works, we will develop recognition of these theories. The metrics used to evaluate object recognition are shown. A method of filling the dataset and creating a classifier is proposed. Shown are graphs, the results of moving object recognition in Yolo8x.

Keywords

AI, machine learning, YOLO, computer vision, military.

1. Introduction

Today, with the rapid development of artificial intelligence based on deep learning technology, convolutional neural networks [1–4] have made a breakthrough in the field of computer vision, greatly increasing the flexibility and automation of production [5–9].

In the field of computer vision, machine vision is involved in the automatic recognition of moving objects, and the detection of obstacles during the movement of objects [10].

2. Review of Literature

The YOLO series has been updated to YOLO v8 [8]. To further improve the performance of existing object detection algorithms, many scientists have begun to study them. In particular, [9] proposed an object detector based on depth features. It enhances the ability to train the network through multi-scaling training methods without increasing the network size and integrates multiple tracking features of observation models for accurate object location. However, this method lacks detection and false detection in complex scenes with many objects.

In an era marked by rapid advancements in Artificial Intelligence (AI) and computer vision, the convergence of these technologies has ignited revolutionary progressions with profound implications. The domain of military applications stands at the forefront of harnessing AI's potential, particularly within the sphere of computer vision. This article embarks on a critical exploration of this aspect, delving into the importance of training AI models, specifically focusing on the You Only Look Once (YOLO) model, for the recognition of military objects, with a specific emphasis on tanks.

As the velocity of technological innovation intensifies, AI and computer vision have evolved into crucial instruments that reshape our perceptions and interactions with the environment. In military contexts, the incorporation of computer vision capabilities holds paramount significance, providing heightened situational awareness, facilitating

CPITS-2024: Cybersecurity Providing in Information and Telecommunication Systems, February 28, 2024, Kyiv, Ukraine
EMAIL mariia.a.nazarkevych@lpnu.ua (M. Nazarkevych); vasyl.v.lytvyn@lpnu.ua (V. Lytvyn); maryna.y.kostiak@lpnu.ua (M. Kostiak); nazar.oleksiv.mnsa.2020@lpnu.ua (N. Oleksiv); nazar.nakonechnyi.mttop.2022@lpnu.ua (N. Naconechnyi)
ORCID: 0000-0002-6528-9867 (M. Nazarkevych); 0000-0002-9676-0180 (V. Lytvyn); 0000-0002-6667-7693 (M. Kostiak); 0000-0001-7821-3522 (N. Oleksiv); 0009-0000-2456-3498 (N. Naconechnyi)

strategic decision-making, and optimizing overall operational efficiency. The rapid and accurate identification of military objects, such as tanks, is imperative in environments characterized by dynamism and unpredictability.

Amidst these technological advancements, recent geopolitical events, and the war in Ukraine, accentuate the urgent requirement for advanced AI models capable of detecting military assets like tanks. The deployment of AI in conflict zones confers a strategic advantage by enabling real-time object recognition, aiding in the precise allocation of resources, and ultimately enhancing the safety and effectiveness of military operations.

This article aims to explore how we train a sophisticated model called YOLO to recognize military objects. We are going to talk about the methods we use, the difficulties we face, and what could happen as a result. The goal is to show how important AI and computer vision are in changing how military technology works, which has big effects on keeping things safe and defending our countries.

Various methods are used to search for objects in the image when applying computer vision methods.

There are three main methods of object recognition in the image:

- Contour analysis [10].
- Template matching [11].
- Feature detection, description & matching [12].

One of the methods for recognizing objects from a video stream is using pattern search methods. This method has information about what the required object looks like, what kind of background it can have, how certain contours of the object look, and at what positions they can be, immediately considering the possible location of the object detection. It allows us to achieve a high quality of recognition and has a good speed. However, when the video camera captures several objects that are similar to each other, different patterns are satisfied and recognition decreases. Therefore, apply a family of models to estimate functions.

Google Collaboratory [13] ("Colab", a cloud version of Jupyter Notebook) will be used to train neural networks. Using Colab does not require installing and running or upgrading your computer hardware to meet Python's CPU/GPU intensive requirements. In addition, Colab provides free access to computing infrastructure such as storage, RAM, computing power, and processing with graphics units (GPUs) [14] and tensor processing units (TPUs) [15].

A workspace will be organized before the project is developed. To identify enemy targets, DataSet padding will be performed for the corresponding data set. This process will include searching for images and videos of the above-mentioned objects and marking the corresponding objects. The data set will consist of tens of thousands of unique images—approximately equally for each class.

2.1. Contour Analysis

One of the methods used to determine moving objects is contour analysis [16], which is a method of describing, storing, recognizing, comparing, and searching for graphic images (objects) based on their contours. The contour completely defines the shape of the image and contains all the necessary information for recognizing images by their shape. This approach allows you not to consider the internal points of the image and thereby significantly reduce the amount of information that is transformed. Contour—a curve that describes the boundary of the object in the image. Therefore, it is possible to consider the contours of objects, which reduces the complexity of algorithms. The main advantage of the contour analysis is the invariance concerning the rotation, scale, and shift of the contour in the image.

It is well suited for searching for an object of a given shape. As a result, it is often possible to ensure the system works in real-time. However, there are significant disadvantages of this method. There are breaks in the outline in places. Thus, the contour analysis has a rather weak resistance to interference, and any violation of the integrity of the contour or poor visibility of the object leads to either the impossibility of detection or false positives. Simplicity and speed of contour analysis allow you to successfully apply this approach, provided there is a well-defined object on a contrasting background.

2.2. Template Matching

The template comparison method [17] is performed according to the criterion of the minimum (maximum) of some function comparing the object and its template. This method is one of the simplest. The input parameters of the method are the image on which the template must be searched, the image of the object that must be found on the tested image, and the size of the template must be smaller than the size of the tested image. The purpose of the algorithm is to find a section on the tested image that matches the template. Searching for a template is done by sequentially moving it by one pixel across the image, and evaluating the similarity of each new area to the template. Based on the results of the check, the section with the highest coincidence ratio is selected. In essence, this is the percentage of overlap between the image area and the template. The described method of matching with templates is simple, but there is a certain complexity in the process of creating templates, that is, in learning.

Template matching does not allow you to say whether the original object was found because it is a probabilistic characteristic that depends on the scale, viewing angles, rotations of the image, and the presence of physical obstacles [18]. There are also possible false positives of the algorithm when the searched object is not there, but there are some general details in the pattern and area on the tested image. Of course, this situation can be avoided by checking the match factor value (so that it is not less than some threshold), but this will not always work properly.

2.3. Search by Special Points

Often algorithms use key points [19] (feature points) of the image for their work. Key points are understood as some areas of the picture that are distinctive for this image. There are a large number of methods for detecting such "special points", all of them differ in the speed of operation, the number of selected points, as well as resistance to image transformations: rotation, changes in viewing angles, changes in scale.

Three components are used to find key points in images and their subsequent comparison:

- Feature detector[20]—searches for key points on the image.
- Descriptor (descriptor extractor)—produces a description of the found key points, evaluating their positions through the description of the surrounding areas.
- Matcher (matcher)—builds correspondences between two sets of image points.

Unlike template matching and contour analysis, algorithms for finding key points are more resistant to obstacles, and transformations and allow finding objects even in the presence of physical obstacles.

To achieve the highest possible level of tracking of an object (marker), it must have a significant number of unique (stable) key points, which the augmented reality library can quickly highlight in the video stream and compare with the existing template set. For this, it is necessary to use the fastest possible detector and descriptor, as well as to develop an algorithm that could confidently say that the object has been found.

This algorithm allows you to recognize images at different angles, at different distances from the camera, under different lighting, and when the image is partially overlapped [21].

3. Architecture of the YOLO Algorithm

The YOLOv8 [22] model is widely employed for object detection purposes. YOLOv8 is available in four primary versions: small (s), medium (m), large (l), and extra large (x), with each version providing increasing levels of accuracy. Additionally, each variant requires a distinct amount of time for the training process.

The objective of the graph is to create a highly efficient object detector model, with performance indicated on the Y-axis and inference time on the X-axis. Initial findings indicate that YOLOv8 performs exceptionally well in achieving this goal compared to other cutting-edge techniques.

Examining the chart, it's evident that all versions of YOLOv8 exhibit quicker training times than EfficientDet. Specifically, the most accurate YOLOv8 model, YOLOv8x, demonstrates the capability to process images at a significantly faster rate while maintaining

a comparable level of accuracy compared to the EfficientDet D4 [23] model.
The architecture of YOLO involves dividing the input image predicting bounding boxes, into a grid and and class probabilities.

Here's a high-level overview of the YOLO architecture:

Input Layer: YOLO takes an input image, typically of fixed size.

Dividing into Grid: The image is divided into an S×S grid. Each grid cell is responsible for predicting bounding boxes and class probabilities.

Bounding Box Prediction: Each grid cell predicts multiple bounding boxes (usually B bounding boxes). These bounding boxes include information about the coordinates (x, y) of the bounding box, width (w), height (h), and confidence scores.

Class Prediction: For each bounding box, the model predicts class probabilities for different object categories. This is done using softmax activation.

Confidence Score: Each bounding box prediction is associated with a confidence score, indicating how likely it is that the bounding box contains an object. This score takes into account both the probability of object presence and the accuracy of the bounding box.

Final Prediction: The final predictions are obtained by combining the bounding box coordinates, class probabilities, and confidence scores. Non-maximum suppression is then applied to filter out redundant and low-confidence predictions.

Output: The final output is a set of bounding boxes, each associated with a class label and a confidence score.

Loss Function: YOLO uses a multi-part loss function that includes terms for object presence/absence, bounding box coordinates, and class probabilities. This allows the model to be trained for accurate detection.

The YOLO algorithm takes an image as input and uses a deep convolutional neural network to detect objects in the image.

The first 20 convolutional layers of the model are pre-trained using ImageNet [24], using a temporal mean pooling layer and a fully connected layer. After that, this trained model is transformed to perform the object detection task, to the trained network improves its performance. The last fully connected layer of YOLO performs the prediction of both class probabilities and bounding box coordinates.

YOLO assumes multiple bounding boxes for each grid cell. During training, the aim is to ensure that each bounding box predictor is "responsible" for predicting the object based on the prediction with the highest value of sample overlap with the correct information.

To improve object detection accuracy, YOLO uses the Non-Maximum Suppression (NMS) method [25]. This technique allows you to identify and remove redundant or incorrect bounding boxes that may be created for a single object. As a result of applying NMS, only one bounding box is selected for each object in the image, which improves the quality and efficiency of the detection process.

3.1. The Difference Between YOLO and Other Deep Learning Algorithms for Object Detection

The main difference between YOLO algorithms used for object detection is that it recognizes objects quickly in real-time. The principle of operation of YOLO involves entering the entire image at once, which passes through the convolutional neural network only once [26].

The performance of the YOLO algorithm is evaluated using the COCO dataset [27] (Common Objects in Context). The COCO dataset is a large dataset consisting of 80 feature classes. YOLOv1, the baseline version, can recognize 24 object classes and has a 21.6% mAP (average accuracy) measured using the COCO dataset. YOLOv2 can recognize 90 feature classes with a COCO dataset mAP of 30.2%. YOLOv3 can recognize 1000 feature classes with an mAP of 57.9%. YOLOv4 has a better performance compared to YOLOv3 but can recognize 80 feature classes with mAP of 60.0%. YOLOv8 achieves significant performance improvements over YOLOv4 and achieves an mAP of 83.5% on the COCO dataset. YOLOv6 achieved 84.4% of the COCO mAP dataset. YOLOv7 achieved a mAP of 85.4%. YOLOv8 provides additional performance improvements over YOLOv7 and achieves an mAP of 86.4%.

However, the YOLO algorithm has its drawbacks. Spatial limitations allow only 8Bbox to be projected per grid cell, making it difficult to distinguish objects that are close together. Multiple samples are used and a lack

of detail is often apparent. The third problem is imprecise localization, and finally, since Bbox training is performed based on data, it is difficult for the algorithm to detect a test dataset that does not exist in the training data. However, the most difficult aspect of deep learning is the preparation of training data, and the data applicable to each application domain is very limited.

Despite all the limitations, the YOLO model still has a significantly higher processing speed than other models and continues to be widely used.

4. Construction of the Model

Artificial neural networks are complex computing systems similar to the corresponding systems of the human brain. They consist of artificial neurons that receive input signals from a certain number of connections to the neurons of the previous layer or the input of the network.

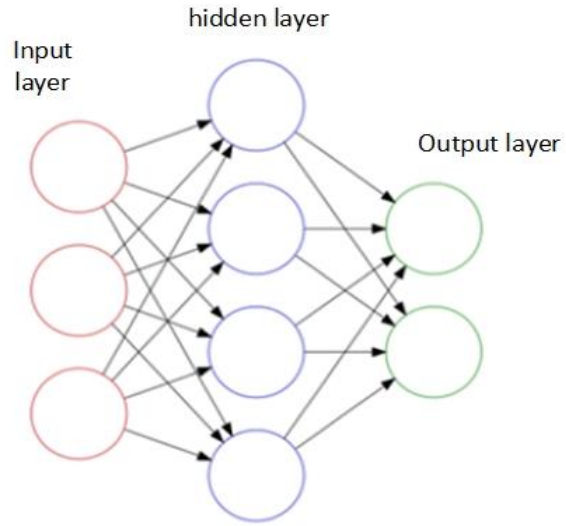

Figure 1: Scheme of a simple neural network with one hidden layer

The scheme of a simple neural network with one hidden layer is shown in Fig. 1. The process of learning a neural network consists of adjusting the parameters of the network to obtain better results. For most problems in neural networks, tutoring is used, when the system is fed data with a ready result to gradually train the network with real data, where the correct answer will not be known in advance. Improving the results and accuracy of the network is done by minimizing a certain error, the difference between the predicted and actual outputs.

Accuracy measures how well the model predicts the correct outcome compared to the actual outcome. It is calculated by dividing the number of correct predictions by the total number of predictions. Although accuracy is a useful metric, it may not be sufficient to evaluate the performance of complex AI models. Therefore, it is important to consider other metrics such as precision, repeatability, and F1-score [28].

Classification errors—confusion matrix [29] (error matrix). If there are two classes and an algorithm that predicts each object to one of the classes, then the classification error matrix will look like this:

Table1
Confusion matrix

	y=1	**y=0**
$\check{y}=1$	True Positive (TP)	False Positive (FP)
$\check{y}=0$	False Negative (FN)	True Negative (TN)

where $\check{y}$ is the answer of the algorithm on the object, and y is the true label of the class on this object.

Accuracy measures the ratio of correct positive predictions to all positive predictions made by the model. It shows how well the model will avoid false positives. Repeatability, on the other hand, measures the ratio of correct positive predictions to all actual positive cases. It shows how well the model will avoid false negatives.

In this way, classification errors are applied, which are: False Negative (FN) and False Positive (FP) [29].

$$P = \frac{TP}{TP+FP} \times 100\%, \quad (1)$$

$$R = \frac{TP}{TP+FN} \times 100\%. \quad (2)$$

The F1-score is a combination of precision and repeatability, providing a balanced estimate of model performanceP).

$$RF1 = \frac{2\times P\times R}{P+R} \times 100\%. \quad (3)$$

Reference data sets are traditionally used for preliminary comparison of models. The COCO dataset uses a special mAP averaged accuracy metric mAP.

$$AP = \int_0^1 P(R)\, dR, \quad (4)$$

$$mAP = \frac{1}{n}\sum_{i=1}^{3} \frac{TP}{FP+TP}. \quad (5)$$

This is an average accuracy measure, which is taken from different values of the intersection over the union.

5. Construction of the Classifier

The construction of the classifier makes it possible to evaluate the preferences of the choice based on the analysis of information about the preferences of users.

We will use an artificial neural network to build a classifier.

A neural network based on a multilayer perceptron is a system of interconnected layers of neurons. Each neuron is characterized by an activation function that converts the neuron's input signal into an output signal. Connections of neurons with other neurons are characterized by connection coefficients—weights. An important factor in learning a neural network is the type of input data. To achieve the best results, it is necessary to preliminarily display the data using centering and scaling operations (1):

$$xn = (x - mx)\, mx. \quad (6)$$

The learning process is an iterative sequence of operations for calculating the output signal of the network and subsequently changing the weights of the connections. As an algorithm for weight adjustment in MLP-based networks, the error backpropagation algorithm is usually used. It refers to methods of learning with a teacher, so it requires that target values be set in the training examples. This algorithm belongs to the class of gradient algorithms, that is, the changes in the weights of connections are made in the direction of minimizing the gradient of the error. The prediction error during training is equal to the difference between the signal at the network output and the output reference value corresponding to the input data (2).

$$ei = (yi - di). \quad (7)$$

The training of the network must be carried out until the average value of the error during one training epoch decreases. Further training usually leads to a deterioration in the analytical capabilities of the neural network.

The network was trained using the error backpropagation algorithm and the gradient descent algorithm. The basis of the idea of the algorithm is the use of the initial error of the neural network (7) to calculate the correction values of the neuron weights in its hidden layers:

$$E = 1\ 2 \sum (y - y')\ k\ 2\ i = 1, \quad (8)$$

where k is the number of output neurons of the network, y is the target value; and y' is the actual output value.

This algorithm is iterative, it uses step-by-step learning. It works as follows: one test example is given to the learning input. Then the weight values are adjusted. At each iteration, forward and reverse passes of the network take place. On a forward input, the vector propagates from the inputs to the outputs of the network. In this way, a certain output vector is formed, which corresponds to the current (actual) state of the scales. The error of the neural network is calculated as the difference between the actual and target values.

On the return pass, this error is propagated from the output of the network to its inputs, and the neuron weights are corrected according to formula (4):

$$\Delta w_{ji}(\mathrm{n}) = -\eta\, \partial E_{av}\, \partial w_{ij}, \quad (9)$$

where w_{ji} is the weight of the i^{th} connection of the j^{th} neuron; η is a learning speed parameter that allows you to additionally control the size of the correction step; Δw_{ji} for more accurate adjustment to the minimum error and is selected experimentally in the learning process (changes in the interval from 0 to 1).

5.1. The Method of Stochastic Gradient Descent

The stochastic gradient descent method belongs to optimization algorithms and is used to adjust the parameters of the machine learning model. The gradient is usually considered the sum of the gradients caused by each training element. The parameter vector changes in the direction of the anti-gradient with a given step. Therefore, standard gradient descent requires one pass over the training data before it can change the parameters. In stochastic (or "operational") gradient descent, the value of the gradient is approximated by the gradient of the cost function calculated on only one training element. The parameters then change in proportion to the approximate gradient. Thus, the parameters of the model change after each training object. For large datasets, stochastic gradient descent can

provide a significant speed advantage over standard gradient descent.

We will use the Python language to build the classifier. One of the main reasons why Python is used for machine learning is that it has many frameworks that simplify the process of writing code and reduce development time.

5.2. Save the Video

So, we shoot a video and process it frame by frame [30], and we want to save this video. We do pre-processing of images [31].

We create a VideoWriter object. Specify the name of the output file (output.avi). Then we specify the FourCC code and transfer the number of frames per second (fps) and the frame size. And the last one is Color. If True, the encoder expects a color frame, otherwise it works with a grayscale frame. FourCC is a 4-byte code used to identify the video codec. XVID codec is better. MJPG creates large-size videos. X264 gives a small video size). On Windows: DIVX.

6. Experiment Environment

As part of the research, a proprietary dataset was utilized, comprising over a thousand images of tanks from various models and perspectives. These images were sourced from electronic books and websites, rendering them diverse and realistic [32]. Accordingly, it was necessary to manually add labels to the images for subsequent model training. This process ensured proper classification and recognition of tanks in the images.

The decision was made to employ the Roboflow platform for efficient uploading and processing of images.

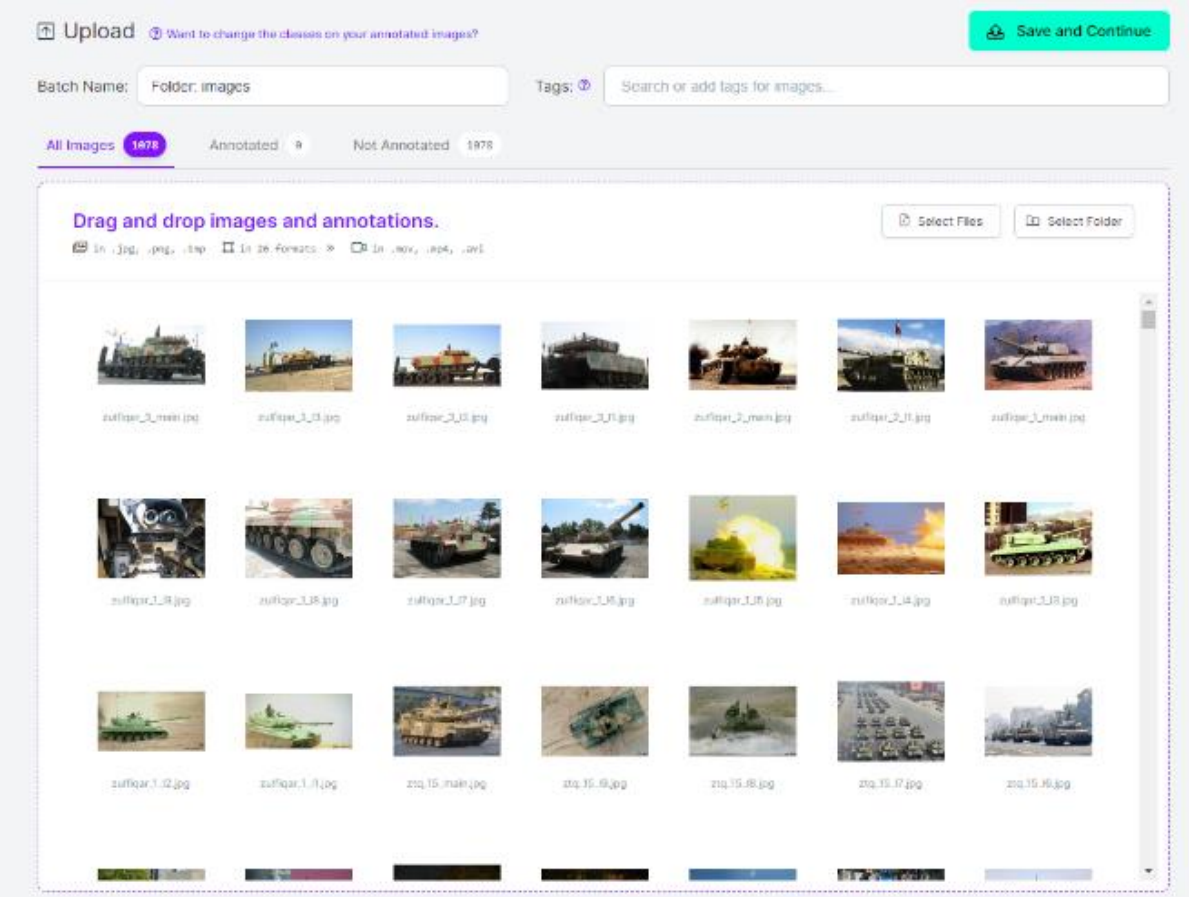

Figure 2: Dataset filling

This not only facilitated a swift resolution of the task but also accurately assigned labels to each object in the images, providing the model with sufficient information for tank recognition and classification.

Figure 3: Filling the dataset with photos of tanks from different viewing angles

Figure 4: Recognition of tanks

These steps in utilizing a proprietary dataset and the Roboflow platform hold significant importance in advancing the tank recognition model, particularly in enhancing its accuracy and efficiency during training.

As a result of training, the model demonstrated impressive accuracy, achieving a high level of correct classification of objects in the images [33]. The graphs attached to the article illustrate the model's high stability and effectiveness during training, confirming its ability to confidently recognize tanks in various scenarios and conditions [34].

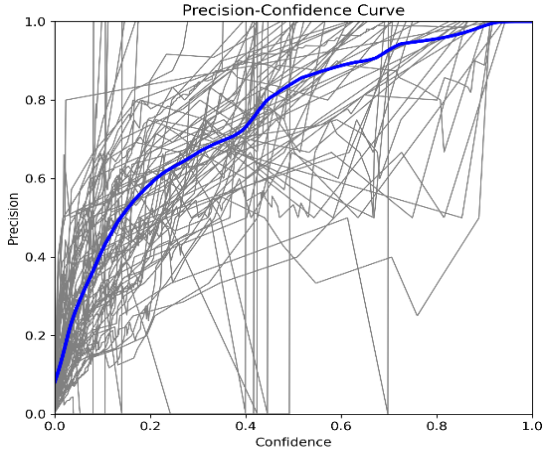

Figure 5: Precision—Confidence curve

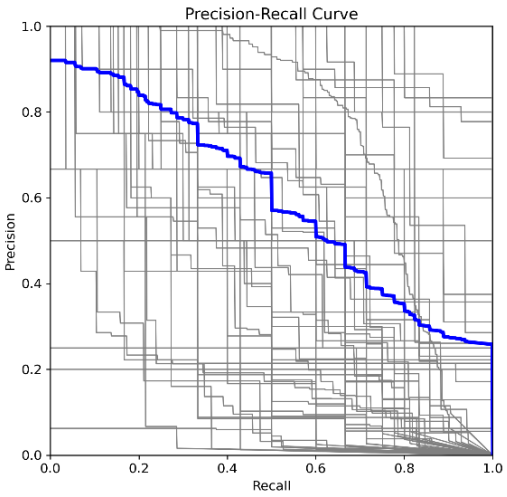

Figure 6: Precision-Recall curve

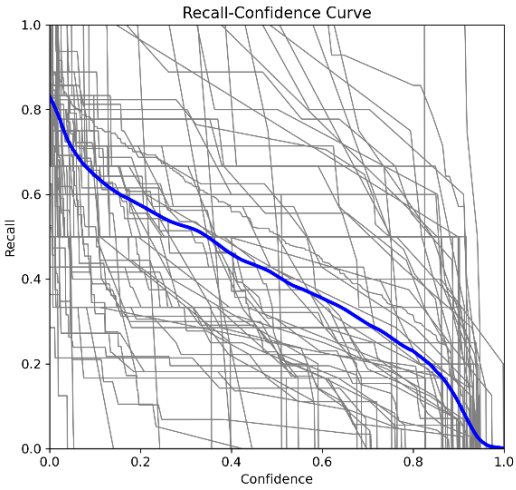

Figure 7: Recall—Confidence curve

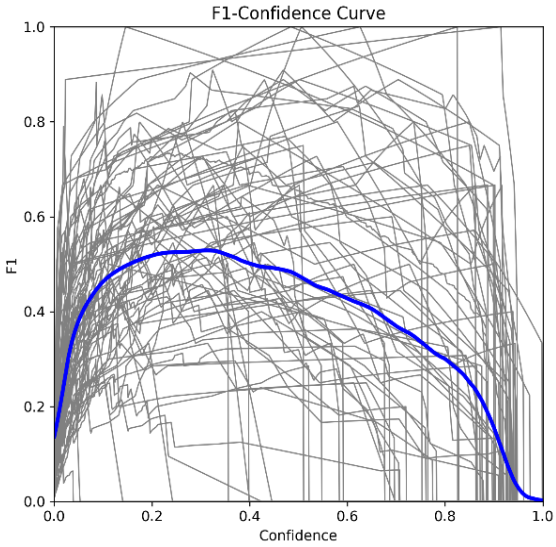

Figure 8: F1 Confidence curve

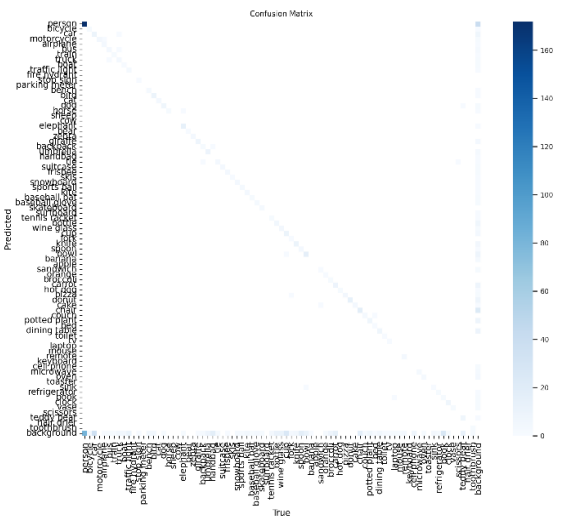

Figure 9: Convolution matrix

Furthermore, the trained model exhibited minimal losses during training, indicating high efficiency and optimal adaptation to the training data. The training and loss graphs depict a stable learning process and the absence of significant fluctuations in losses during model optimization [35, 36].

Overall, the obtained results affirm the high potential and efficiency of the developed model for tank recognition, making it a significant contribution to the field of military applications and security objects.

6.1.1. Model Testing

We conducted model testing under various conditions, including image analysis and video streaming:

6.1.2. Image Testing

The model was evaluated using diverse images of tanks, encompassing various models sourced from different outlets [37, 38]. Under normal lighting conditions and various viewing angles, the model successfully recognized tanks, demonstrating a high level of accuracy.

6.2. Video Stream Testing

We also conducted testing using a video stream to assess the model's real-time tank recognition capabilities. In this mode, the model proved to be quite effective, adapting quickly to changes in the images and confidently recognizing tanks in different scenarios.

Figure 10: The result of Yolo8 work, recognition in a video stream

Figure 11: The result of Yolo8 work, recognition in a video stream

Figure 12: The result of Yolo8 work

Despite the overall success, it is worth noting that the model struggled to recognize tanks in low-light conditions, such as during nighttime without thermal imaging illumination or in snowy conditions. This aspect should be considered as a potential area for further improvements and optimizations to the model.

7. Conclusions

Our research project encompassed several key stages aimed at creating and refining a tank recognition model using YOLO (You Only Look Once). We generated our dataset, consisting of

diverse tank images from various sources. This dataset provided crucial material for training and recognizing different tank models. Utilizing the created dataset, we successfully trained the YOLO model to recognize tanks. The training methodology involved annotation and systematic training to achieve high accuracy. We assessed the model's effectiveness across various media, including photos and videos. The model demonstrated a high level of tank recognition in real-time and images. The testing results revealed impressive accuracy and efficiency in recognizing tanks under different conditions and perspectives. Upon analysis, the next step could involve refining the model for tank recognition in low-light conditions, such as nighttime without thermal illumination, or during adverse weather conditions like snowfall. In conclusion, the developed model holds significant potential in military applications and security domains. Further enhancements could make it even more effective in real-world conditions.

References

[1] Z. Li, et al., A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects, IEEE Transactions on Neural Networks and Learning Systems 33(12) (2021) 6999–7019. doi: 10.1109/TNNLS.2021.3084827.

[2] V. Grechaninov, et al., Decentralized Access Demarcation System Construction in Situational Center Network, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems II, vol. 3188, no. 2 (2022) 197–206.

[3] P. Anakhov, et al., Protecting Objects of Critical Information Infrastructure from Wartime Cyber Attacks by Decentralizing the Telecommunications Network, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems, vol. 3550 (2023) 240–245.

[4] H. Hulak, et al., Dynamic Model of Guarantee Capacity and Cyber Security Management in the Critical Automated System, in: 2nd International Conference on Conflict Management in Global Information Networks, vol. 3530 (2023) 102–111.

[5] V. Buriachok, V. Sokolov, P. Skladannyi, Security Rating Metrics for Distributed Wireless Systems, in: Workshop of the 8th International Conference on "Mathematics. Information Technologies. Education:" Modern Machine Learning Technologies and Data Science, vol. 2386 (2019) 222–233.

[6] M. Vladymyrenko, et al., Analysis of Implementation Results of the Distributed Access Control System. in: IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (2019). doi: 10.1109/picst47496.2019.9061376.

[7] Y. Qiao, et al., Automatic Recognition of Static Phenomena in Retouched Images: A Novel Approach, Advanced Technologies for the Implementation of New Ideas (2024) 287–291.

[8] Z. Wang, et al., E-YOLO: Recognition of Estrus Cow Based on Improved YOLOv8n Model, Expert Syst. Appl. 238 (2024) 122212. doi: 10.1016/j.eswa.2023.122212.

[9] Y. Zhang, et al., DsP-YOLO: An Anchor-Free Network with DsPAN for Small Object Detection of Multiscale Defects, Expert Syst. Appl. 241 (2024) 122669. doi: 10.1016/j.eswa.2023.122669.

[10] M. Patel, et al., 3D Back Contour Metrics in Predicting Idiopathic Scoliosis Progression: Retrospective Cohort Analysis, Case Series Report and Proof of Concept, Children 11(2) (2024) 159. doi: 10.3390/children11020159.

[11] Q. Wang, et al., Transformer-Based Multiple-Object Tracking via Anchor-Based-Query and Template Matching, Sensors 24(1) (2024). 229 doi: 10.3390/s24010229.

[12] H. Lu, J. Nie, Coarse Registration of Point Cloud Base on Deep Local Extremum Detection and Attentive Description, Multimedia Syst. 30(1) (2024), 23. doi: 10.1007/s00530-023-01203-w.

[13] S. Shekhar, N. Thakur, Deep Learning Framework for Forecasting Diabetic Retinopathy: An Innovative Approach, Int. J. Innov. Res. Comput. Sci. Technol. 12(1)

(2024) 17–20. doi: 10.55524/ijircst.2024.12.1.4.

[14] U. Utkarsh, et al., Automated Translation and Accelerated Solving of Differential Equations on Multiple GPU Platforms, Comput. Methods Appl. Mech. Eng. 419 (2024) 116591. doi: 10.48550/arXiv.2304.06835.

[15] P. Feng, et al., Co-Continuous Structure Enhanced Magnetic Responsive Shape Memory PLLA/TPU Blend Fabricated by 4D Printing, Virtual Phys. Prototyp. 19(1) (2024) e2290186. doi: 10.1080/17452759.2023.2290186.

[16] J. Malik, et al., Contour and Texture Analysis for Image Segmentation, Int. J. Comput. Vision 43 (2001) 7–27. doi: 10.1023/A:1011174803800.

[17] N. Hashemi, et al., Template Matching Advances and Applications in Image Analysis, arXiv preprint (2016).

[18] G. Cox, Template Matching and Measures of Match in Image Proce-Ssing, University of Cape Town (1995).

[19] D. Lowe, Distinctive Image Features from Scale-Invariant Keypoints, Int. J. Comput. Vision 60 (2004) 91–110.

[20] D. Mukherjee, M. Wu, G. Wang, A Comparative Experimental Study of Image Feature Detectors and Descriptors, Mach. Vision Appl. 26 (2015) 443–466. doi: 10.1007/s00138-015-0679-9.

[21] L. Liu, et al., CLFR-Det: Cross-Level Feature Refinement Detector for Tiny-Ship Detection in SAR Images, Knowledge-Based Syst. 284 (2024). doi: 10.1016/j.knosys.2023. 111284.

[22] Q. Liu, et al., YOLOv8-CB: Dense Pedestrian Detection Algorithm Based on In-Vehicle Camera, Electronics 13(1) (2024) 236. doi: 10.3390/electronics 13010236.

[23] F. Pan, et al., Zero-shot Building Attribute Extraction from Large-Scale Vision and Language Models, IEEE/CVF Winter Conference on Applications of Computer Vision (2024) 8647–8656.

[24] H. Li, C. Wang, Y. Liu, YOLO-FDD: Efficient Defect Detection Network of Aircraft Skin Fastener, Signal, Image and Video Processing (2024) 1–15.

[25] S. Koga, et al., Optimizing Food Sample Handling and Placement Pattern Recognition with YOLO: Advanced Techniques in Robotic Object Detection, Cognitive Robotics (2024). doi: 10.1016/j.cogr.2024.01.001.

[26] Y. Wang, et al., GT-YOLO: Nearshore Infrared Ship Detection Based on Infrared Images, J. Marine Sci. Eng. 12(2) (2024) 213. doi: 10.3390/jmse12020213.

[27] K. Gupta, A. Asthana, Reducing the Side-Effects of Oscillations in Training of Quantized YOLO Networks, IEEE/CVF Winter Conference on Applications of Computer Vision (2024) 2452–2461.

[28] P. Giudici, M. Centurelli, S. Turchetta, Artificial Intelligence risk measurement, Expert Systems with Applications 235 (2024) 121220.

[29] S. Shinde, et al., Artificial Intelligence Approach for Terror Attacks Prediction Through Machine Learning, Multidiscip. Sci. J. 6(1) (2024) 2024011–2024011.

[30] M. Nazarkevych, et al., Evaluation of the Effectiveness of Different Image Skeletonization Methods in Biometric Security Systems, Int. J. Sens. Wirel. Commun. Control 11(5) (2021) 542–552. doi**:** 10.2174/2210327910666201210151809.

[31] M. Nazarkevych, et al., The Ateb-Gabor Filter for Fingerprinting, International Conference on Computer Science and Information Technology (2019) 247–255. doi: 10.1007/978-3-030-33695-0_18.

[32] M. Nazarkevych, et al., Data Protection Based on Encryption Using Ateb-Functions, 9th International Scientific and Technical Conference Computer Sciences and Information Technologies (2016) 30–32.

[33] M. Medykovskyy, et al., Methods of Protection Document Formed from Latent Element Located by Fractals, Xth International Scientific and Technical Conference Computer Sciences and Information Technologies (2015) 70–72.

[34] V. Sheketa, et al., Formal Methods for Solving Technological Problems in the Infocommunications Routines of Intelligent Decisions Making for Drilling Control, IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (2019) 29–34.

[35] V. Sheketa, et al., Empirical Method of Evaluating the Numerical Values of Metrics in the Process of Medical Software Quality Determination, International Conference on Decision Aid Sciences and Application (2020) 22–26. doi: 10.1109/DASA51403.2020.9317218.
[36] N. Boyko, N. Tkachuk, Processing of Medical Different Types of Data Using Hadoop and Java MapReduce, in: 3rd International Conference on Informatics & Data-Driven Medicine Vol. 2753 (2010) 405–414.
[37] N. Boyko, et al., Fractal Distribution of Medical Data in Neural Network, IDDM (2019) 307–318.
[38] I. Tsmots, et al., The Method and Simulation Model of Element Base Selection for Protection System Synthesis and Data Transmission, Int. J. Sens. Wirel. Commun. Control 11(5) (2021) 518–530.