UDC 004.85

Development of a Fire Detection Application

Cherkashin Alexander Mihailovich
Sholom-Aleichem Priamursky State University
student

Abstract
In this article, the process of training a neural network for fire detection is described. The work used the Python programming language and the Yolo library, as well as a dataset with ready-made annotations for fire detection. As a result, a single-class model was trained for fire detection, and metrics and model evaluations were also included.
Keywords: Python, Yolo, detection, computer vision.

Разработка приложения для распознавания огня

Черкашин Александр Михайлович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье описан процесс обучения нейронной сети для распознавания огня. В работе использовались язык программирования Python и библиотека Yolo, а также набор данных с готовыми разметками для распознавания огня. В результате работы была обучена модель одного класса для распознавания огня, а также включены метрики и оценки модели.
Ключевые слова: Python, Yolo, распознавания, компьютерная зрения.

1 Introduction

1.1. Relevance of the Research

The relevance of the research lies in the fact that studying emergency situations using neural networks can help develop effective methods for determining and predicting such situations, ensuring the safety and protection of the population. This can be useful for emergency management agencies, rescue services, and analytical centers.

1.2. Research Objective

The objective of this work is to train a model of artificial intelligence for fire detection.

1.3. Research Review

The study by H. Yai and J. Ebert presents a method based on computer vision for fire detection in color video sequences. The authors propose using computer vision algorithms to analyze color videos and identify fire features. Their method allows for automatic fire detection in videos, which can be useful for rapid response to emergency situations and preventing fire damage [1].

The study by S. S., M. Mougiakou, G. G. Samatas, and G. A. Papakostas focuses on the application of computer vision for fire detection on unmanned aerial vehicles (UAVs). They investigate the transition from software to hardware for fire detection using computer vision. The authors propose methods that enable UAVs to automatically detect fires using specialized equipment and software. Their study has the potential to improve the speed of fire detection from aerial means and enhance firefighting efficiency [2].

The study by A. E. Setin et al. reviews methods for fire detection in videos. They conducted a review of existing approaches to fire detection in videos and identified various techniques and methods used in this field. The authors examined both classical and modern approaches to fire detection in videos, as well as evaluated their advantages and disadvantages. Their study provides valuable information for security and fire safety professionals, and can serve as a basis for developing more effective fire detection systems in videos [3].

The study by P. Chamoso et al. focuses on developing a computer vision system for fire detection and reporting using UAVs. The authors propose a method that allows UAVs to automatically detect fires using a computer vision system and generate detailed reports on the situation. Their study has the potential to improve the speed of fire detection from aerial means and provide operational information for firefighting. Developing such a system can be an important step in enhancing firefighting efficiency and emergency management [4].

The study by B. U. Töreyin et al. focuses on developing a computer vision-based method for real-time fire and flame detection. The authors propose using computer vision algorithms for continuous monitoring and detection of fires and flames in real-time. Their method allows for rapid and effective response to fires, which can be critically important for preventing damage and ensuring safety. The study makes a significant contribution to the field of fire safety and can find applications in various areas where rapid fire detection is required [5].

The study by A. Khondaker, A. Khondaker, and J. Uddin focuses on developing a method for early fire detection based on computer vision using improved chromatic segmentation and optical flow analysis. The authors propose a combined approach that integrates improved chromatic segmentation and optical flow analysis for more effective fire detection at early stages. Their method allows for rapid response to fires, which can significantly reduce response time and damage. The study makes an important contribution to the field of fire safety and can be useful for developing more effective fire detection systems [6].

2. Practical Part

The 'train' directory contains 3527 images of random sizes and 3522 annotations. Each annotation contains information about the location of the region in the form of 4 vectors in the format from 0 to 1, representing relative image coordinates. At the beginning of each new line, the index 0 is indicated, marking the 'fire' label (Fig. 1).

For example, the annotation val/labels/01ac2c5a9adf2e6e.txt

```
0 0.535156 0.595760 0.164062 0.168129
```

The directory is divided into two parts: 'train' for training the neural network and 'val' for testing and validating the model's predictions. Inside the directory, there are folders 'images' containing images (Fig. 2.1) and 'labels' containing annotation data indicating the fire detection regions. The 'val' folder contains 150 images and 150 annotations.

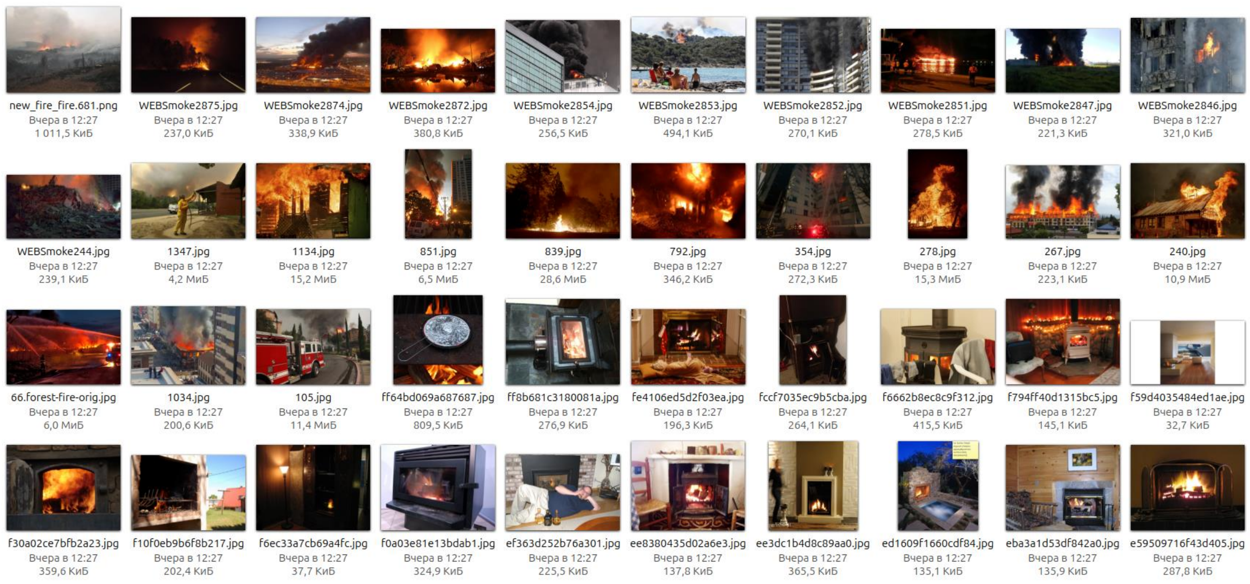
new_fire_fire.681.png Вчера в 12:27 1 011,5 КиБ
WEBSmoke2875.jpg Вчера в 12:27 237,0 КиБ
WEBSmoke2874.jpg Вчера в 12:27 338,9 КиБ
WEBSmoke2872.jpg Вчера в 12:27 380,8 КиБ
WEBSmoke2854.jpg Вчера в 12:27 256,5 КиБ
WEBSmoke2853.jpg Вчера в 12:27 494,1 КиБ
WEBSmoke2852.jpg Вчера в 12:27 270,1 КиБ
WEBSmoke2851.jpg Вчера в 12:27 278,5 КиБ
WEBSmoke2847.jpg Вчера в 12:27 221,3 КиБ
WEBSmoke2846.jpg Вчера в 12:27 321,0 КиБ
WEBSmoke244.jpg Вчера в 12:27 239,1 КиБ
1347.jpg Вчера в 12:27 4,2 МиБ
1134.jpg Вчера в 12:27 15,2 МиБ
851.jpg Вчера в 12:27 6,5 МиБ
839.jpg Вчера в 12:27 28,6 МиБ
792.jpg Вчера в 12:27 346,2 КиБ
354.jpg Вчера в 12:27 272,3 КиБ
278.jpg Вчера в 12:27 15,3 МиБ
267.jpg Вчера в 12:27 223,1 КиБ
240.jpg Вчера в 12:27 10,9 МиБ
66.forest-fire-orig.jpg Вчера в 12:27 6,0 МиБ
1034.jpg Вчера в 12:27 200,6 КиБ
105.jpg Вчера в 12:27 11,4 МиБ
ff64bd069a687687.jpg Вчера в 12:27 809,5 КиБ
ff8b681c3180081a.jpg Вчера в 12:27 276,9 КиБ
fe4106ed5d2f03ea.jpg Вчера в 12:27 196,3 КиБ
fccf7035ec9b5cba.jpg Вчера в 12:27 264,1 КиБ
f6662b8ec8c9f312.jpg Вчера в 12:27 415,5 КиБ
f794ff40d1315bc5.jpg Вчера в 12:27 145,1 КиБ
f59d4035484ed1ae.jpg Вчера в 12:27 32,7 КиБ
f30a02ce7bfb2a23.jpg Вчера в 12:27 359,6 КиБ
f10f0eb9b6f8b217.jpg Вчера в 12:27 202,4 КиБ
f6ec33a7cb69a4fc.jpg Вчера в 12:27 37,7 КиБ
f0a03e81e13bdab1.jpg Вчера в 12:27 324,9 КиБ
ef363d252b76a301.jpg Вчера в 12:27 225,5 КиБ
ee8380435d02a6e3.jpg Вчера в 12:27 137,8 КиБ
ee3dc1b4d8c89aa0.jpg Вчера в 12:27 365,5 КиБ
ed1609f1660cdf84.jpg Вчера в 12:27 135,1 КиБ
eba3a1d53df842a0.jpg Вчера в 12:27 135,9 КиБ
e59509716f43d405.jpg Вчера в 12:27 287,8 КиБ

Figure 1. Dataset of images in the 'train' directory.

The source code was taken from [7] the yolov5-fire-detection project, which we ran and used to train the model by executing the command:

```
python3 train_dual.py --workers 4 --device 0 --batch 8 --data ../fire.yaml --img 640 --cfg models/detect/yolov9-c.yaml --weights '' --name yolov9-c --hyp hyp.scratch-high.yaml --min-items 0 --epochs 50 --close-mosaic 15
```

Table 1. Model Training Cycle, Object Detection, Classification, and DFL Loss Function

epoch	train/box_loss	train/cls_loss	train/dfl_loss
0	4.823	5.2688	5.5014
1	4.4994	4.3322	4.9547
2	3.667	3.4889	3.8901
3	3.1495	3.0431	3.1826
4	2.8886	2.8312	2.8504
5	2.7362	2.6631	2.6381
6	2.6327	2.5817	2.541
7	2.5424	2.5042	2.46

8	2.5032	2.4557	2.3908
9	2.4521	2.3881	2.3344
10	2.4161	2.3331	2.3124
11	2.3813	2.2934	2.2678
12	2.3628	2.2642	2.2419
13	2.327	2.2314	2.2192
14	2.3111	2.2112	2.2107
15	2.3002	2.163	2.1866
16	2.2617	2.1506	2.1654
17	2.2454	2.1162	2.1452
18	2.2328	2.1223	2.1463
19	2.2096	2.0925	2.1357
20	2.2027	2.0666	2.0994
21	2.1969	2.0459	2.1048
22	2.1697	2.0207	2.0959
23	2.1588	1.9833	2.0847
24	2.1494	1.9553	2.0576
25	2.1347	1.9547	2.0564
26	2.1153	1.9334	2.0389
27	2.0965	1.9104	2.0334
28	2.0895	1.895	2.0143
29	2.0776	1.8859	2.011
30	2.077	1.8773	2.0153
31	2.0585	1.8334	2.001
32	2.0452	1.8302	2.0003
33	2.0334	1.8094	1.9858
34	2.0116	1.7916	1.9738
35	1.9889	1.7576	1.9943
36	1.9649	1.7167	2.0149
37	1.9539	1.7002	1.9873
38	1.9399	1.6647	1.9738
39	1.9362	1.6398	1.9685
40	1.9116	1.617	1.9541

41	1.8957	1.5936	1.9405
42	1.8658	1.5388	1.9168
43	1.8631	1.538	1.9254
44	1.8369	1.5067	1.8959
45	1.8216	1.4884	1.899
46	1.794	1.4543	1.8788
47	1.8033	1.4457	1.8788
48	1.7916	1.4492	1.8583
49	1.7626	1.4014	1.8483

Table 2. Model Evaluation Metrics on the Training Dataset

epoch	metrics/precision	metrics/recall	metrics/mAP_0.5	metrics/mAP_0.5:0.95
0	0.011539	0.082031	0.0051703	0.0013739
1	0.063978	0.082031	0.029623	0.0084217
2	0.087793	0.16797	0.051857	0.017441
3	0.1825	0.35938	0.14001	0.046879
4	0.37182	0.31676	0.28535	0.10248
5	0.3954	0.375	0.34054	0.13525
6	0.43161	0.33203	0.31764	0.1325
7	0.5177	0.4375	0.41882	0.17734
8	0.39752	0.34766	0.2855	0.1148
9	0.4247	0.40661	0.36829	0.15922
10	0.52363	0.41406	0.38302	0.17197
11	0.42389	0.42188	0.37049	0.16451
12	0.41898	0.38281	0.34741	0.15302
13	0.44008	0.45312	0.44658	0.18982
14	0.51718	0.44922	0.4463	0.19577
15	0.47925	0.42188	0.42056	0.18857
16	0.52005	0.39062	0.42444	0.18612
17	0.51268	0.41507	0.44299	0.20017
18	0.48829	0.43359	0.39115	0.18648
19	0.43527	0.46484	0.43549	0.20394
20	0.50035	0.42188	0.40464	0.19323
21	0.58169	0.40625	0.42942	0.20731

22	0.47172	0.51562	0.47053	0.21829
23	0.51952	0.44531	0.45146	0.19951
24	0.6034	0.39844	0.48934	0.23731
25	0.51776	0.42188	0.45	0.20665
26	0.53957	0.46484	0.45138	0.20326
27	0.48444	0.47717	0.44943	0.20929
28	0.57517	0.44922	0.44819	0.21164
29	0.52486	0.49219	0.45937	0.21285
30	0.46531	0.53031	0.49244	0.21879
31	0.51783	0.47266	0.47028	0.22866
32	0.59466	0.44126	0.47249	0.22654
33	0.4774	0.48828	0.46262	0.22299
34	0.64001	0.37502	0.44367	0.2149
35	0.52939	0.43502	0.4407	0.20693
36	0.50658	0.51953	0.50224	0.23411
37	0.55323	0.49823	0.50031	0.22689
38	0.53259	0.49609	0.49235	0.23174
39	0.46303	0.54297	0.4872	0.23493
40	0.50665	0.50391	0.47995	0.23158
41	0.49863	0.47396	0.46456	0.225
42	0.48403	0.48047	0.43576	0.21101
43	0.49092	0.47266	0.43889	0.20946
44	0.4824	0.50781	0.44257	0.21943
45	0.56401	0.44973	0.4729	0.22909
46	0.55541	0.48047	0.45677	0.22224
47	0.51185	0.47266	0.45717	0.21914
48	0.52332	0.47173	0.45538	0.21988
49	0.54729	0.49609	0.46379	0.22665

Table 3. Model Training Speed

epoch	x/lr0	x/lr1	x/lr2
0	0.070068	0.0033258	0.0033258
1	0.039936	0.0065273	0.0065273
2	0.0096723	0.0095967	0.0095967

3	0.009406	0.009406	0.009406
4	0.009406	0.009406	0.009406
5	0.009208	0.009208	0.009208
6	0.00901	0.00901	0.00901
7	0.008812	0.008812	0.008812
8	0.008614	0.008614	0.008614
9	0.008416	0.008416	0.008416
10	0.008218	0.008218	0.008218
11	0.00802	0.00802	0.00802
12	0.007822	0.007822	0.007822
13	0.007624	0.007624	0.007624
14	0.007426	0.007426	0.007426
15	0.007228	0.007228	0.007228
16	0.00703	0.00703	0.00703
17	0.006832	0.006832	0.006832
18	0.006634	0.006634	0.006634
19	0.006436	0.006436	0.006436
20	0.006238	0.006238	0.006238
21	0.00604	0.00604	0.00604
22	0.005842	0.005842	0.005842
23	0.005644	0.005644	0.005644
24	0.005446	0.005446	0.005446
25	0.005248	0.005248	0.005248
26	0.00505	0.00505	0.00505
27	0.004852	0.004852	0.004852
28	0.004654	0.004654	0.004654
29	0.004456	0.004456	0.004456
30	0.004258	0.004258	0.004258
31	0.00406	0.00406	0.00406
32	0.003862	0.003862	0.003862
33	0.003664	0.003664	0.003664
34	0.003466	0.003466	0.003466
35	0.003268	0.003268	0.003268

36	0.00307	0.00307	0.00307
37	0.002872	0.002872	0.002872
38	0.002674	0.002674	0.002674
39	0.002476	0.002476	0.002476
40	0.002278	0.002278	0.002278
41	0.00208	0.00208	0.00208
42	0.001882	0.001882	0.001882
43	0.001684	0.001684	0.001684
44	0.001486	0.001486	0.001486
45	0.001288	0.001288	0.001288
46	0.00109	0.00109	0.00109
47	0.000892	0.000892	0.000892
48	0.000694	0.000694	0.000694
49	0.000496	0.000496	0.000496

Epoch — epoch
train/box_loss — object detection loss function
train/cls_loss — object classification loss function
train/dfl_loss — DFL (Dual Focal Loss) loss function

metrics/precision, metrics/recall, metrics/mAP_0.5, metrics/mAP_0.5:0.95 - Model evaluation metrics on the training dataset, including precision, recall, mean average precision (mAP) at IoU 0.5, and mean average precision in the range from 0.5 to 0.95 IoU.

metrics/precision - Precision (accuracy) - the ratio of correctly predicted objects (True Positives) to all objects.

metrics/recall - Recall (completeness) - the ratio of correctly predicted objects (True Positives) to all actual objects in the dataset.

metrics/mAP_0.5 - Mean Average Precision (mAP) at IoU 0.5

metrics/mAP_0.5:0.95 - Mean Average Precision (mAP) in the range from 0.5 to 0.95 IoU.

The values of the val/box_loss, val/cls_loss, and val/dfl_loss metrics during model training were 0, since the validation set was not used in this dataset.

x/lr0 - refers to the initial learning rate at the first stage of model training.
x/lr1 - refers to the learning rate at the second stage of model training.
x/lr2 - refers to the learning rate at the third stage of model training.

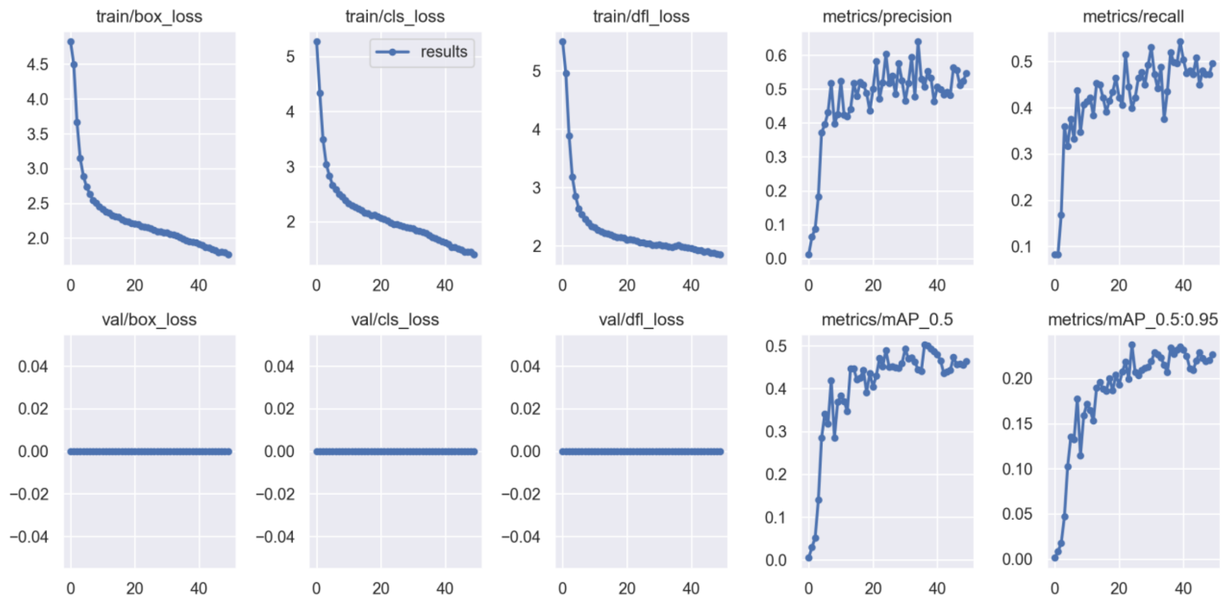

Figure 2. Model Evaluation Metrics on the Training Dataset

As a result of training the model (Fig. 2) for 50 epochs, the model significantly improved in fire detection (see Tables 1, 2, and 3).

After training the YOLOv9 model, two models were saved: best.pt and last.pt. Unlike them, best.pt corresponds to the best model achieved during training.

To test the model on real data, we ran a program that detected video sequences and outputted video with detected regions, as well as a threshold value for precision matching.

```
python detect.py --source "../Video/*" --weights ./runs/train/yolov9-c6/weights/best.pt
```

Program Output

```
video 11/11 (9539/9548) fire-detection/yolov5-fire-detection/Video/Yosemite Forest Fire Time Lapse and Flyover [bhvjfC5qenQ].mp4: 384x640 (no detections), 78.3ms
video 11/11 (9540/9548) fire-detection/yolov5-fire-detection/Video/Yosemite Forest Fire Time Lapse and Flyover [bhvjfC5qenQ].mp4: 384x640 (no detections), 78.2ms
video 11/11 (9541/9548) fire-detection/yolov5-fire-detection/Video/Yosemite Forest Fire Time Lapse and Flyover [bhvjfC5qenQ].mp4: 384x640 (no detections), 78.2ms
video 11/11 (9542/9548) fire-detection/yolov5-fire-detection/Video/Yosemite Forest Fire Time Lapse and Flyover [bhvjfC5qenQ].mp4: 384x640 (no detections), 78.2ms
video 11/11 (9543/9548) fire-detection/yolov5-fire-detection/Video/Yosemite Forest Fire Time Lapse and Flyover [bhvjfC5qenQ].mp4: 384x640 (no detections), 79.4ms
video 11/11 (9544/9548) fire-detection/yolov5-fire-detection/Video/Yosemite Forest Fire Time Lapse and Flyover [bhvjfC5qenQ].mp4: 384x640 (no detections), 78.3ms
video 11/11 (9545/9548) fire-detection/yolov5-fire-detection/Video/Yosemite Forest Fire Time Lapse and Flyover [bhvjfC5qenQ].mp4: 384x640 (no detections), 78.2ms
video 11/11 (9546/9548) fire-detection/yolov5-fire-detection/Video/Yosemite Forest Fire Time Lapse and Flyover [bhvjfC5qenQ].mp4: 384x640 (no detections), 78.2ms
video 11/11 (9547/9548) fire-detection/yolov5-fire-detection/Video/Yosemite Forest Fire Time Lapse and Flyover [bhvjfC5qenQ].mp4: 384x640 (no detections), 78.3ms
video 11/11 (9548/9548) fire-detection/yolov5-fire-detection/Video/Yosemite Forest Fire Time Lapse and Flyover [bhvjfC5qenQ].mp4: 384x640 (no detections), 78.3ms
Speed: 0.4ms pre-process, 71.7ms inference, 0.5ms NMS per image at shape (1, 3, 640, 640)
Results saved to runs/detect/exp11
```

As a result of running the program, a total of 11 videos and 49,483 frames were processed, with a frame size of 384x640 and approximately 78 ms.

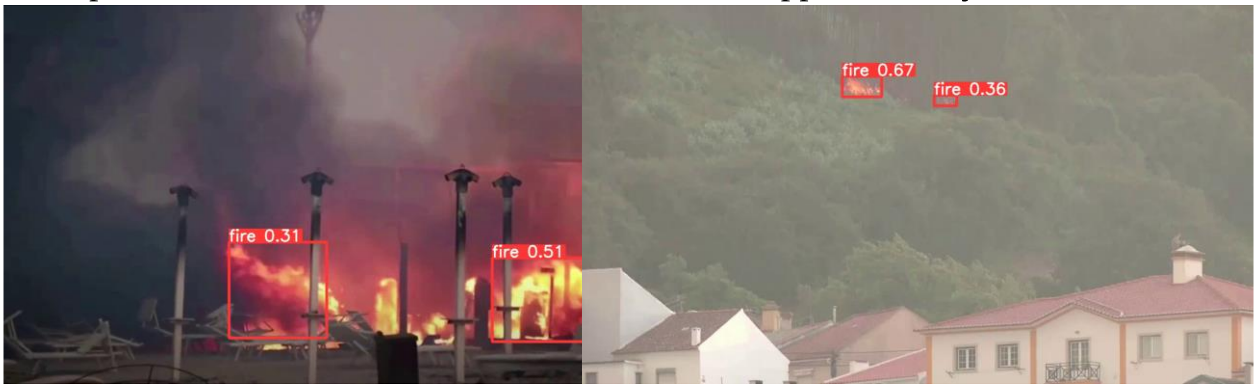

Figure 3. Fire in the forest

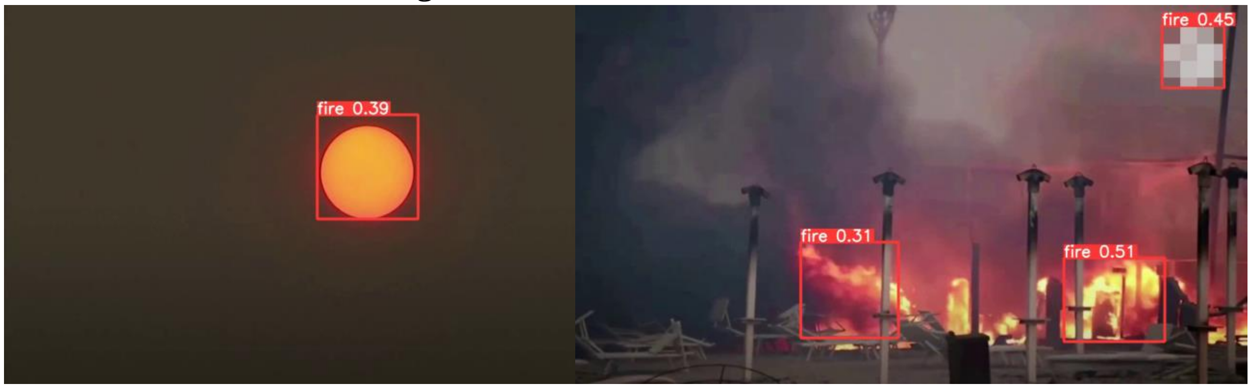

Figure 4. With the left model, the sun recognizes as a fire, and on the right it recognizes the fire, in both sides the logo recognizes as a fire by mistakes

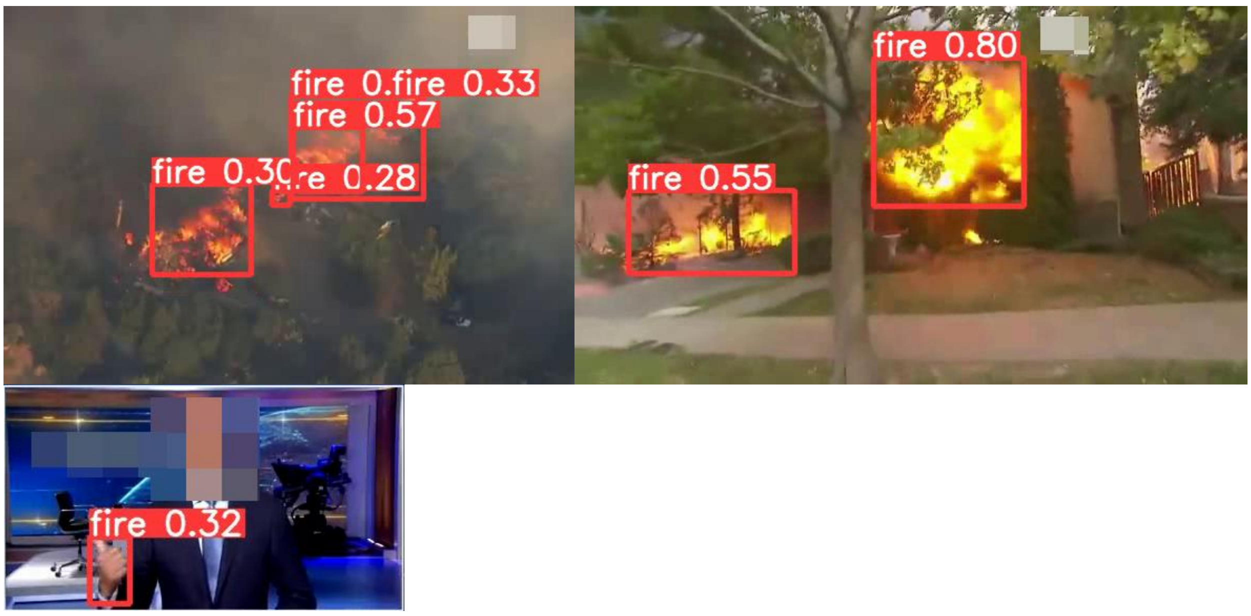

Figure 5. Recognition of fires in the forest in California, below the model mistakenly recognizes a person’s hand as a fire

3 Conclusions

During the practical part of the study, artificial intelligence models were developed and tested for automatic fire recognition. The results showed the average accuracy of recognition and good application productivity, which confirms its potential for use in real conditions.

Bibliographic list

1. Qi X., Ebert J. A computer vision based method for fire detection in color videos //International journal of imaging. 2009. T. 2. №. S09. C. 22-34.
2. Moumgiakmas S. S., Samatas G. G., Papakostas G. A. Computer vision for fire detection on UAVs—From software to hardware //Future Internet. 2021. T. 13. №. 8. C. 200.
3. Çetin A. E. et al. Video fire detection–review //Digital Signal Processing. 2013. T. 23. №. 6. C. 1827-1843.
4. Chamoso P. et al. Computer vision system for fire detection and report using UAVs //RSFF. 2018. C. 40-49.
5. Töreyin B. U. et al. Computer vision based method for real-time fire and flame detection //Pattern recognition letters. 2006. T. 27. №. 1. C. 49-58.
6. Khondaker A., Khandaker A., Uddin J. Computer vision-based early fire detection using enhanced chromatic segmentation and optical flow analysis technique //Int. Arab J. Inf. Technol. 2020. T. 17. №. 6. C. 947-953.
7. GitHub - spacewalk01/yolov5-fire-detection: Training YOLOv5/YOLOv9 to detect fire in a video // GitHub URL: https://github.com/spacewalk01/yolov5-fire-detection.git (date of submission: 2024-05-21).