Abstract

The aim of this work is to develop a recurrent neural network based on the YOLO architecture for document detection in a sequence of frames. The main tasks include developing and training a basic YOLOv3 model, implementing algorithmic tracking to improve object detection in video streams, and integrating a recurrent layer into YOLOv3 to account for temporal dependencies between frames.

During the research, a model based on YOLOv3 was created, demonstrating high object detection accuracy on individual frames. To improve the analysis of video sequences, algorithmic tracking was implemented. The integration of a recurrent layer (GRU) into the YOLOv3 model allowed for the consideration of temporal dependencies between frames, which improved the quality of object detection in video streams.

Based on this work, it is recommended to use the proposed architecture for tasks requiring high accuracy and stability of real-time object detection, such as surveillance systems, autonomous driving, and document recognition. Future research may focus on optimizing the model for various types of data, improving tracking algorithms, and applying other recurrent layers to further enhance detection accuracy.

Contents

1 Symbols and Abbreviations

Symbol	Explanation
YOLO	You Only Look Once (object detection algorithm)
IoU	Intersection over Union (quality metric for evaluating the overlap of predicted and true bounding boxes)
GRU	Gated Recurrent Unit (a type of recurrent neural network)
BCE	Binary Cross-Entropy (binary cross-entropy, loss function)
MSE	Mean Squared Error (mean squared error, loss function)
ReLU	Rectified Linear Unit (activation function)
SymReLU	Symmetrical Rectified Linear Unit (symmetrical ReLU activation function)
PyTorch	Framework for deep learning
Adam	Adam optimizer (adaptive moment estimation method)

2 Introduction

In the context of the rapid development of video analysis and computer vision technologies, the task of object detection in a sequence of frames becomes especially relevant in areas such as surveillance systems, autonomous driving, and document recognition. The use of recurrent neural networks in combination with advanced convolutional neural network architectures, such as YOLO, opens up new possibilities for accurate and efficient real-time object detection.

The aim of the research is to develop a system for efficient object detection in a sequence of frames using the YOLO architecture and the integration of recurrent layers. The research tasks include the development and training of a basic model based on YOLOv3, the implementation of algorithmic tracking to improve object detection in video streams, and the integration of a recurrent layer to account for temporal dependencies between frames.

Research methods include the analysis of scientific and technical literature, dataset modification, the development and testing of neural network architectures using PyTorch, conducting experiments with various network configurations and data augmentations, and a comparative analysis of the models' results.

The scientific and theoretical significance of the work lies in the development of a new neural network architecture that combines convolutional and recurrent components to improve the accuracy and efficiency of object detection in video sequences. The practical significance consists in creating a system that can be applied in real conditions for tasks such as surveillance and document detection, significantly improving the quality and reliability of these systems.

3 Research Overview

3.1 Literature Review

The study "You Only Look Once: Unified, Real-Time Object Detection" (YOLO) [1], conducted by Redmon and colleagues, represents a significant breakthrough in the field of computer vision. YOLO rethinks the object detection task by presenting it as a regression problem for spatially separated bounding boxes and associated class probabilities. The main advantages of YOLO include high processing speed, a global approach to image analysis, and generalizability to various types of images.

The study "Large-Scale Video Classification with Convolutional Neural Networks" [2], conducted by Karpathy and colleagues, focuses on applying convolutional neural networks for large-scale video classification. Key points of the study include scaling CNNs for video, temporal information fusion architectures, and multi-level architectures.

The study "Long-Term Recurrent Convolutional Networks for Visual Recognition and Description" (LRCN) [3], conducted by Donahue and colleagues, focuses on integrating convolutional neural networks (CNN) and recurrent neural networks (RNN) for processing and analyzing video data. LRCN combines CNNs for visual feature extraction and recurrent networks for modeling temporal dependencies.

The study "Detecting Objects with Recursive Feature Pyramid and Switchable Atrous Convolution" [4], conducted by Qiao and colleagues, presents an innovative approach to improving object detection models' performance. The authors introduce the concepts of a recursive feature pyramid and switchable atrous convolution, which enhance the model's ability to detect objects of different scales and improve detection in complex and diverse scenes.

The study "Analysis of the Features of Using Stationary and Mobile Small-Sized Digital Video Cameras for Document Recognition" [5], conducted by Arlazarov and colleagues, discusses the challenges and opportunities of using different types of digital cameras for text document recognition. The study compares stationary and mobile cameras regarding their capabilities and limitations for document recognition tasks.

The study "Algorithms for Border Detection of Printed Characters Used in Optical Character Recognition" [6], conducted by Arlazarov and colleagues, is devoted to character segmentation methods for optical text recognition (OCR). The study examines various approaches to correct character segmentation, which is a critical step in the text recognition process.

The book "Multiple View Geometry in Computer Vision" [7], written by Richard Hartley and Andrew Zisserman, is a fundamental work in computer vision dedicated to the geometry of multiple views. It covers a wide range of topics related to the analysis of images obtained from different viewpoints and offers mathematical methods for their processing and interpretation.

The study "Small Object Intelligent Detection Method Based on Adaptive Recursive

Feature Pyramid" [8], conducted by Zhang and colleagues, proposes a new method for improving small object detection accuracy using an adaptive recursive feature pyramid network (AR-PANet). The adaptive recursive structure and feature fusion methods can be used to enhance object detection accuracy, especially for small objects.

The study "Recurrent Neural Networks for Video Object Detection" [9], conducted by Ahmad and Pettirsch, analyzes the application of recurrent neural networks (RNN) for object detection in video. The authors compare various methods, including those using feature maps, bounding box levels, and streaming networks, allowing for the consideration of temporal context and improving object detection accuracy.

3.2 Definition of the Object and Subject of the Study

Object of Study: systems for object detection in a sequence of frames.
Subject of Study: neural network architectures for document detection in video sequences, including convolutional and recurrent components.

3.3 Research Goals and Objectives

Research Goal: to develop a system capable of efficiently detecting objects in a sequence of frames using the YOLO architecture with the integration of recurrent layers.

Research Objectives:

1. Development and training of a basic detection model based on YOLOv3.
2. Implementation of algorithmic tracking to improve object detection in video streams.
3. Integration of a recurrent layer into YOLOv3 to account for temporal dependencies between frames.

3.4 Dataset

The MIDV-2020 dataset was used for the study. It includes 1000 video clips of identity documents. The documents are ID cards from Albania, Spain, Estonia, Finland, Slovakia, and passports from Azerbaijan, Greece, Latvia, Russia, and Serbia.

The video clips were recorded using iPhone XR and Samsung S10 in 10 different conditions:

1. Low light conditions;
2. Against a keyboard background;
3. Outdoor shooting in natural light;
4. Against a table background;
5. Against fabrics of various textures;

6. Against a text document background;

7. Strong projective distortions of the document;

8. Glare from the sun or a lamp hiding part of the document.

Examples of each condition are presented below.

Figure 3.1 – Examples of images from the MIDV-2020 dataset. Taken from source [10]

The videos are split into frames, and each video has an annotation. The annotation contains the coordinates of the bounding box of the document; the first vertex of the quadrilateral corresponds to the upper left corner of the physical document, and the remaining vertices go clockwise. The annotation also contains other data, but they are not needed for this study. Examples of images from the dataset:

Albanian ID Card

Russian Passport

Spanish ID Card

Figure 3.2 – Examples of images from the MIDV-2020 dataset

3.5 Quality Function

To evaluate the prediction quality of the model, the Intersection over Union (IoU) quality function was used. This metric measures the degree of overlap between the predicted and true bounding boxes.

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

where Area of Overlap is the area of overlap between the predicted and true boxes, and Area of Union is the area of their union. The higher the IoU value, the more accurate the model's prediction.

3.6 Loss Function

A combined loss function was used, including mean squared error for the bounding box coordinates and binary cross-entropy for the prediction confidence.

$$\begin{aligned} \text{Loss} =& \text{BCEWithLogitsLoss}(\text{pred\_confidences}, \text{target\_confidences}) \\ &+ \text{MSELoss}(\text{pred\_boxes}, \text{target\_boxes}) \end{aligned} \tag{1}$$

where pred_confidences are the predicted confidences, target_confidences are the true confidences, pred_boxes are the predicted box coordinates, and target_boxes are the true box coordinates.

3.7 Optimizer

The Adam optimizer with parameters $lr = 0.001$ was used as the optimization algorithm. The optimizer performs the gradient descent step according to the following formula:

$$
\begin{aligned}
&m_0 = 0, v_0 = 0 \\
&g_t = \nabla f_t(\theta_{t-1}) \\
&m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \\
&v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \\
&\hat{m}_t = \frac{m_t}{(1 - \beta_1^t)} \\
&\hat{v}_t = \frac{v_t}{(1 - \beta_2^t)} \\
&\theta_t = \theta_{t-1} - lr \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}
\end{aligned}
$$

where θ_t is the model parameter at step t, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$.

3.8 Training and Validation

The model was trained using the optimizer and loss function described above for 100-300 epochs.

3.9 Research Methods

To achieve the goals and objectives, the following methods were used:

- Analysis of scientific and technical literature on the research topic.
- Modification of the dataset for more convenient work.
- Development and testing of neural network architectures using the PyTorch framework.
- Conducting experiments with various network configurations and data augmentations.
- Comparative analysis of the models' results using key metrics.

3.10 Scientific-Theoretical and Practical Significance of the Research

The scientific and theoretical significance of the work lies in the development of a new neural network architecture that integrates convolutional and recurrent components to improve the accuracy and efficiency of object detection in video sequences. The practical significance of the research consists in creating a system that can be applied in real conditions for surveillance tasks, document detection, and other objects, significantly improving the quality and reliability of these systems.

4 Dataset Modification

The YOLO model requires a specific format for object annotations on images, which necessitates providing four numerical values: the coordinates of the center, the width, and the height of the bounding box. All values must be normalized relative to the dimensions of the image. In accordance with these requirements, the dataset was transformed as follows:

- The annotation file for each image contains only 4 numbers: the coordinates of the center (x, y), width, height.
- All images are placed in one folder, annotations in another.
- The size of the images is reduced to 224x224.

Examples of images with bounding boxes:

Figure 4.1 – Examples of images with bounding boxes from the transformed dataset

5 Basic Model

The YOLOv3 architecture was implemented using PyTorch:

- A Python class was written for the network architecture.
- A class was developed for loading and processing the dataset in PyTorch.
- Prototyping of the main network components was carried out.

Table 5.1 – YOLOv3 Neural Network Architecture

#	Layer	Parameters	Activation Function	Output Size
0	Input	-	-	$3 \times 224 \times 224$
1	Conv	12 filters 5x5, stride 1x1, padding 2x2	SymReLU	$12 \times 224 \times 224$
2	MaxPool	2x2, stride 2x2	-	$12 \times 112 \times 112$
3	Conv	16 filters 5x5, stride 2x2, padding 1x1	SymReLU	$16 \times 56 \times 56$
4	Conv	16 filters 3x3, stride 2x2, padding 1x1	SymReLU	$16 \times 28 \times 28$
5	Conv	16 filters 3x3, stride 1x1, padding 1x1	SymReLU	$16 \times 28 \times 28$
6	MaxPool	2x2, stride 2x2	-	$16 \times 14 \times 14$
7	Conv	24 filters 3x3, stride 1x1, padding 1x1	SymReLU	$24 \times 14 \times 14$
8	Conv	48 filters 3x3, stride 2x2, padding 1x1	SymReLU	$48 \times 7 \times 7$
9	Conv	48 filters 3x3, stride 1x1, padding 1x1	SymReLU	$48 \times 7 \times 7$
10	Conv	48 filters 3x3, stride 1x1, padding 1x1	SymReLU	$48 \times 7 \times 7$
11	Conv	48 filters 3x3, stride 1x1, padding 1x1	SymReLU	$48 \times 7 \times 7$
12	Conv	64 filters 3x3, stride 1x1, padding 1x1	SymReLU	$64 \times 7 \times 7$
13	Conv	64 filters 7x7, stride 7x7, padding 0x0	SymReLU	$64 \times 1 \times 1$
14	Conv	4 filters 1x1, stride 1x1, padding 0x0	SymReLU	$4 \times 1 \times 1$
15	Conv	4 filters 3x3, stride 1x1, padding 1x1	SymReLU	$4 \times 1 \times 1$
16	Conv	4 filters 3x3, stride 1x1, padding 1x1	SymReLU	$4 \times 1 \times 1$
17	Upsample	scale_factor=(1, 1), mode='nearest'	-	$4 \times 1 \times 1$
18	Conv	96 filters 3x3, stride 1x1, padding 1x1	SymReLU	$96 \times 1 \times 1$
19	Conv	256 filters 1x1, stride 1x1, padding 0x0	SymReLU	$256 \times 1 \times 1$
20	Conv	4 filters 1x1, stride 1x1, padding 0x0	-	$4 \times 1 \times 1$
21	Flatten	start_dim=1	-	4

Convolution is the basic building block of convolutional neural networks. The convolution operation applies a filter (also known as a kernel) to the input image to extract features. The filter slides over the image, performing element-wise multiplication and summation, creating a feature map.

In the example of figure 5.3, convolution is applied to the input image (left), using a 3x3 filter to create the output image (right). Padding is also added to the edges of the input image, which is called padding.

Max pooling is used to reduce the dimensionality of the feature map and reduce computational costs, as well as to prevent overfitting. This operation selects the maximum value in each sub-array of a fixed size.

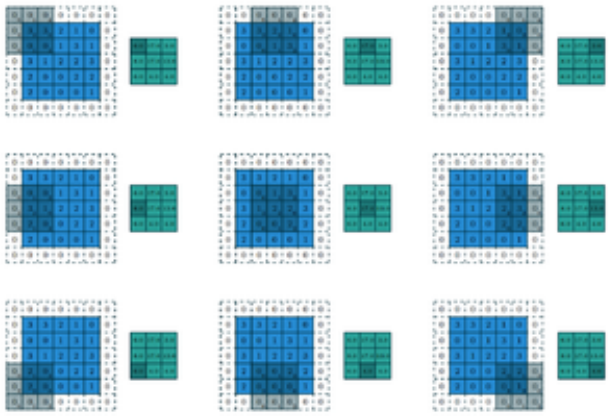


Figure 5.1 – Example of Convolution Operations

Upsampling is used to increase the dimensionality of the feature map. One of the most common upsampling methods is nearest neighbor, which duplicates the nearest values to increase the size. The nearest neighbor method is used in the proposed architecture.

This network architecture is quite lightweight, allowing for quick model training and experimentation. It also allows for operation on CPUs on low-powered devices. The activation function chosen was Symmetrical ReLU (SymReLU):

$$\text{SymReLU}(x) = \max(0, x) - \alpha \min(0, x)$$

where α is a parameter that determines the influence of negative values.

Unlike standard ReLU, which zeroes out all negative values, SymReLU allows negative values to be considered.

5.1 Results

The YOLOv3 model was configured and trained on the prepared data. Initial results were obtained, demonstrating non-trivial quality metrics. IoU was 0.77.

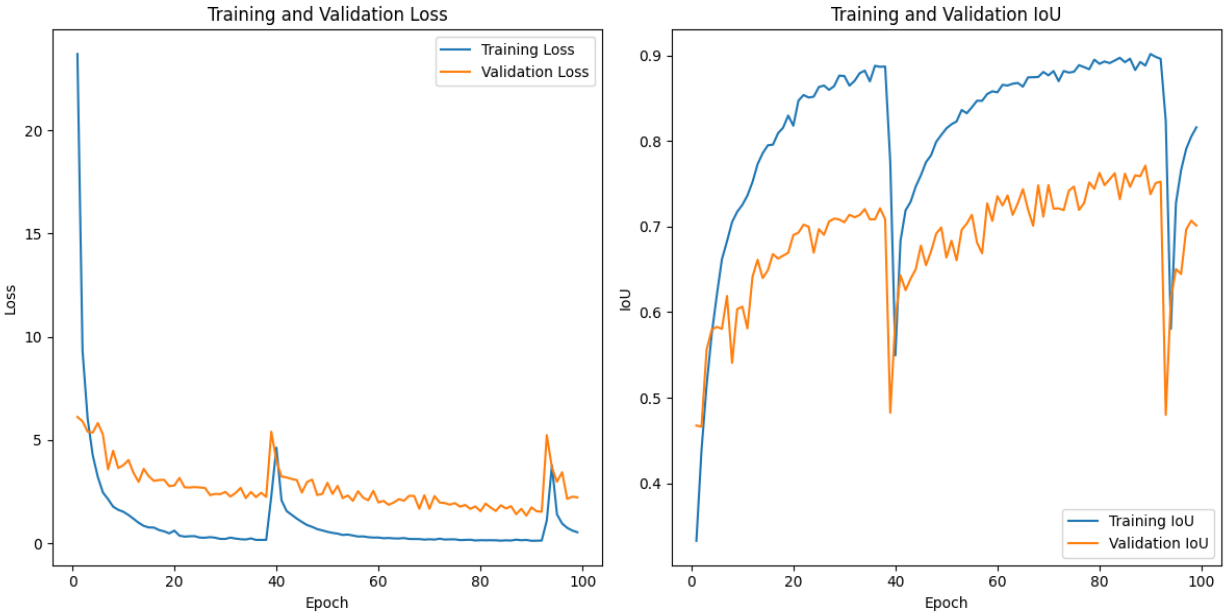


Figure 5.2 – Loss and Quality Function Values.

Figure 5.3 – Example of the Basic Architecture's Performance.

6 Implementation of Algorithmic Tracking

To improve the analysis of sequential frames, a tracking algorithm was developed that uses an additional metric - the model's confidence in the prediction. In addition to the bounding box coordinates, another value is fed into the model - confidence, which is initially set to 1, since the object is always present in the image.

During training, the model adjusts the weights so that the confidence at the output is as close to 1 as possible. This means that the model is confident in the correctness of its prediction. However, if something unusual appears in the image, with which the model has rarely encountered before, the level of confidence decreases. Thus, confidence serves as an indicator of the reliability of the model's predictions: the lower the confidence, the less confident the model is in the correctness of its predictions.

If the confidence is below a certain threshold, the algorithm provides the same response as for the previous frame. We work on the assumption that the object has not moved significantly within one frame.

6.1 Results

The IoU on validation was 0.76.

Figure 6.1 – Results in the task with algorithmic tracking.

7 Development of Recurrent YOLO

At this stage, changes were made to the neural network architecture. A GRU recurrent layer was added before the fully connected layer. As a result of these changes, the network processes a batch of data consisting of 8 consecutive video frames instead of processing each image separately.

7.1 Dataset Structure Modification

Previously, all images were stored in one folder. Now, there is a need to take 8 consecutive images. Therefore, the dataset was transformed as follows:

- For each sequence of 8 consecutive images, a separate folder is created.
- Each batch contains 16 such folders, and a minibatch represents one folder containing 8 images.

7.1.1 Advantages of the Changes

- **Sequential Frame Processing:** The network processes minibatches consisting of consecutive frames.
- **Temporal Dependency Accounting:** Temporal dependencies between frames are taken into account, improving object detection quality in videos.
- **Data Shuffling:** Proper dataset restructuring allows data shuffling when loading into batches, improving the training process as the model sees images in a random order within each batch.

Table 7.1 – Neural Network Architecture with Recurrent Layer

#	Layer	Parameters	Activation Function	Output Size
0	Input	-	-	$8 \times 3 \times 224 \times 224$
1	Conv	12 filters 5x5, stride 1x1, padding 2x2	SymReLU	$8 \times 12 \times 224 \times 224$
2	MaxPool	2x2, stride 2x2	-	$8 \times 12 \times 112 \times 112$
3	Conv	16 filters 5x5, stride 2x2, padding 1x1	SymReLU	$8 \times 16 \times 56 \times 56$
4	Conv	16 filters 3x3, stride 2x2, padding 1x1	SymReLU	$8 \times 16 \times 28 \times 28$
5	Conv	16 filters 3x3, stride 1x1, padding 1x1	SymReLU	$8 \times 16 \times 28 \times 28$
6	MaxPool	2x2, stride 2x2	-	$8 \times 16 \times 14 \times 14$
7	Conv	24 filters 3x3, stride 1x1, padding 1x1	SymReLU	$8 \times 24 \times 14 \times 14$
8	Conv	48 filters 3x3, stride 2x2, padding 1x1	SymReLU	$8 \times 48 \times 7 \times 7$
9	Conv	48 filters 3x3, stride 1x1, padding 1x1	SymReLU	$8 \times 48 \times 7 \times 7$
10	Conv	48 filters 3x3, stride 1x1, padding 1x1	SymReLU	$8 \times 48 \times 7 \times 7$
11	Conv	48 filters 3x3, stride 1x1, padding 1x1	SymReLU	$8 \times 48 \times 7 \times 7$
12	Conv	64 filters 3x3, stride 1x1, padding 1x1	SymReLU	$8 \times 64 \times 7 \times 7$
13	Conv	64 filters 7x7, stride 7x7, padding 0x0	SymReLU	$8 \times 64 \times 1 \times 1$
14	Conv	4 filters 1x1, stride 1x1, padding 0x0	SymReLU	$8 \times 4 \times 1 \times 1$
15	Conv	4 filters 3x3, stride 1x1, padding 1x1	SymReLU	$8 \times 4 \times 1 \times 1$
16	Conv	4 filters 3x3, stride 1x1, padding 1x1	SymReLU	$8 \times 4 \times 1 \times 1$
17	Upsample	scale_factor=(1, 1), mode='nearest'	-	$8 \times 4 \times 1 \times 1$
18	Conv	96 filters 3x3, stride 1x1, padding 1x1	SymReLU	$8 \times 96 \times 1 \times 1$
19	Conv	512 filters 1x1, stride 1x1, padding 0x0	SymReLU	$8 \times 512 \times 1 \times 1$
20	Conv	512 filters 1x1, stride 1x1, padding 0x0	-	$8 \times 512 \times 1 \times 1$
21	Flatten	start_dim=1	-	8×512
22	GRU	input_size=512, hidden_size=512, num_layers=1	-	$1 \times 8 \times 512$
23	FC	512 -¿ 5	-	$1 \times 8 \times 5$

7.2 Data Augmentation

To improve the model's accuracy, various data augmentations were added: blurring, noise, and rotations of images by 0, 90, or 180 degrees. Rotations at other angles were not performed. Below is an example of an image with augmentations.

The model was trained on these data. It was noted that the model found it more difficult to detect the object in images rotated by 90 degrees:

As a result, only rotations of 0 and 180 degrees were kept.

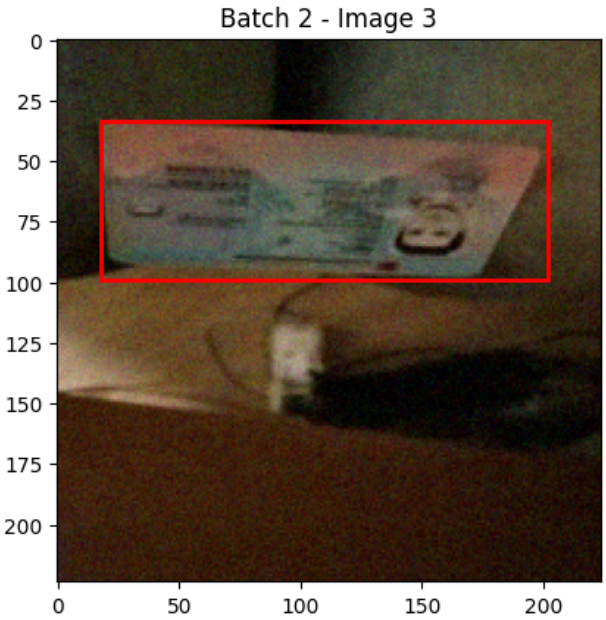


Figure 7.1 – Example of an image with augmentations

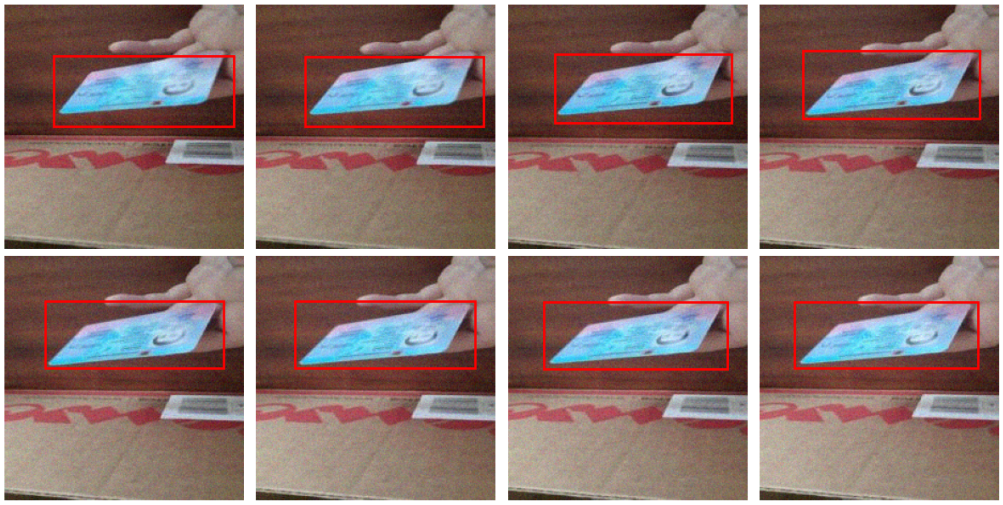


Figure 7.2 – Model response for a horizontal document.

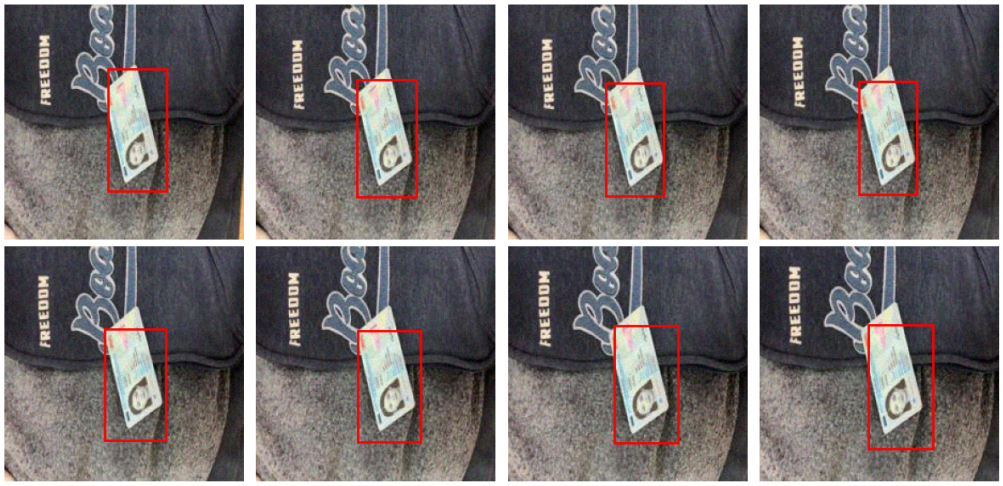


Figure 7.3 – Model response for a vertical document.

7.3 Results

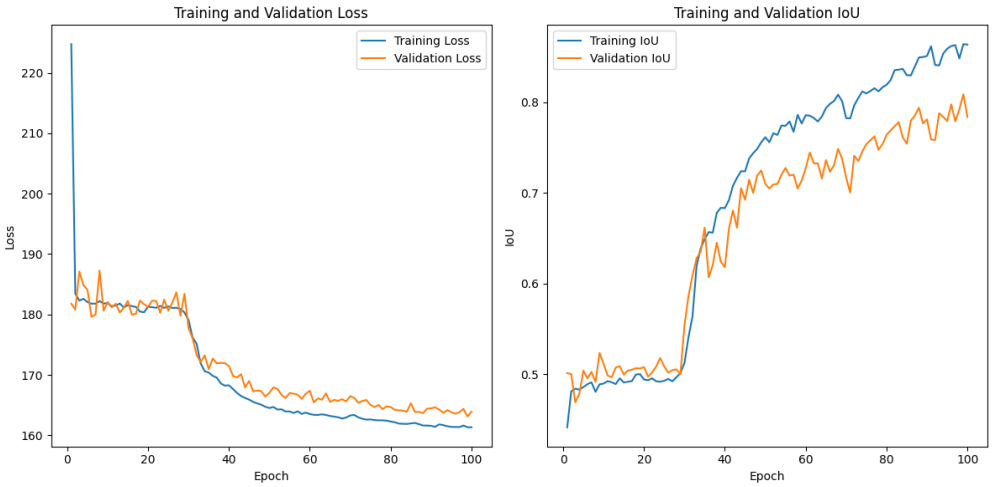


Figure 7.4 – Results of the model with the recurrent layer.

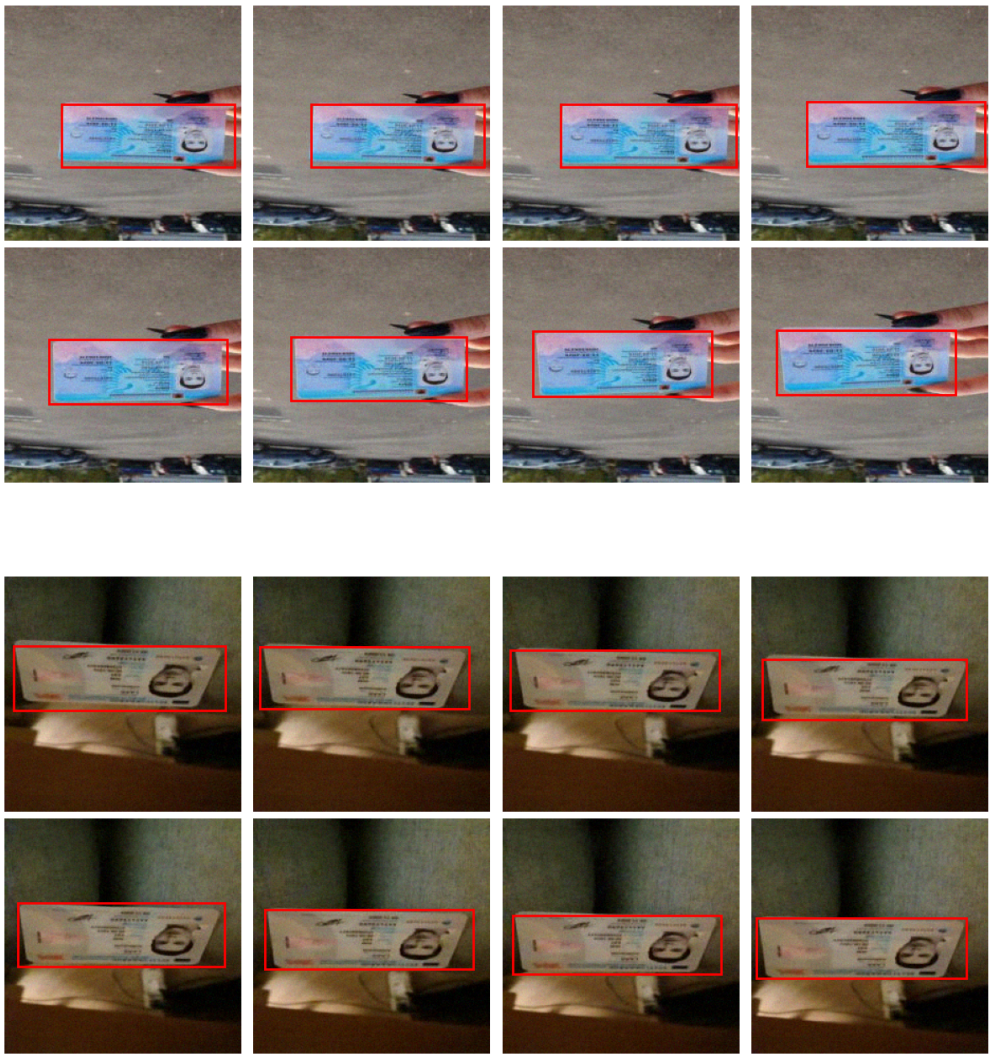


Figure 7.5 – Examples of the model with the recurrent layer.

8 Experiments

The dataset was incorrectly divided into training and validation parts: the first 80% of the images went to the training part, and the remaining 20% went to the validation part. As a result, the validation part consisted of images with strong projective distortions and glare from the sun or a lamp, since such images are at the end of the dataset.

Later, the correct division was performed: 80% of the images from all categories were included in the training part, and the remaining 20% were in the validation part. The results of the incorrect split can also be useful, as in real conditions it is sometimes impossible to get a perfect dataset, for example, when images are artificially generated or when the design of the object changes.

8.1 Incorrect Dataset Split

8.1.1 With Augmentations

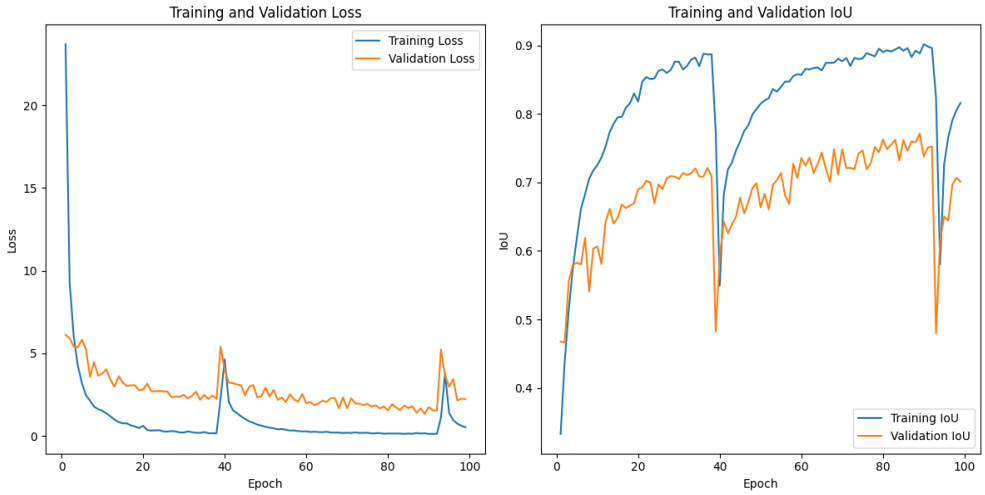


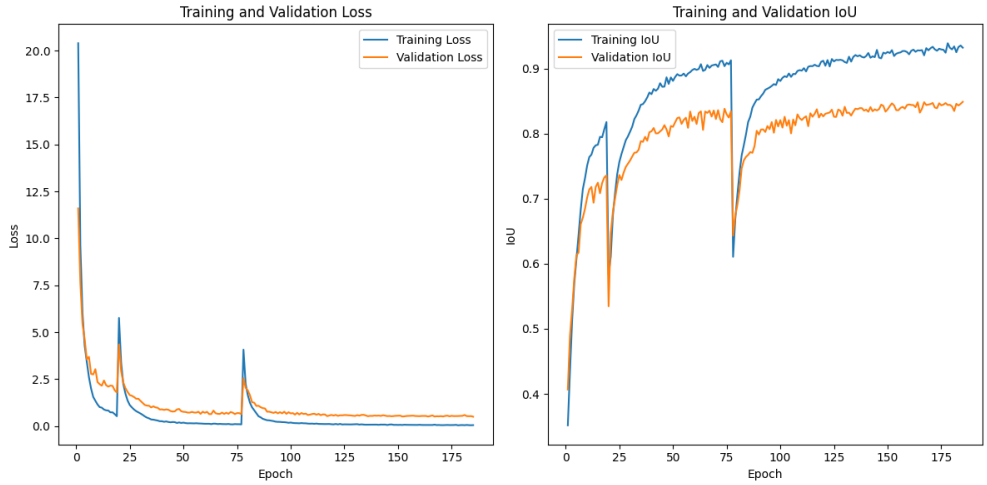
Figure 8.1 – Basic Architecture.

Figure 8.2 – Architecture with Algorithmic Tracking.

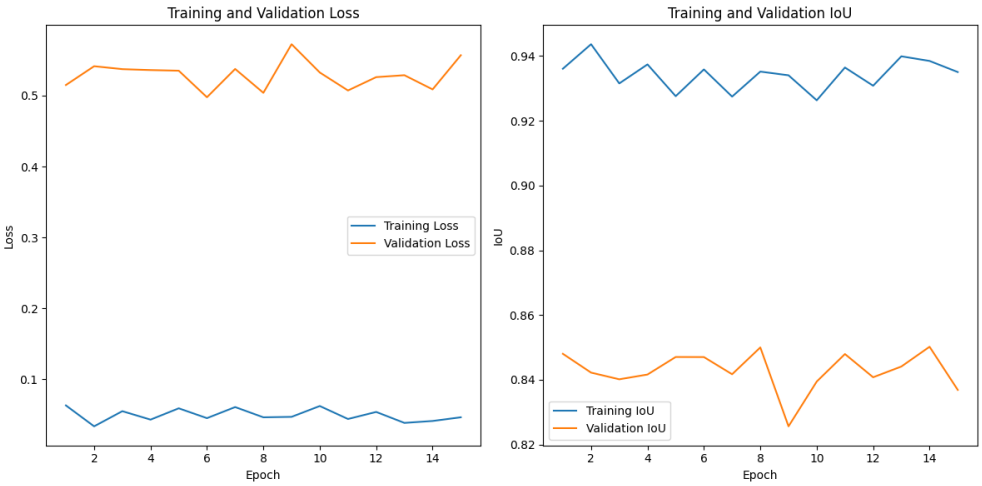
Figure 8.3 – Architecture with Recurrent Layer.

8.2 Correct Dataset Split

8.2.1 Without Augmentations



(a) Basic Architecture - 185 Epochs.



(b) Basic Architecture - Subsequent 15 Epochs.

Figure 8.5 – Architecture with Algorithmic Tracking.

Figure 8.6 – Architecture with Recurrent Layer.

8.2.2 With Augmentations

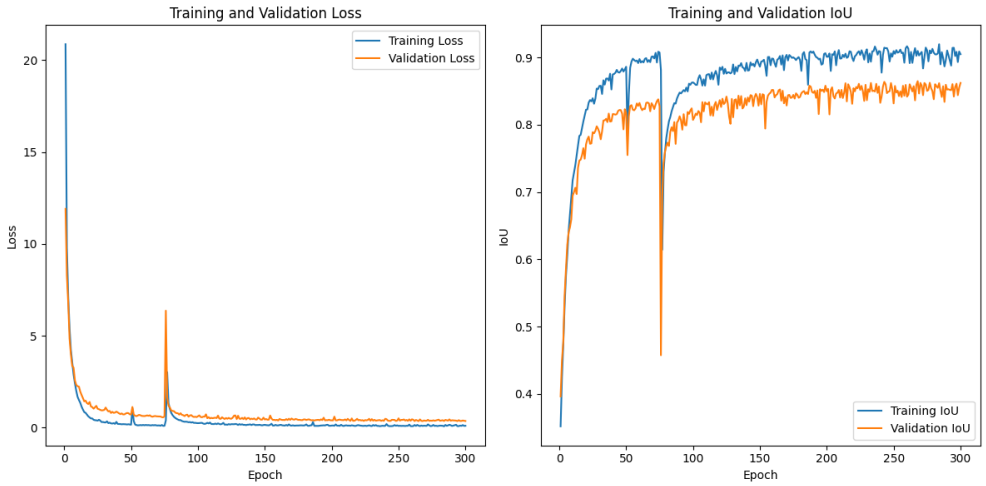


Figure 8.7 – Basic Architecture.

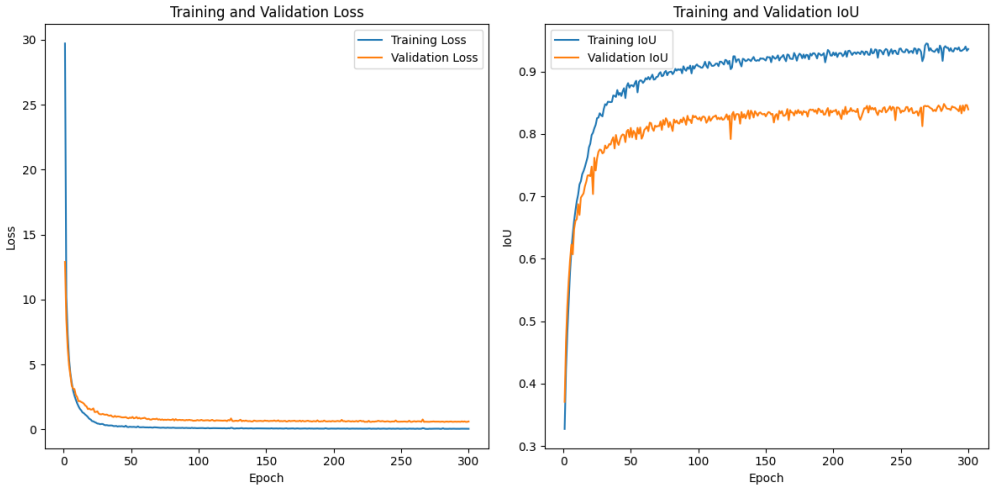


Figure 8.8 – Architecture with Algorithmic Tracking.

Figure 8.9 – Architecture with Recurrent Layer.

8.3 Overall Results

Table 8.1 – IoU Values for Different Architectures

	Incorrect Split	Correct Split	
	With Augmentations	Without Augmentations	With Augmentations
Basic	0.77	0.85	0.86
Tracking	0.76	0.85	0.85
Recurrent	0.81	0.83	0.88

It is also possible to look at the results in specific examples. Below are examples of the performance of the basic model and the model with a recurrent layer with augmentations and the correct dataset split.

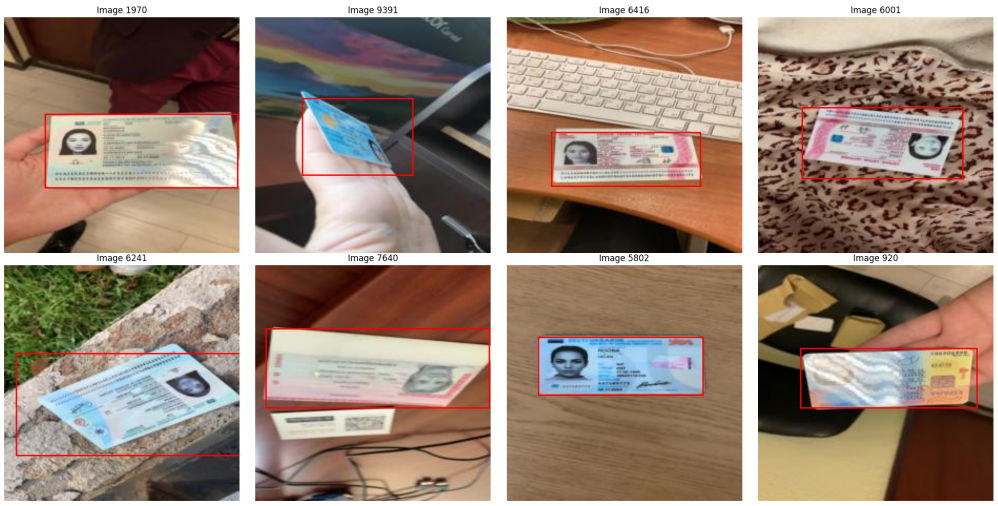


Figure 8.10 – Example of Basic Architecture Performance.

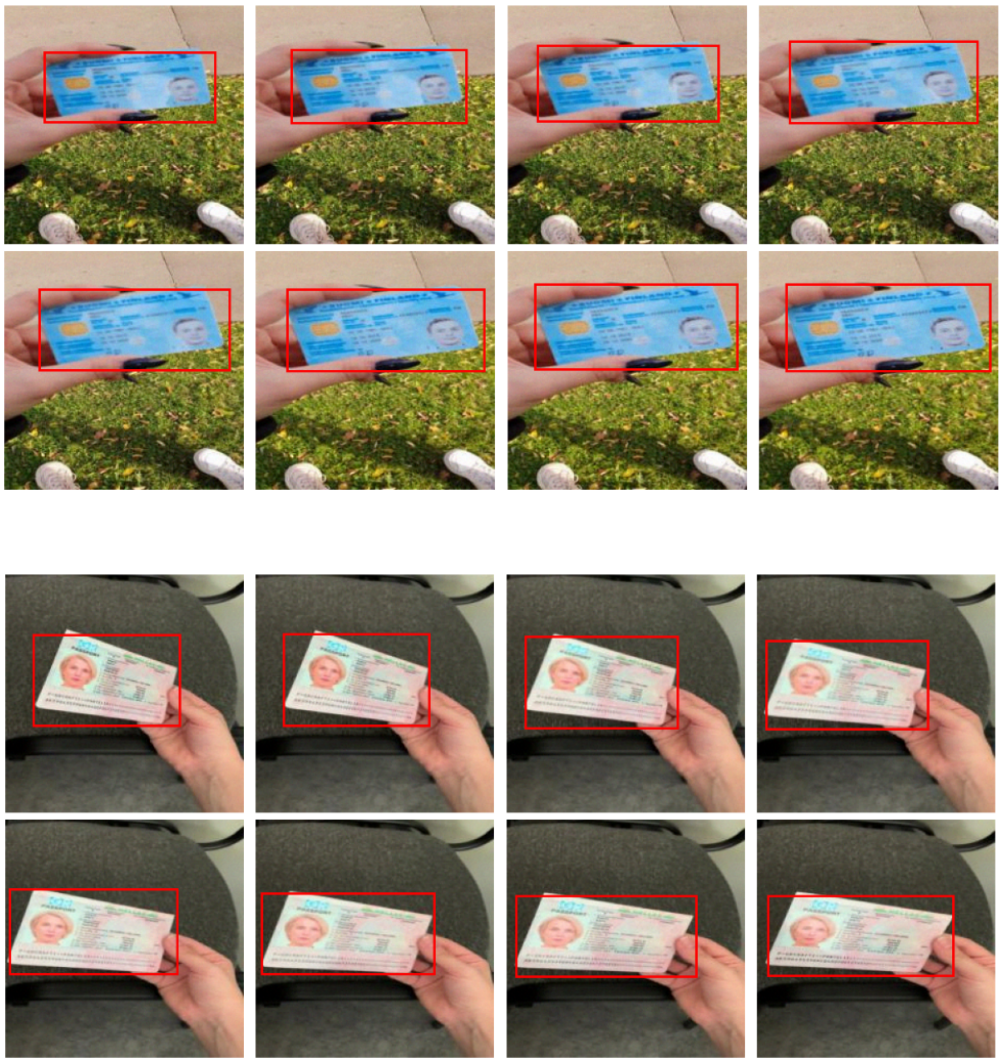


Figure 8.11 – Examples of Performance of the Model with a Recurrent Layer.

9 Conclusion

The following tasks were accomplished as part of the thesis:

1. A detailed literature review was conducted, covering modern object detection methods and the use of convolutional and recurrent neural networks.
2. The YOLO architecture was developed and tested, demonstrating high accuracy in object detection.
3. Algorithmic tracking was implemented, which calculates the model's confidence in its response and provides the response for the previous frame if the confidence is low.
4. A recurrent layer was integrated into YOLO to account for temporal dependencies between frames, which improved detection quality.
5. Experiments were conducted under various conditions.
6. The recurrent network architecture showed better results compared to other models.

Conclusion

In the task of document detection in video with augmentations, adding a recurrent layer to the YOLO model improves detection.

Ideas for Further Research

- Conduct experiments on other datasets.
- Experiment with augmentations and hyperparameters.
- Conduct experiments with more recent YOLO models.

References

[1] Redmon, J., Divvala, S., Girshick, R., Farhadi, A. You only look once: Unified, real-time object detection // Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. P. 779-788.

[2] Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L. Large-scale video classification with convolutional neural networks // Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

[3] Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T. Long-term recurrent convolutional networks for visual recognition and description // arXiv preprint arXiv:1411.4389. 2015.

[4] Qiao, S., Chen, L. C., Yuille, A. DetectoRS: Detecting Objects with Recursive Feature Pyramid and Switchable Atrous Convolution // arXiv preprint arXiv:2006.02334. 2020.

[5] Arlazarov, V.V., Zhukovsky, A.E., Krivtsov, V.E., Nikolaev, D.P., Polevoy, D.V. Analysis of the features of using stationary and mobile small-sized digital cameras for document recognition // Problems of Information Technology. 2016. Vol. 16, No. 2. P. 34-42.

[6] Arlazarov, V.L., Kuratov, P.A., Loginov, A.S., Slavin, O.A. Algorithms for searching printed character boundaries used in optical character recognition // Problems of Information Technology. 2019. Vol. 17, No. 3. P. 56-67.

[7] Hartley, R., Zisserman, A. Multiple view geometry in computer vision. Cambridge: Cambridge University Press, 2003.

[8] Bochkovskiy, A., Wang, C. Y., Liao, H. Y. M. YOLOv4: Optimal speed and accuracy of object detection // arXiv preprint arXiv:2004.10934. 2020.

[9] Zhang, J., Zhang, H., Liu, B., Qu, G., Wang, F., Zhang, H., Shi, X. Small object intelligent detection method based on adaptive recursive feature pyramid // Heliyon. 2023. Vol. 9, No. 5. e09863.

[10] Smart Engines. Habr. URL: https://habr.com/en/companies/smartengines/articles/714250/ (accessed: 15.09.2023).