


Electric Pylon Detection in Various Environmental Conditions Through High-Resolution Satellite Images

Natalia Denisenko¹, Dmitrii Shadrin² and Svetlana Illarionova²

¹*Northwestern university, 633 Clark St, Evanston, Illinois, USA*

²*Skolkovo Institute of Science and Technology, 143026, Bolshoy Bulvar 42, bldg. 1, Skolkovo, Moscow, Russia*
d.shadrin@skoltech.ru, s.illarionova@skoltech.ru

Keywords: Object detection; satellite imagery; augmentation; computer vision.

Abstract: Earth remote sensing data can be applied to detect and assess the condition of infrastructure objects on vast territories. One such object is electric pylons, which ensure the sustainability of the energy supply in rural and urban areas. In some remote regions, power lines can be damaged by natural hazards such as earthquakes, strong winds, or floods. Currently, the main limitation in developing highly effective algorithms for electric utilities assessment is associated with data availability and diverse environmental conditions. Therefore, in this study, we aim to explore solutions for new study territories with various backgrounds and forms of electric pylons. We examined several detection algorithms from the YOLO-family. The study includes experiments with datasets for Chinese territories and additionally collected data for regions in Russia. We managed to improve the initial score for polygon detection, achieving an mAP of 79.8%. The obtained results demonstrate high potential for power lines assessment and damage detection through satellite data and deep learning algorithms.

1 INTRODUCTION

Object detection is a fundamental task involving the localization and classification of objects within an image. This diverse field can be categorized into two distinct subtypes: object-centric and non-object-centric. The former involves data in which each image contains a single object, whereas the latter deals with instances in which multiple objects may be included in a single image. There are a broad number of datasets within the first category such as MNIST (MNIST, 2019) and ImageNet (ImageNet 2017). Contrariwise, non-object-centric datasets, typified by the widely recognized COCO (COCO, 2021), often feature backgrounds alongside multiple objects belonging to predefined classes. It could be noted that all these datasets commonly exhibit a balanced ratio of background to object, with objects captured at the average height of a person. Datasets that differ in these aspects, for example, those that consist of fine-grained objects, are less common. These datasets consist of data taken from satellites or airplanes and in addition, stand out due to the presence of both types

of data: with a single object or even none and with an abundance of objects on a single image. Furthermore, these objects typically occupy a small fraction of the total image space in comparison to the background (Illarionova S. et al., 2021 a).

Remote sensing datasets hold a significant role in the detection of damage resulting from natural and anthropogenic disasters (Shadrin D. et al., 2024). Electric pylons are an important part of infrastructure that ensures the sustainability of the economy in regions. However, they can be damaged by earthquakes or strong winds. The rapid and accurate assessment of the electric utilities and its possible damages can be performed using remote sensing data and machine learning techniques (Illarionova S., 2022). A number of papers have been dedicated to the topic of pylon detection in satellite-collected data, such as (Xiaoling L., 2021), (Ou W., 2019), and (Tian G., 2020). However, the task of electric pylon detection presents challenges, primarily stemming from the limitations of available open-source datasets. This absence not only poses challenges related to dataset diversity but also exacerbates the

¹ <https://orcid.org/0000-0002-9719-2461>

² <https://orcid.org/0000-0003-3486-8214>

³ <https://orcid.org/0000-0003-2448-9907>

complexities associated with the varied nature of different regions within the Russian Federation.

This study focuses on augmentations and models designed to improve performance on datasets with fine-grained attributes, with a particular focus on the Electric Pylon Detection (EPD) dataset introduced in (Sijia Q., 2020).

In this study, we accomplished the objective of detecting pylons within the Russian territory utilizing data from sensors with varying resolutions. Through training on the initial dataset, our approach demonstrated robust pylon detection capabilities which we then further improved with the implementation of the augmentation method contributing to enhanced accuracy and effectiveness in the pylon detection task.

2 RELATED WORKS

2.1 Field Overview

In this study, we focus on remote sensing data. The basic example of such a dataset was presented in (David N., 2021). It is the dataset from the MNIST family distinguishing feature of which is lack of background. The lack of diversity in classes is another of its drawbacks. Later the same year another aerial dataset was introduced in the paper (Xian S., 2021). The authors of FAIR1M have done an outstanding job of gathering data from multiple sources including Google Earth and one of the Chinese satellites. All data was marked up into 37 classes of different types of vehicles and geographical structures and compiled into one uniform format. The main advantage compared to other datasets is the number of instances and diversity of classes. The detriment of this dataset is the strong dispersion of data. Previous to that, another work on this field was presented in the paper (Gui-Song X., 2017). The authors introduced a dataset for aerial detection, data was collected from different sources and was marked up into 15 categories. But in order to narrow the research field, we decided to work with the EPD dataset (Sijia Q. 2020), - a dataset that consists of aerial imagery containing pylons.

2.2 Initial Dataset

The EPD dataset was presented in the (Sijia Q., 2020) article alongside the result of comparison of multiple computer vision models. The dataset contains 1450 images for training purposes and an additional 50 for

validation. Images were collected from two main sources - Google Earth and the Pleiades satellite. Images from the latter source are orthoimages, while images from the former are multispectral products acquired by various sensors. The ability of deep learning detectors to generalize will be tested using such multi-source data. Another difficulty occurs due to the great variety in sizes and shapes of pylons as well as other factors like differences in colors and the variety in the angle of observation.

2.3 Models

Models used for the task of object detection usually belong to the supervised learning category, where labeled data is used. As a result, algorithms should return the classes of all objects and the coordinates of their bounding boxes. The problem occurs in the length of an output layer as it is not constant therefore we can not preset the number of objects we expect to find. The possible solution is to choose multiple regions in which we expect to discover an object, however, the number of them could be exceedingly large. Multiple models were developed to overcome this.

The authors of (Ross G., 2014) proposed a method that firstly searches for 2000 regions of an image that presumably contains an object; later those areas are sent to the CNN model to extract features and build a prediction on the presence of an object in it. Although this algorithm reduces the number of regions of interest, it still suggests 2000 of them, therefore it takes a long time to make a prediction. Based on the drawbacks of previous work, one of the authors introduced an improved method in (Ross G., 2015). The main advantage of that work is in the processing of proposed regions, it allows convolution operation once per image rather than for every area separately. In the next work (Shaoqing R., 2016), the authors managed to improve another part of the algorithm. The use of selective search used to go through possible regions was time-consuming, so the authors managed to eliminate it, instead, they used a separate network to predict regions of interest.

All previous works had two parts: searching for the regions and then predicting using them. The paper (Joseph R., 2016) contains a method that uses a single CNN to find bounding boxes and category probabilities. This algorithm works significantly faster than other methods but it is known for having problems with detecting small objects within an image. Other attempts to improve existing algorithms were made.

RetinaNet introduced in (RetinaNet, 2018) improves results on small-scaled objects. It features two novelties introduced in (Tsung-Yi L. Piotr D., 2017) and (Tsung-Yi L., Priya G., 2017). The algorithm from the former article is used in the feature extraction and the focal-loss that was presented in the latter one is used as a solution for the class imbalance in the target prediction. Focal loss resolves the problem that appears due to an imbalance in the background and the foreground. It prevents the detector from being overwhelmed by an exceedingly large number of easy negatives. As a feature extractor, RetinaNet uses ResNet (Kaiming H., 2015) with 101 layers as it is a quite new network with great results in various object detection tasks. Overall this model performs well and belongs to the class of one-stage detectors as it requires just one pass through the network to extract features and make a classification. The difference between one-staged and two-staged detectors can also be seen in Figure 1.

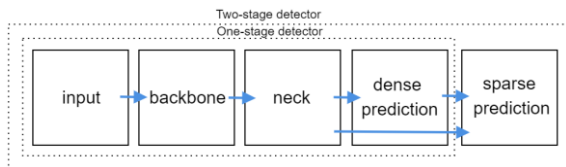


Figure 1: Object detector.

Trained on COCO, RetinaNet has advantages in the metrics compared to the third version of YOLO, however, it can not make multiple predictions simultaneously. The latter model is also a one-stage detector but it uses Darknet-53 (Joseph R., 2013-2016) for feature extraction. Darknet-53 is composed of multiple blocks that contain two convolution modules 1x1 and 3x3 after which comes one residual module, overall, it has 53 convolutional layers. Also, its convolution module consists of Conv2d, BatchNormalization, and leaky ReLU layers. Input images are passed five times through downsampling. Furthermore, it uses the idea of multi-scale features presented in (Tsung-Yi L., Piotr D., 2017). YOLOv3 (Joseph R., 2018) uses the bag of freebies and the bag of specials to improve the algorithm's performance.

The bag of freebies includes

- Complete IoU loss (CIOU)
- Drop block regularization
- Several augmentation approaches

The bag of specials

- Mish activation
- Diou-NMS
- Modified path aggregation networks

As for the performance of the YOLOv3, it could be noticed that the detection of small objects improved compared to the previous versions of YOLO. At the same time, it has comparatively worse results on medium and large object detection.

Another YOLO version was proposed in YOLOv4 (Alexey B., 2020) by a different author. Improvements were made in the architecture of the model itself and in the feature extraction module. The fourth version's backbone consists of CSPDarknet53 (Chien-Yao W., 2019) over the Darknet-53 in its predecessor. The former addresses the repeating gradient information in big backbones and integrates gradient change into a feature map, which improves inference speed, and accuracy, and reduces the model size by decreasing the number of parameters. SPP (Kaiming He, 2014) and PANet (Shu L, 2018), which adopt FPN, were used afterward in order to increase the receptive field of the feature extractor module and simultaneously shorten its important features. Similar to the third version, the fourth uses the bag of freebies and the bag of specials. Also, a new method was utilized - Mosaic augmentation which expands data by combining 4 images in a 2x2 grid. Furthermore, several methods were improved in order to adapt training to the usage of a single GPU.

Compared to the YOLOv4 the YOLOv5 (YOLOv5, 2020) abandons the previously utilized SSP. At the same time, it introduces back the Focus layer from YOLOv3 and improves it by replacing the first three layers with just a single one. Similar to the third and the fourth versions it also generates three output of feature maps that helps to achieve multi-scale predictions.

2.4 Augmentations

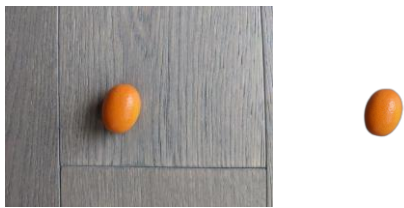
Nevertheless, all of the presented algorithms use a lot of data during training which is a major drawback in the areas that are resource-consuming when it comes to data collection.

Due to demand in the size of data used for training, overfitting or inability to generalize is a possible outcome. At the same time, data acquisition is a costly process, therefore, improving quality using smaller datasets is a desirable goal. Traditional methods of data augmentation are used to diversify data (Nesteruk S., 2024). Due to the properties of the researched dataset such as a few objects on an image, methods of augmentation that replenish a number of them could be used. There are several possible approaches. The simplest one concludes by taking a copy of the area within the bounding box and placing it in the other place of an image without any

processing (Ghiassi, G., 2021). That could be not applicable if the background of the new place significantly differs from the original location. In this case, other algorithms could be used in order to seamlessly blend objects into the new position. There are different methods to achieve these results. Library REMBG introduced in (Daniel G., 2020) is one of them. It works well in the case of medium and large objects. It also demonstrates a great performance when the background and the foreground are clearly distinct. When the object of detection is small, REMBG performance deteriorates significantly. All the examples described can be seen in Figure 2.



(a) the result of rembg on an image of a pylon



(b) the result of rembg on images of oranges on the floor

Figure 2: The results on the oranges imitate the expected results on the images of the pylon.

Another method of augmentation based on background blending was introduced in (Hilmi K., 2020). It proposes a two-staged method consisting of training and augmentation.

The first stage entails separate training of a generative network and a detector network. Patches of images are being fed to both models since the main goal of this method is to generate a new instance on a small patch of an original image. Similarly, the detector network is trained to be able to separate the generated samples from the original ones.

The second stage involves augmentation based on the previously trained generative network. Afterward, the results are sent to the detector to determine whether or not they are feasible. Eventually, samples that pass the detector are collected in the augmented set.

The remote sensing domain poses additional challenges in data processing. Therefore, the advanced augmentation algorithms are proposed to address these challenges by combining multispectral information (Illarionova S., 2021 b) or integrate the spatial properties of the studied environment (Dolgaia L., 2023).

3 METHODOLOGY

3.1 The Collected Dataset

Based on our research, there is a lack of openly available datasets containing satellite images and information on electrical pylons in Russia that can be directly used for machine learning model development. Therefore, we collected a special dataset and created annotations. For this reason, we chose to use the software QGIS (QGIS, 2022), which allows users to upload multiple different maps and obtain aerial photographs from them. For the maps, we settled on the Google Earth map since it appears to be the most accurate and also offers multiple resolutions. We chose to collect data from several districts because Russia's size results in radically varying conditions of biomes. Nonetheless, to simplify the task we have limited seasons to spring, summer, and the beginning of autumn.

We utilized Roboflow (RoboFlow, 2020) to annotate the photographs after they were collected. Most of the models we were working with use the Yolo format and this website allows users to easily process large quantities of data and save them in this format. Another great feature is that it also allows you to preprocess and augment images beforehand. As a result, using the data obtained from that website becomes rather simple and straightforward.

It is known that there is a large spectrum of different types of pylons and we decided not to handle some of them. Examples of used and unused pylons can be found in Figure 3.

3.2 Models Overview

We conducted experiments with the following models: YOLOv3 (Joseph R., 2018), YOLOv4 (Alexey B., 2020), and YOLOv5 (YOLOv5, 2020). The EPD dataset (Sijia Q., 2020) was used for the training. For the inference, we collected samples containing pylons on the territory of Russia using QGIS tools.



(a) Annotated pylons



(b) Poles skipped during annotation

Figure 3: Examples of different types of pylons.

3.2.1 YOLOv4

The implementation YOLOv4 is from the official GitHub repository (Alexey B., 2020). The configuration file was changed according to the information from the research (Sijia Q., 2020). There are some differences in the used terms between the model and the article, as the latter states that the batch size should be 4, and the whole number of epochs should be 25. At the same time, the model takes a number of iterations therefore, there should be $(1450 / 4) * 25 \sim 9063$ iterations. According to the information in the article, other changes in the original configuration file include:

- the sizes of anchors should be changed to (26, 22, 30, 41, 35, 60, 45, 29, 47, 48, 61, 86, 63, 35, 88, 55, 168, 140)
- parameter “random” should be set to 0
- there should be 500 iterations of warm-up
- parameters “momentum” should be set to 0.9
- learning rate should be set as 0.0125
- on epochs 20 and 23 (7250 and 8338 iterations respectively) learning rate should be decreased by 10

However, these parameters lead to an almost immediate explosion of loss in the first few iterations. Multiple experiments were conducted to determine the cause. In the end, it was identified that the learning rate from the article - 0.0125, is not optimal for the current model. Multiple possible changes were tested, and the learning rate with the number of warm-up iterations was the main focus. It was decided to use $lr = 0.0015625$ and $warm-up = 500$ since this

initialization leads to better results in the metric on the validation.

3.2.2 YOLOv3

YOLOv3 has some similarities in configuration to the fourth version. For example, the same sizes for anchors were used (26, 22, 30, 41, 35, 60, 45, 29, 47, 48, 61, 86, 63, 35, 88, 55, 168, 140). At the same time, some new parameters are also presented, such as $\gamma = 0.8$ and $\beta = 1.0$ used in the focal loss. Other changes include:

- batch size set to 16
- the total number of epochs to 20 (1813 iterations)
- the learning of 0.01
- momentum should be set to 0.9
- on epochs 16 and 18 (1450 and 1632 iterations respectively) learning rate should be decreased by 10
- warm-up should be removed (set to 0 iterations)

Similar to the previous model, these parameters also lead to the explosion of the loss, therefore, it was decided to use the learning rate from the official EPD repository (Sijia Q., 2022). We conducted additional experiments on that parameter. The final configuration includes $lr = 0.000625$, the model trained with this parameter leads to the best results we could achieve on YOLOv3 architecture.

3.2.3 YOLOv5

YOLOv5’s implementation differs from the previous versions as it was written on PyTorch. Initially, the only change made to the original configuration file was to decrease the learning rate to 0.001 from 0.01 as it was the main improvement made in the fourth and third versions of YOLO.

3.2.4 YOLOv5 evolution

One of the advantages of the fifth version over the previous is that it has an implemented hyperparameter evolution. This method optimizes parameters using genetic algorithms. About 25 parameters are employed in YOLOv5 for various training settings. Even when using the evolution method for parameter optimization it is quite important to initialize them with viable values. After evolution, the best results were achieved using these parameters:

- lr_0 should be set to 0.01 and lrf to 0.02

- momentum and weight decay used in an optimizer should be 0.937 and 0.0005 respectively
- there should be 5 epochs for warmup
- HSV saturation augmentation should be set to 0.7, value augmentation to 0.3, and HSV augmentation in hue to 0.015
- set focal loss gamma as 0
- set probability of scale, shear, rotation, and translation augmentations as 0
- set IoU training threshold to 0.2
- set probability of mosaic and mixup augmentations to 1 and 0 respectively
- set probability of flipping image up-down to 0.3, and for left-right to 0.5
- scale augmentation should be set to 0.7



(a) EPD pylons (collected from the territory of China)



(b) Our pylons (collected from the territory of Russia)

Figure 4: Comparison of pylons from China and Russia.

3.3 Augmentation

We can compare the types of pylons from China and Russia, as shown in Figure 4. It is evident that they exhibit significant differences, with some shapes being notably distinct from those found in the other country. Unfortunately, there is insufficient accurate data available to train a model for pylon detection using Russian data. However, we were able to obtain a sufficient amount of similar data from China, with a larger number of collected and annotated objects.

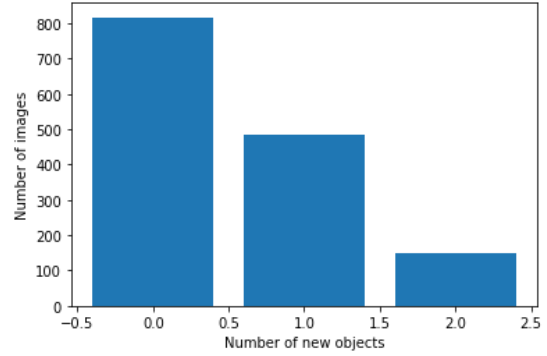


Figure 5: Distribution of the number of new objects.

Therefore, we propose an augmentation strategy that would integrate some of our data into the already collected dataset. Firstly, we separated 5 images from our dataset, since some parts of these images are going to be used in augmentation we removed them from validation as, otherwise, it could compromise the results. Using these 5 images, we collected 12 shadows from those images and outlined them by hand (you can see an example in Figure 7). For augmentation, we used all objects from the EPD train dataset (EPD dataset-S), number of new objects that are going to be added to an image is a random number from 0 to 2, where the probability for 0 is ~55%, for 1 - ~33%, and for 2 - ~11%. The distribution of the number of new objects can be found in Figure 5.

Augmentation algorithm

```

for image in train dataset do:
  number of new images ← random(0, 0, 0, 0, 0, 1, 1, 1, 2)
  new labels ← image_labels
  while number of new images ≠ 0 do
    number of new images ← number of new images - 1

    new object ← random(image_bounding_boxes)
    new shadow ← resize_shadow(random(collected_shadows),
                               new object)

    new object location ← get_vacant_location(new object, image)
    new labels ← {new labels} ∪ {new object location}

    shadow direction ← random(north, east, south, west)
    new shadow location ← get_shadow_location(new shadow,
                                              shadow direction, new object, new object location, image)

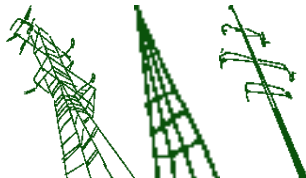
    image ← paste_region(image, new object, new object location,
                          blend_mode=normal)
    image ← paste_region(image, new shadow, new shadow location,
                          blend_mode=multiply)
  end while
  save image to the new location
  save new labels to the new location
end for

```

Figure 6: Augmentation algorithm.

For every new object, one of the ground truth boxes in an image is chosen, copied, and pasted into the new place. A shadow is chosen randomly and resized to match the size of the object. The direction

of the shadow is chosen randomly from north, east, south, and west. The shadow is pasted on top of an image as on a layer with multiply types. The pseudocode for that segment can be found in Figure 6. The examples of shadows and results of augmentation can be found in Figure 7.



(a) Examples of masks used for augmentation



(b) Original and Augmented images

Figure 7: Augmentation examples.

The numbers of images and objects in datasets can be found in Table 1.

Table 1: Dataset statistics.

	EPD Train	EPD Augmented	EPD Validation	Our Validation
Number of Objects	3072	3854	159	72
Number of Images	1450	1450	50	26

YOLOv5 was trained using augmented data. Hyperparameters were left the same as stated in the previous section.

3.4 Training on Half of the Data

Experiments on the number of images used during the training were conducted. We chose 725 images from the initial EPD dataset and used them to train the model with parameters from our initial run of YOLOv5. At the same time, augmented counterparts

of the chosen images were used to train another YOLOv5 model with the same parameters.

3.5 Evaluation Metrics

As a criterion of comparison between different models and training setups, we used mAP which is calculated as the area under the Precision-Recall curve. By definition:

$$Precision = \frac{TruePositive}{TruePositive + TrueNegative}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

Since the predictions of object detection cannot be easily separated into groups of TruePositives, TrueNegatives, FalsePositives, and FalseNegatives, we used Intersection over union (IoU) threshold to distribute them. It is calculated as follows:

$$IoU = \frac{AreaOfOverlap}{AreaOfUnion}$$

Here areas of overlap and areas of the union are calculated based on pair of ground truth and predicted bounding boxes. The closer IoU to 1 the closest prediction to the ground truth, therefore, based on the threshold we can classify whether a prediction is a TruePositive or a FalsePositive. In this case, we also can simplify precision for the task of object detection as follows:

$$Precision = \frac{TruePositive}{NumberOfPredictions}$$

To calculate mAP, we should separate classes under consideration, for each of the predictions should be sorted based on probabilities, we need to calculate the precision and recall of each prediction, we can calculate average precision (AP) for a class as follows:

$$AP = \sum_{i=0}^{n-1} (Recall_i - Recall_{i+1}) * Precision_i$$

Here n is a number of predictions. The mAP metric is defined:

$$mAP = \frac{1}{c} \sum_{i=1}^c AP_i$$

Here c is a number of classes and AP_i is an average precision of class i. The best parameters for the models are chosen based on the introduced metrics.



(a) Ground Truth (b) Yolov4 Predictions



(c) Yolov3 Predictions (d) Yolov5 Predictions

Figure 8: Examples of prediction with different models.

4 RESULTS AND DISCUSSION

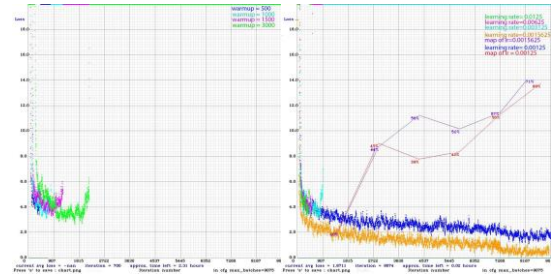
To evaluate the quality of detection, we used the mAP metric which stands for mean average precision for the class of pylons. The results were compared to methods introduced in the initial article on the EPD Dataset, to ensure that our results could compete with existing works. Multiple experiments were conducted for each model.

We were able to determine the cause of the loss explosion during the training of YOLOv4. Multiple approaches have been tested, and the results of them can be found in Figure 9a and Figure 9b. The usage of our configuration leads to better results than those presented in the original article. We managed to get to 70.72 percent by comparison to 65. An example of

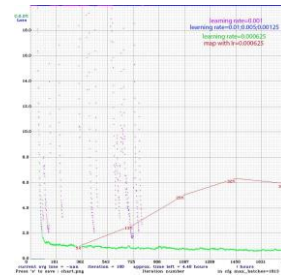
the prediction using our additional collected data is shown in Figure 8b.

Our results on YOLOv3 were not as good. The loss value on the training dataset and the final result on the validation dataset of this model are shown in Figure 9c. More experiments should be conducted to improve the results. At the current stage, the accuracy of the original validation and on our data that this model produces is the lowest among all the results we achieved. An example of the prediction on our data can be found in Figure 8c.

YOLOv5 was tested most thoroughly. Without any major configuration changes, the model was able to achieve the best results on validation - 79,8% mAP. The training chart is shown in Figure 10a and the prediction on our data is shown in Figure 8d.



(a) Loss charts with different warm-up iterations YOLOv4 (b) Loss charts with different learning rate values YOLOv4



(c) Loss charts with different learning rate values YOLOv3

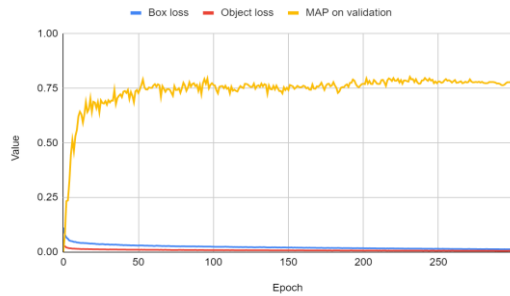
Figure 9: Training with different hyperparameters.

Evolution was used in order to optimize hyperparameters. It leads to slightly worse results on validation and significantly worse results on our data training chart of training can be found in Figure 10b.

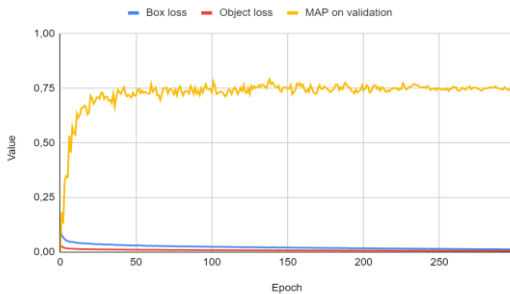
Table 2: mAP results with IoU threshold 0.5.

	EPD Train	EPD Validation	Our data
YOLOv4	92.30%	70.72%	44.88%
YOLOv3	47.20%	30.20%	15.30%
YOLOv5	99.50%	79.80%	45.20%
YOLOv5 after evolution	99.50%	76%	39.60%

All mAP results and comparisons between initial models are shown in Table 2. It should be noted that the results of YOLOv5 model on train data could be interpreted as overfitting. Therefore, for further improvement, we have implemented data augmentation to diversify existing training data.



(a) YOLOv5 training results without evolution



(b) YOLOv5 training results after evolution on initial hyperparameters

Figure 10: Training results.

YOLOv5 was trained using augmented data. In comparison to previous results of evolution, this model achieves slightly worse results on validation - 75,4% but better results on our additional collected data - 41.5%. Also, in Figure 11a, a higher train loss is achieved but mAP on the training dataset stays almost the same - 99.4%.

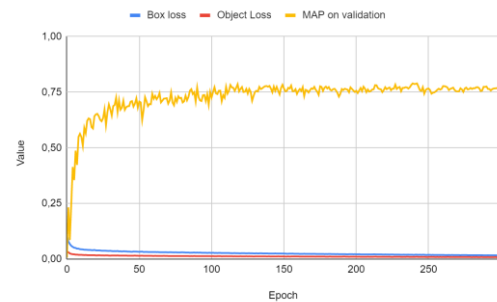
Experiments on the amount of data used in training show a positive contribution of the

augmentation in the results on our additional data. Training charts of models trained on original and augmented data are shown in Figure 11b and Figure 11c.

Table 3: Results of training on the half of the dataset.

	EPD Train	EPD Validation	Our data
YOLOv5 trained on original dataset	99.5%	75.4%	37.2%
YOLOv5 trained on augmented dataset	97.3%	71.2%	39.9%

The training results are presented in Table 3. It is shown that our model trained on the augmented data produces better results on our data. However, it should be noted that at the same time, it loses some precision on the original validation dataset.



(a) YOLOv5 trained on augmented data

(b) YOLOv5 training on half of the original data



(c) YOLOv5 training on half of the augmented data

Figure 11: Additional training results.

We managed to surpass the original results. The best results were achieved using the YOLOv5 model and trained on the EPD dataset with the result of 79,8% and 45,2% on the original and our datasets respectively. Evolution was used to further improve results on YOLOv5.

In order to mitigate overfitting and improve results on our data, we implemented augmentation that was used to diversify training data. Model, trained on the augmented dataset, was able to outperform YOLOv5 after evolution.

Experiments with the quantity of information utilized in training show that our augmentation leads to improvement in the results on our dataset.

We were able to achieve our goal of being able to detect pylons on the territory of Russia, collected using sensors of different resolutions, after training on the initial dataset. And also we were able to implement augmentation that aids tested models to produce better results on the collected dataset.

5 CONCLUSION

In this paper, we integrated existing methods to detect objects collected within the territory of Russia. Most studies on object detection assume that the objects in images occupy significant space and that all data has the same resolution. Building upon these observations, we aimed to enhance the accuracy of object detection on the EPD dataset, which contains images from various sources and often has an excessive amount of background compared to the area occupied by objects. This task could be instrumental for future advancements in object detection. Furthermore, this research could facilitate monitoring the number of pylons in extremely windy areas to identify any that may have fallen due to weather conditions and require maintenance.

ACKNOWLEDGEMENTS

The work was supported by the Russian Science Foundation (Project No. 23-71-01122).

REFERENCES

- Xian, S., Peijin, W., Zhiyuan, Y., Feng, X., Ruiping, W., Wenhui, D., Jin, C., Jihao, L., Yingchao, F., Tao, X., Martin, W., Stefan, H., Cheng, W., Kun, F., 2021. FAIR1M: A Benchmark Dataset for Fine-grained Object Recognition in High-Resolution Remote Sensing Imagery. In *arxiv: 2103.05569*.
- David, N., Samantha, E. Miller, N., 2021. MNIST: A Benchmark Satellite Dataset. In *arxiv: 2102.04266*.
- Gui-Song, X., Xiang, B., Jian, D., Zhen, Z., Serge, B., Jiebo, L., Mihai, D., Marcello, P., Liangpei, Z., 2018. DOTA: A Large-scale Dataset for Object Detection in Aerial Images. In CVPR 2018.
- YOLOv5 Accessed on: Jan. 23, 2022. [Online]. Available: <https://github.com/ultralytics/yolov5>
- Zheng G., Songtao L., Feng W., Zeming L., Jian S., 2021. YOLOX: Exceeding YOLO Series in 2021. In arxiv: 2107.08430.
- RetinaNet Accessed on: Jan. 3, 2022. [Online]. Available: <https://github.com/yhenon/pytorch-retinanet>
- Shaoqing, R., Kaiming, H., Ross, G., Jian, S., 2016. Faster R-CNN: Towards Real Time Object Detection with Region Proposal Networks. In IEEE Trans. Pattern Anal. Mach. Intell. 39(6).
- Zhaowei, C., Nuno, V., 2018. Cascade R-CNN: Delving into High Quality Object Detection. In CVPR 2018.
- Wouter Van, G., Simon, V., Stamatios, G., Luc Van, G., 2021. Unsupervised Semantic Segmentation by Contrasting Object Mask Proposals. In ICCV 2021.
- Daniel G, 2020. Rembg. Accessed on: Feb. 10, 2022. Available online: <https://github.com/danielgatis/rembg>
- Hilmi, K., Cihan, O., Alptekin, T., 2020. Data Augmentation for Vehicle Detection in Aerial Images. In ICPR Workshops (8)
- 2019 MNIST. Accessed on: Feb. 20, 2022. Available online: <https://deepai.org/dataset/mnist>
- 2017 ImageNet. Accessed on: Feb. 20, 2022. Available online: image-net.org/download.php
- 2021 COCO. Accessed on: Feb. 20, 2022. Available online: <https://cocodataset.org/home>
- Joseph, R., Santosh, D., Ross, G., Ali, F., 2016. You Only Look Once: Unified, Real-Time Object Detection. In CVPR 2016
- Ross, G., Jeff, D., Trevor, D., Jitendra, M., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR 2014:
- Ross, G., 2015 Fast R-CNN. In ICCV 2015.
- Tsung-Yi, L., Piotr, D., Ross, G., Kaiming, H., Bharath, H., Serge, B., 2017. Feature Pyramid Networks for Object Detection. In CVPR 2017
- Tsung-Yi, L., Priya, G., Ross, G., Kaiming, H., Piotr, D., 2017. Focal Loss for Dense Object Detection. In ICCV 2017
- ”torchvision.transform” Accessed on: Feb. 12, 2022. Available-online: <https://pytorch.org/vision/stable/transforms.html>
- Irvin, J., Sheng, H., Ramachandran, N., Johnson-Yu, S., Zhou, Sh., Story, K., Rustowicz, R., Elsworth, C., Austin, K., Ng, Y., 2020. ForestNet: Classifying Drivers of Deforestation in Indonesia using Deep Learning on Satellite Imagery. In arxiv: 2011.05479. 2020
- Liang, D., Geng, Q., Wei, Z., Vorontsov, D. A., Kim, E.L., Wei, M., Zhou, H., 2021. Anchor Retouching via

- Model Interaction for Robust Object Detection in Aerial Images, In arxiv:2112.06701 2021
- Qiao, S., Sun, Y., Zhang, H., 2020. Deep Learning Based Electric Pylon Detection in Remote Sensing Images. In Remote. Sens.
- He, K., Zhang, X., Ren, Sh., Sun, J., 2016. Deep Residual Learning for Image Recognition. In CVPR 2016.
- Darknet. J.R., 2022. Accessed on: Feb 11 2022. Available online: <http://pjreddie.com/darknet/>
- Misra, D., 2019. Mish: A self regularized nonmonotonic neural activation function. In arXiv: 1908.08681
- Zheng, Zh., Wang, P., Liu, W., Li, J., Ye, R., Ren, D., 2020. Distance-IoU Loss: Faster and better learning for bounding box regression. In AAAI 2020
- Bochkovskiy, A., Wang, Ch.-Y., Liao, H.-Y. M., 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. In arXiv: 2004.10934
- Wang, Ch.-Y., Liao, H.-Y. M., Wu, Y.-H., Chen, P.-Ya., Hsieh, J.-W., Yeh, I.-H., 2019. CSPNet: A new backbone that can enhance learning capability of cnn. In arXiv: 1911.11929
- He, K., Zhang, X., Ren, Sh., Sun, J., 2014. Spatial pyramid pooling in deep convolutional networks for visual recognition. In ECCV (3) 2014
- Liu, Sh., Qi, L., Qin, H., Shi, J., Jia, J., 2018. Path Aggregation Network for Instance Segmentation. In CVPR 2018
- Redmon, J., 2022. YOLOv5. Accessed on: Apr 24. Available online: github.com/ultralytics/yolov5
- Redmon, J., Farhadi, A., 2018. YOLOv3: An Incremental Improvement. In arXiv: 1804.02767
- Bochkovskiy, A., Darknet repository, Accessed on: Apr 24, 2022. Available online: github.com/AlexeyAB/darknet
- Qiao, S., Sun, Y., Zhang, H., 2022. Electric Pylon Detection in RSI. Accessed on: Apr 24, 2022. Available-online: [//github.com/qsxjvz/Electric-Pylon-Detection-in-RSI](https://github.com/qsxjvz/Electric-Pylon-Detection-in-RSI)
- QGIS: A Free and Open Source Geographic Information System, Accessed on: Apr 24, 2022. Available online: <https://qgis.org/en/site/>
- Roboflow. Accessed on: Apr 24, 2022. Available online: <https://roboflow.com>
- Xiaoling, L., Chen, J., Qiu, X., 2021. A Pylon Detection Method Based on Faster R-CNN in High-Resolution SAR Images. In APSAR.
- Ou, W., Yang, Z., Zhao, B., Fei, X., Ma, X., Yang, G., 2019. Research on Automatic Extraction Technology of Power Transmission Tower Based on SAR Image and Deep Learning Technology. In ICISCE.
- Guishen, T., Song, M., Xuejiao, B., Lijuan, L., Yang, Zh., Wenhao, O., Xiangze F., Yuanpeng, T., 2020. Electric Tower Target Identification Based on High-resolution SAR Image and Deep Learning, In Journal of Physics: Conference Series
- Illarionova, S., Nesteruk, S., Shadrin, D., Ignatiev, V., Pukalchik, M., Oseledets, I., 2021. Object-based augmentation for building semantic segmentation: Ventura and santa rosa case study. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 1659-1668).
- Illarionova, S., Nesteruk, S., Shadrin, D., Ignatiev, V., Pukalchik, M., Oseledets, I., 2021. MixChannel: Advanced augmentation for multispectral satellite images. Remote Sensing, 13(11), 2181.
- Shadrin, D., Illarionova, S., Gubanov, F., Evteeva, K., Mironenko, M., Levchunets, I., Burnaev, E., 2024. Wildfire spreading prediction using multimodal data and deep neural network approach. Scientific Reports, 14(1), 1-17.
- Ghiasi, G., Cui, Y., Srinivas, A., Qian, R., Lin, T. Y., Cubuk, E. D., Zoph, B., 2021. Simple copy-paste is a strong data augmentation method for instance segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 2918-2928).
- Nesteruk, S., Illarionova, S., Somov, A., 2016. Image Dataset Augmentation A Survey and Taxonomy. Measurements and Instrumentation for Machine Vision, 110-136.
- Dolgaia, L., Illarionova, S., Nesteruk, S., Krivolapov, I., Baldycheva, A., Somov, A., Shadrin, D., 2023. Apple Tree Health Recognition Through the Application of Transfer Learning for UAV Imagery. In 2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA) (pp. 1-8). IEEE.
- Illarionova, S., Shadrin, D., Ignatiev, V., Shayakhmetov, S., Trekin, A., Oseledets, I., 2022. Estimation of the canopy height model from multispectral satellite imagery with convolutional neural networks. IEEE Access, 10, 34116-34132.