Synthetic Dataset Generation for Efficient Neural Network Training

Oleg Rudenko[1], Oleksandr Bezsonov[1], Kateryna Vashchenko[1] and Sofiia Rutska[1]

[1] *Kharkiv National University of Radio Electronics, Nauky Ave. 14, Kharkiv, 61166, Ukraine*

Abstract

The shortage of real images for training datasets preparation and complication of their annotation process are significant issues in real-life machine learning applications. This work considers an alternative approach based on replacing real images with synthetic data generated with usage of 3D models. The various use cases for the generated datasets and their possible applications are considered, as well as the advantages of the fast annotation process for this kind of data. The Mask-RCNN neural network for solving the problem of image segmentation related to the detection of rescuers at disaster sites was trained on synthetic datasets and showed sufficient performance.

Keywords

Deep Learning, Mask-RCNN, 3D models, Synthetic Dataset, Augmentation

1. Introduction

It's always a serious problem to prepare a good dataset for neural network training. Usually, it is impossible to take many pictures of real-life scenes because of privacy, ethics and costly equipment. In this work we consider synthetic data utilization generated from 3D models of the objects. As an example, we chose the task of recognizing rescuers at the scenes of disasters to better coordinate their actions in a full-scale war in Ukraine.

Russian invaders often carry out shelling of various infrastructure in Ukraine, residential areas, houses, and neighborhoods. As a result, many different buildings are partially or completely destroyed. Missile fires continues on a daily basis. Many civilians have suffered due to these attacks. Destroyed homes are not the only problem of Ukrainians, many people find themselves in desperate situations, when there is no way to get out and save themselves, and there is practically no oxygen under the rubble of the house, in such moments not in minutes, every second can cost someone's life. To optimize the work of rescuers, to eliminate contingencies that are beyond the control of the human eye, unmanned aerial vehicle (UAV) or drones in combination with neural networks might be used. Nowadays there is no need to prove that neural networks based autodetection works much faster than the human eye. It is possible to track the location and surroundings of a rescuer, prevent new collapses, keep civilians from dying and save more lives. However, there is a problem with getting several hundreds or thousands of samples to train a neural network. Inability to get hundreds or even thousands of rescuers images has several reasons: the privacy of each worker, his and his family's safety, strict restrictions on photographing during wartime to prevent the correction of missile attacks on the city, and an unacceptable and unsuitable environment for taking pictures. As a result, using 3D models of sensitive objects is a good choice. We used different pictures of disasters all over Ukraine, created 3d models of rescuers and simulated different catastrophic scenes. It is also a very difficult task to annotate existing pictures because it is mainly about manual processing thousands of images.

COLINS-2023: 7th International Conference on Computational Linguistics and Intelligent Systems, April 20–21, 2023, Kharkiv, Ukraine
EMAIL: oleh.rudenko@nure.ua (O. Rudenko); oleksandr.bezsonov@nure.ua (O. Bezsonov); kateryna.vashchenko@nure.ua (K. Vashchenko); sofiia.rutska@nure.ua (S. Rutska)
ORCID: 0000-0003-0859-2015 (O. Rudenko); 0000-0001-6104-4275 (O. Bezsonov); 0009-0005-5449-2455 (K. Vashchenko); 0009-0009-0575-8172 (S. Rutska)

The purpose of this article is to show the advantages and disadvantages of training a neural network on synthetic data. And also to show the advantages and possibility of not annotating photos manually anymore, but to let a program that makes hundreds of annotations in a couple of seconds on the basis of synthetic data.

2. Related works

Synthetic data sets are widely used in the area of machine learning and image recognition. Microsoft creates realistic, multifaceted and lifelike synthetic faces. It starts with a universal face template, and then applies a random combination of personalities, expressions, textures, hairstyles and clothing. Finally, the resulting face is shown in a random environment using Cycles, a path-tracing software that uses physical data to achieve the highest level of realism. But there is still a big difference between a real face and a synthetic model. Mixing different textures and introducing new techniques help to achieve similarity. For machine learning models to be effective, they must be trained in a way that works well among humans. Microsoft evaluated the performance of detectors using the MUCT Face Database, which represents different lighting, age and ethnicities, and verified that models trained on synthetic data alone can successfully be similar to real data of different people [1]. A method of the synthetic faces generation is presented at Fig. 1.

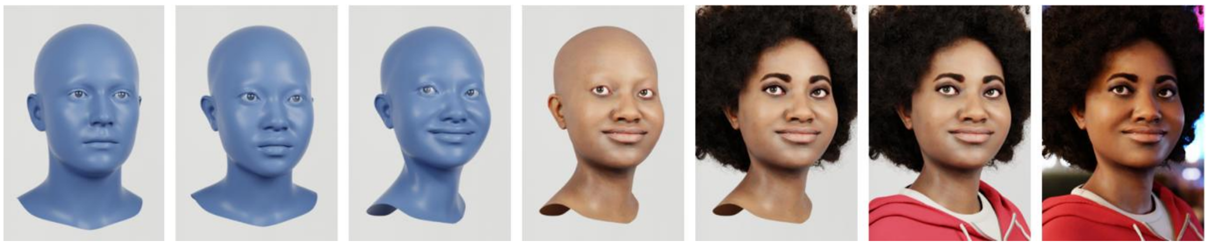


Figure 1: Microsoft's method of creating a synthetic face

The article [2] provides an overview of various image synthesis methods, such as generative-adversarial networks (GAN), autoencoders (autoencoder), variational autoencoders (VAE) and others. In addition, the article examines the application of synthetic data to improve the training of deep neural networks in medical diagnosis and treatment tasks.
In particular, the authors note that the use of synthetic data can help to cope with the problem of the limited number of available medical images for training neural networks, as well as reduce the time and cost of collecting and annotating real data. They also discuss the advantages and disadvantages of different methods of generating synthetic data, as well as the possibilities of their application in medical practice. In conclusion, they express their opinion that, Generative Adversarial Networks (GANs) have been extensively employed in Radiotherapy (RT). GANs are capable of automatically learning anatomical features from various modalities of images, improving the quality of images, generating synthetic images, and performing automatic dose and plan calculation in a shorter amount of time. Although the GAN model cannot yet replace the work of radiotherapy doctors, it possesses significant potential to enhance radiologists' workflow. There are numerous opportunities to enhance diagnostic accuracy, reduce potential risks during radiotherapy, and decrease the time and cost for plan calculation.

In [3] an overview of recent research on the use of synthetic data generation techniques for health records is given. Health data is a valuable resource for researchers and healthcare professionals, but access to this data can be limited due to privacy concerns and data protection regulations. To overcome these limitations, researchers have explored the use of synthetic data generation techniques to create artificial datasets that can be used for research and analysis without compromising patient privacy. The article reviews various approaches to synthetic data generation, including data masking, generative adversarial networks, and differential privacy techniques. The authors of the article conducted a systematic review of the literature on synthetic data generation for health records and identified 29 relevant studies. The studies covered a range of health data types, including electronic health records, medical imaging data, and genomic data. The review found that synthetic data generation techniques have the potential to overcome many of the challenges associated with accessing real health data while still providing valuable insights for research and analysis. However, the authors note that there are still limitations and challenges associated with synthetic data generation, such as ensuring the accuracy and validity of the synthetic data and addressing potential biases introduced by the generation process. Overall, the article provides a comprehensive overview of recent research on synthetic data generation for health records and highlights the potential benefits and limitations of these techniques for healthcare research and analysis.

3. Methods and Materials

Data scientists often run into cases where they either do not have actual data or cannot use it because of various external factors, such as the confidentiality of the data itself or difficulties in its obtaining. To solve this problem, replacing authentic data or extending the existing dataset with synthetic data is used.

In order for the algorithm to work properly, an appropriate replacement of realistic authentic data is required. It is needed to use data like this to provide privacy, test systems, or create training data for machine learning algorithms.

Substitution of real data with synthetic with automatically created annotations by computer algorithms has following advantages:

- it is possible to generate as much data as needed;
- synthetic data generation is cheap and not require too expensive equipment;
- synthetic data is generated with absolutely precise annotations, manual reaction of which is very expensive or often even impossible;
- the synthetic environment can be easily modified by improving the 3D models;
- synthetic data can be used to replace some sensitive segments of real data. Because synthetic data does not include information about real people, privacy concerns are eliminated;
- synthetic data comply with all data privacy and copyright laws;
- synthetic data allows model’s training by using different variants of the same person with different hairstyles, facial vegetation, glasses, head positions, etc., as well as skin tone, ethnic traits, bone structure, freckles, and other characteristics to create unique faces. It significantly improves the model's robustness.

Like any new data collection methodology, even synthetic data faces with some problems. The biggest problem is that synthetic data not always accurately represent reality. The quality of synthetic data can vary across the data set. The quality of synthetic data depends on the quality of the raw input data. Distortions in the raw input data usually lead to uncertainty in the final dataset as well.

The generated objects can be used in the teacher-assisted training tasks to extend the training set, reducing it to partial learning and self-learning tasks. A fairly common approach is to train initially a model on the large synthetic dataset, and then fine-tune the model on a small set of available real data. Sometimes real data is not used in training at all. It is worth to note, however, that synthetic datasets cannot be used in test training sets: they must always contain only real objects.

Synthetic data, which refers to artificially generated data that mimics real-world data, has a wide range of potential applications across various industries. Here are a few examples of where synthetic data can be used:

- Machine learning;
- Autonomous vehicles;
- Video game development;
- Robotics;
- Security and fraud detection.

Overall, synthetic data has the potential to be used in a wide range of applications where real-world data may be difficult, expensive, or impractical to obtain. By generating artificial data that mimics real-world scenarios, synthetic data can help to improve the performance and accuracy of various systems and algorithms across different industries.

Synthetic data can solve many problems these days. Some spheres of activity need these kinds of datasets just now. The use of synthetic data is becoming a hot topic and its popularity is growing rapidly despite some known drawbacks.

3.1. Using 3D models

3D technology has revolutionized the way we design and create objects, offering a wide range of possibilities for modifying existing objects or creating entirely new ones from scratch. With 3D modeling software and 3D printing technology, it is possible to make precise and complex modifications to objects with ease, allowing greater creativity and customization than ever before.

Generating a dataset requires high accuracy with a minimum of time. Another advantage of 3D technology is the ability to create completely new objects from scratch. With 3D modeling software, designers can create highly detailed and complex models of everything they can imagine, from toys and gadgets to furniture and architectural structures.

One of the most exciting possibilities of 3D technology is the ability to create highly detailed and realistic simulations of objects and environments. This is particularly useful in such fields as medicine and engineering, where it is important to accurately model and test new products and designs before they are manufactured. By using 3D modeling software, researchers can create highly detailed models of human anatomy, complex machinery, and other objects, allowing for more accurate and effective testing and development.

3.2. Processing 2D images

It is also possible to create an image not only in 3D space, but also by modifying existing 2D images. This will still require some set of images, however, using digital image processing programs such as Photoshop and the like, it is possible to modify images significantly. This method is more time-consuming than the method of creating images using 3D graphics. Some privacy aspects may be violated because real images are used in the process. In addition, it is worth taking seriously the fact that such tasks require quite good image processing skills, as a high level of realism is required for the subsequent use of the resulting dataset. There is also a difference in the software that can be used. In this project only free software for 2-3D processing is used.

3.3. Data generation with Neural Networks

Another option for creating synthetic data is to generate images using neural networks. To train such a network real images are needed. Such an approach requires pairs of real and synthetic images with pixel matching or real images with annotations. This is probably an easier task, but such data is very difficult to collect. To create a pixel correspondence, it is needed to create a synthetic image that corresponds to a given real image. In addition, this method can give unpredictable results.

4. Neural Networks

Neural networks are an incredible machine learning tool that imitates the human brain and can solve complicated problems. Neural networks perform different tasks to help humans, they can create their own images and videos. Learning takes place in different ways, especially it is interesting to learn by feedback, when a neural network learns from the results of its own actions. Neural networks can learn by looking at the performance of other neural networks. This is called reinforcement learning. It reminds how real people learn by watching the actions of someone who knows what to do. A very painful topic is creating graphics for games and realistic effects, and a neural network can do that. It could even be used in movies, which would make life a lot easier for a lot of people. Not all data can be reliable and it is impossible to accept all information taken from a neural network as true, because neural networks can make mistakes, what has become more clear with introduction of ChatGPT [14].

Weights in neural networks are such parameters that determine how important the input signal is for solving some specific task the neural network has set. Each neuron receives some number of signals as input. The signals are multiplied by the corresponding weight. And signals that have passed this stage are considered weighted, they are added, and the result of their addition is sent to the output layer. The weights are adjustable, they are set and in the process of training they are adjusted for the accuracy of the result. The way the weights are adjusted depends on the training method.

Some approaches for training neural networks without a teacher:

- Data clustering.

Based on this approach, the neural network takes input data and finds similarities between them. It divides all the information into groups of data based on similarity. This is not an easy task, so different clustering algorithms such as K-means, DBSCAN, hierarchical clustering are used.

- Associative memory.

In this kind of teacherless learning, the neural network receives information in the input data and searches through it as a detector for connections between objects. It finds common features or highlights key similarities.

- Autoencoders.

In this type of learning, neural networks try to represent input data in a minimal form, which is called latent representation. The network learns to find encoded input data and decoded by minimizing errors.

It is important to understand that to get good results you need a large sample of data and a lot of computational resources. There are methods to optimize learning and prevent re-learning.

Neural networks solve a variety of tasks, but we are only considering those that work with the image, here are a few modern computer vision tasks:

- Classification – classifying an image by the type of object it contains.
- Semantic segmentation – determination of all pixels of objects of a certain class or background in the image. If several objects of the same class overlap, their pixels are not separated from each other in any way.
- Object detection – detection of all objects of the specified classes and determination of the encompassing frame for each of them.
- Segmentation of objects – detection of pixels belonging to each object of each class separately.

4.1. Convolutional Neural Network

Deep learning algorithms, in particular the convolutional neural network (CNN), have gained wide acceptance as a robust approach for learning predictive features directly from raw images. The basic principles of feature extraction with CNNs still apply. CNNs scan every pixel of an image and store information based on pixel value patterns.

At the core of CNNs are convolutional layers, which extract features from the input image by applying a set of filters to the image. These filters, also known as kernels, detect specific patterns and features, such as edges, corners, and textures. The output of the convolutional layer is a set of feature maps, which highlight the locations of the detected features.

In addition to convolutional layers, CNNs also typically include pooling layers, which downsample the feature maps by taking the maximum or average value of a local region. This helps to reduce the dimensionality of the feature maps and make the network more efficient.

Finally, CNNs typically include one or more fully connected layers, which use the extracted features to make a prediction about the input image. The output of the fully connected layers is a set of probabilities, indicating the likelihood of the input image belonging to each possible class.

One of the key strengths of CNNs is their ability to learn features directly from the raw image data, without the need for manual feature engineering. This makes them highly adaptable to new tasks and data sets. However, CNNs can require large amounts of data and computational resources to train, especially for complex tasks. In addition, they may not perform well in situations where there is insufficient data or when the input data is noisy or biased.

The topology of the network is guided by the problem to be solved, data from scientific articles, and one's own experimental experience.

The following steps that influence the choice of topology can be distinguished:

- Determine the problem to be solved by the neural network;
- Determine the constraints in the problem to be solved;
- Define the input and output.

The Region-based CNN (R-CNN) approach to object detection in a constrained domain is to consider a manageable number of candidate object domains and evaluate convolutional networks independently on each domain. R-CNN has been extended to allow visiting Region of Interests on feature maps using RoIPool, resulting in high speed and better accuracy. Faster R-CNN [15] has evolved this direction by training the attention mechanism with a regional suggestion network (RPN). The Faster R-CNN is flexible and robust to many subsequent refinements, and is now the leading system in several benchmarks.

In convolutional neural networks, which are widely used for image processing, cross-entropy can be used to train the network on classification tasks. Typically, the last layer of a convolutional network has a Softmax activation function that converts the network outputs into probabilities of belonging to each class, and the cross-entropy is used as a loss function for training the network.

The stochastic gradient descent algorithm is used to train the convolutional neural network

$$\Delta\theta(k) = -\eta\nabla_{\theta}J(\theta(k)); \tag{1}$$

$$\theta(k+1) = \theta(k) - \eta\nabla_{\theta}J(\theta(k)), \tag{2}$$

where – θ network parameters (elements of weight matrices, shifts, slopes of activation functions, etc.) – $J(\theta(k))$objective function (loss function); η – learning rate parameter.

If cross-entropy loss is chosen as an option, the training procedure for the output (fully connected) layer of the convolutional neural network will take the form

$$\Delta\theta(k) = -\eta\nabla_{\theta}C(\theta(k)); \tag{3}$$

$$\theta(k+1) = \theta(k) - \eta\nabla_{\theta}C(\theta(k)), \tag{4}$$

where

$$\theta(k) = (w,b) \tag{5}$$

Calculating the individual derivatives according to the configured parameters, we have

$$y_i^L = f_i^L \tag{6}$$

$$\nabla_w C = \frac{\partial C}{\partial w_{ik}^L} = \frac{\partial C}{\partial y_i^L}\frac{\partial y_i^L}{\partial w_{ik}^L} = \frac{\partial C}{\partial y_i^L} f_k^{L-1} = f_k^{L-1}(f_k^{L-1} - d_i); \tag{7}$$

$$\nabla_b C = \frac{\partial C}{\partial b_i^L} = \frac{\partial C}{\partial y_i^L}\frac{\partial y_i^L}{\partial b_i^L} = f_i^L - d_i. \tag{8}$$

$$\nabla_f C = \frac{\partial C}{\partial f_j^L} = -\frac{\partial}{\partial f_j^L}\left(\sum_{i=1}^{n} d_i \ln f_i^L\right) = -\frac{d_j}{f_j^L} = f_i^L - d_i; \tag{9}$$

The convolutional neural network kernel is a filter that slides over the entire image and finds its features anywhere, i.e., it provides invariance to offset.

The formulas for updating the convolution kernel are as follows

$$\frac{\partial E}{\partial w_{ab}^L} = \sum_i\sum_j \frac{\partial E}{\partial y_{ij}^L}\frac{\partial y_{ij}^L}{\partial x_{ij}^L}\frac{\partial x_{ij}^L}{\partial w_{ab}^L} = \sum_i\sum_j \frac{\partial E}{\partial y_{ij}^L}\frac{\partial y_{ij}^L}{\partial x_{ij}^L}\cdot\frac{\partial\left(\sum\limits_{a'=-\infty}^{+\infty}\sum\limits_{b'=-\infty}^{+\infty} w_{a'b'}^L \cdot y_{(is-a')(js-b')}^{L-1} + b^L\right)}{\partial w_{ab}^L}, \tag{10}$$

$$\forall a \in (-\infty,\ldots,+\infty), \forall b \in (-\infty,\ldots,+\infty).$$

On the other hand, decomposing the sum in the numerator by a and b, we have

$$\frac{\partial E}{\partial w_{ab}^{L}} = \sum_i \sum_j \frac{\partial E}{\partial y_{ij}^{L}} \frac{\partial y_{ij}^{L}}{\partial x_{ij}^{L}} \cdot y_{(is-a)(js-b)}^{L-1}, \ \forall a \in (-\infty,...,+\infty) \forall b \in (-\infty,...,+\infty). \tag{11}$$

For one feature map, only one offset can be used, which is "associated" with all the elements of that map. Accordingly, when adjusting the value of this offset, all the values from the map obtained during the backward propagation of the error should be taken into account. In this case (when using the signs of one offset for one map), taking into account the fact that

$$x_{ij}^{L} = \sum_{a=-\infty}^{+\infty} \sum_{b=-\infty}^{+\infty} w_{ab}^{L} \cdot y_{(is-a)(js-b)}^{L-1} + b^{L}, \tag{12}$$

finally we have

$$\frac{\partial E}{\partial b^{l}} = \sum_i \sum_j \frac{\partial E}{\partial y_{ij}^{l}} \frac{\partial y_{ij}^{l}}{\partial x_{ij}^{l}} \frac{\partial x_{ij}^{l}}{\partial b^{l}} = \sum_i \sum_j \frac{\partial E}{\partial y_{ij}^{l}} \frac{\partial y_{ij}^{l}}{\partial x_{ij}^{l}}. \tag{13}$$

4.2. Mask R-CNN

Mask R-CNN [16] is a convolutional neural network and is advanced in image segmentation and object segmentation. The concepts behind Mask R-CNN have undergone a step-by-step development through the architectures of several intermediate neural networks that solve different problems. Mask R-CNN was developed from Faster R-CNN, a region-based convolutional neural network, by adding a branch for predicting segmentation masks on each Region of Interest (RoI), in parallel with the existing branch for classification and bounding box regression.

It is also worth noting that, unlike Faster R-CNN, Mask R-CNN adds a third branch which outputs a mask of the object. The additional mask output differs from the class and box outputs by requiring the extraction of a much finer spatial location of the object. Mask R-CNN works by adding a branch for object mask prediction in parallel with the existing branch for bounding box recognition

$$L = L_{cls} + L_{box} + L_{mask} . \tag{14}$$

During training Mask R-CNN determines multitask losses on each sample RoI (14). The classification losses Lcls and boundary box losses L_{box} are identical to those defined in Faster R-CNN. Sigmoid per pixel is applied to the mask branches and L_{mask} is defined as the average cross-entropy loss of the binary masks.

The L_{mask} definition allows the network to generate masks for each class without competition between classes. Mask R-CNN relies on a special classification branch to predict the class label used to select the output mask. This allows us to separate mask and class prediction. This differs from the usual practice of using Faster R-CNN for semantic segmentation, which typically uses pixel softmax and multinomial cross-entropy loss. In this case, masks of different classes compete with each other. Experimentally, Mask R-CNN shows that this formulation is the key to obtaining good results for instance segmentation.

The mask branch of the network in Mask R-CNN is a convolutional network designed to create masks for the positive regions identified by the RoI classifier. Generated masks are represented by floating point numbers, which allow for greater detail than binary masks. The process of RoI pooling involves selecting a portion of a feature map and resizing it to a predetermined size, analogous to cropping and resizing a portion of an image. A graphic representation can be seen in Fig. 2.

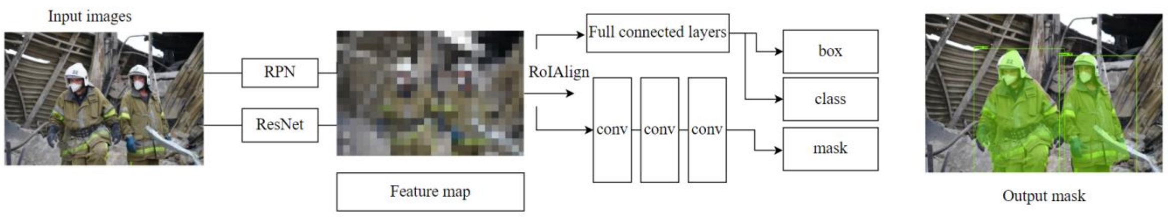


Figure 2: Graphical representation of Mask R-CNN

In this paper Mask R-CNN is used due to number of its advantages, specifically:

- Simplicity: Mask R-CNN is simple to train.
- Performance: Mask R-CNN outperforms all existing, single-model entries on many tasks.
- Efficiency: The method is very efficient and adds only a small overhead to Faster R-CNN.
- Flexibility: Mask R-CNN is easy to generalize to other tasks. For example, it is possible to use Mask R-CNN for human pose estimation in the same framework.

5. Experiment

This section describes the steps that were taken during this study. Blender, a professional free and open software for creating three-dimensional computer graphics, is used in this project. Creating a scene and arranging objects in Blender is not too complicated, and by sorting the objects in the collection it is easy to remove or put back those objects that were not involved in creating a mask and are not required to be detected by the neural network. It is a cross-platform application and consumes less memory and disk space compared to other 3D modeling programs. Blender uses OpenGL to draw its interface, so it looks the same on all platforms. Blender contains a wide range of tools, making it suitable for producing almost any kind of media production.

The advantages of 3D graphics are the following: high speed of creation, a wide variety and a wide range of tools for editing images. 3D models also make it easy to create annotations of the objects in a single click, which can significantly speed up both the annotation and training processes.

Appearance of the background may be changed in different ways. It's also possible to animate characters as it shown on the Fig. 3 or different objects which helps to generate many new images. In this project destroyed cars and houses is presented at Fig. 4. Destroyed apartment Fig. 5-a were used to fully replicate the catastrophic scene. Different weather conditions, such as rain is presented at Fig. 5-b and snow at Fig. 6, which will eventually make up the complete scene are used as well.

Figure 3: Animated 3D rescuers models

5.1. Annotations

Annotations are needed to mark the boundaries and contours of the required subject. These images can be used to train a neural network to recognize everything the researcher needs. From existing services to annotate photos:

- Annotate.net [4];
- RectLabel [5];
- Labelbox [6];
- LabelImg [7];
- SuperAnnotate [8];

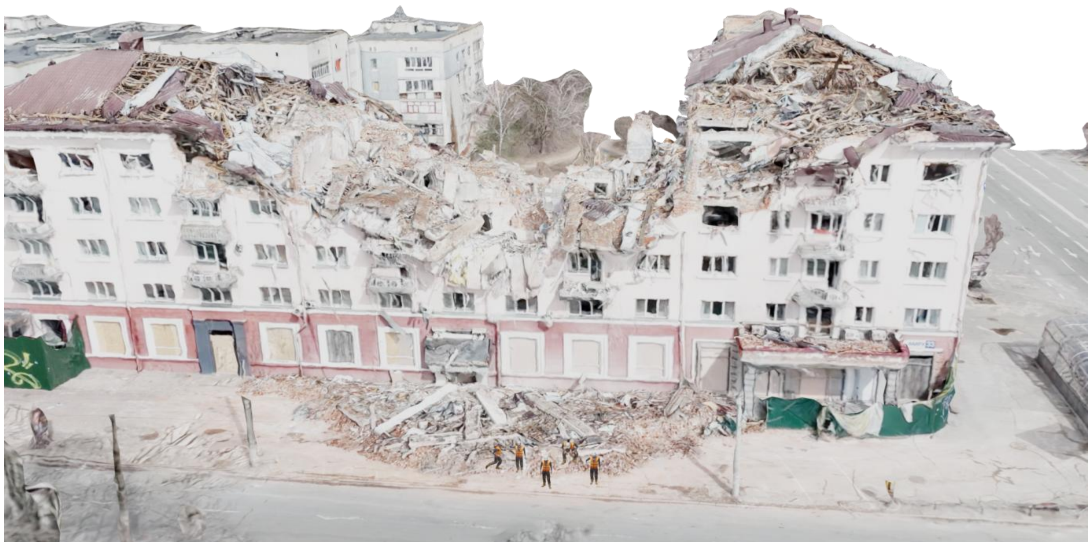


Figure 4: 3D model of a destroyed house

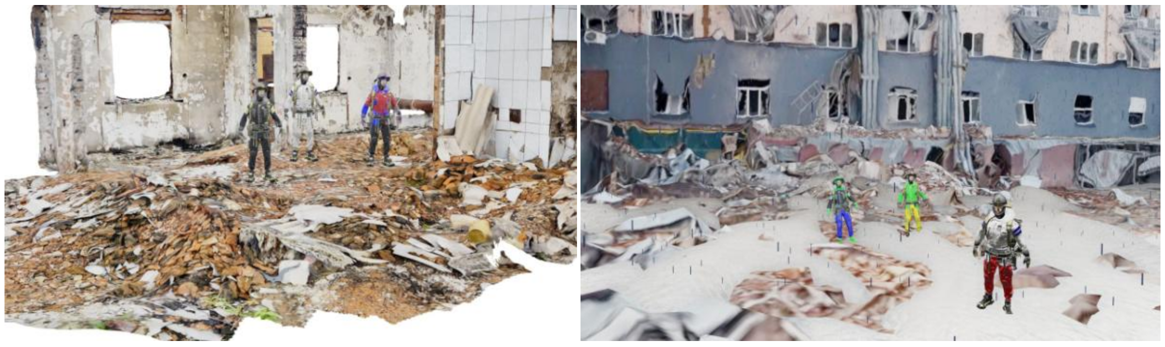

a) **b)**

Figure 5: 3D model of rescuers **a)** in a destroyed apartment; **b)** with rain and snow in Blender

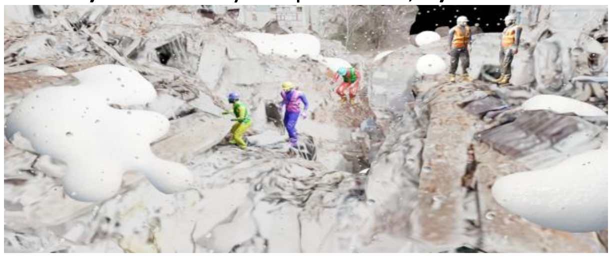


Figure 6: 3d model of the scene and Snowfall in Blender

- COCO Annotator [9]
- OpenCV Annotation Tool [10];
- ImageJ [11];
- GIMP [12];

In this paper VGG Image Annotator version 2.0.11 [13] is used. It is simple software for annotating images manually, easy for anyone to understand. VIA is simple and can be used without downloading, everything works with modern web browsers. The advantage of this software is that it is free. One can add a name and type and describe the selected area. A human can easily recognize known objects in the image. But our task is to streamline the monotonous work of putting points around the outline of an object and do it automatically. If a researcher is processing 10 photos and in each photo he needs to mark 3 large objects, it turns out that he needs to mark 30 areas, which may be equal to 200 or more points. It is a very monotonous and routine job to do with only 10 photos. To speed up the annotation process, an image and its masks are loaded into the program. The mask is a photo with a completely black background, where the object being studied is marked in white as it shown on the Fig. 7. In the program, points are put on the contour of the subject, the resulting file in json format is loaded into the VIA and the selected area appears on the required subject.

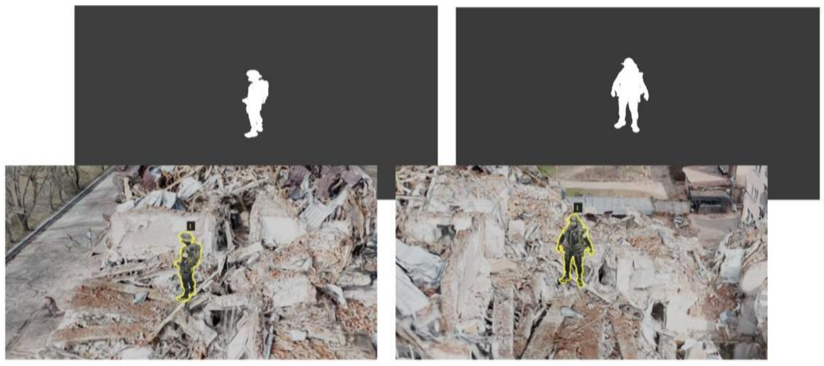


Figure 7: Example of masks and annotated images

5.2. Augmentation

For increasing the dataset diversity image augmentation is used examples are shown at the Fig. 8. The difficulty with annotations is that the photos are not always of high quality. Usually photos from everyday life are taken in a variety of places, it can be outside in the park or near the house or wherever, or photos taken inside, in an apartment or classroom, in a pool, absolutely anywhere. Often there is no right equipment like the newest camera with a super magnifying lens or special lamps for lighting. Sometimes the pictures are not of high quality, for example with glare, or lit from natural sources such as the sun. Very dark photos are also not considered to be the standard. The main idea is that most of the real photos are taken not in the specialized photo studios, but in the living conditions. This all greatly affects the quality of the models training and inference. Data scientists have to be prepared for the cases when the model will be used with real photos that are spoiled. Therefore, it makes sense to use augmentation for dataset preparation. The goal of this project is to train the neural network and give the photos as close as possible to the real life conditions with rain or snow. Augmentation allows to generate all sorts of distortions.

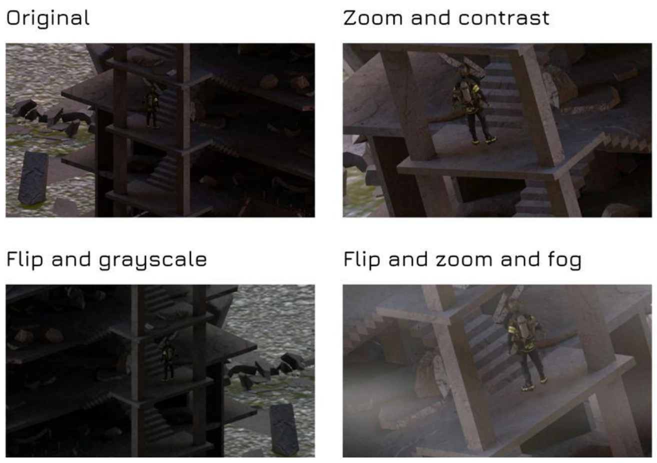


Figure 8: Example of the augmented images

After changing the photo, the points of the object borders are changed as well and have different coordinates, so the program sets them automatically in the following way: an image that needs to be augmented is loaded into a program; key points like contoured object of the study are added to the image; the image is cropped, flipped, highlighted, contrasted, zoomed of damaged with Gaussian blur.

Gaussian blur reduces the detail of images and increases the blurred elements. The use of Gaussian blur results in the reduction of high-frequency components of the image. This means that the Gaussian blur is a low-pass filter. In this blurring method, the Gaussian function is used to calculate the transformation applied to each pixel of the image. The Gaussian function in one dimension

$$G(r) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-r^2}{2\sigma^2}} \tag{15}$$

In two dimensions it is the product of two Gaussian functions, one for each dimension.

$$G(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}} \tag{16}$$

where u, v are point coordinates, and σ is the standard deviation of the normal distribution. When applied in two dimensions, this formula gives a surface with contours that are concentric circles with a normal distribution with respect to the central point. The new value of each pixel is set equal to the weighted average value of that pixel's neighborhood. The value of the original pixel gets the highest weight (having the highest Gaussian function value), and neighboring pixels get lower weights as their distance to the original pixel increases. This results in a blur that preserves borders and edges better than other more uniform blur filters. All the image processing can take place in different order and a variety of distortions can be applied.

Importantly, augmentation helps prevent retraining on some very similar data. Augmentation creates conditions so that images are as different as possible and prevents the neural network from being too sensitive to unsimilar images, which would result in a large number of incorrectly recognized objects. Using augmentation, the neural network learns much better and increases the accuracy of object recognition in the photo.

Fig. 9 shows the process from the appearance of the problem to the training of the neural network. First the problem to be solved appears, and then the solution is step by step. If there are not a lot of images, then 3D models are used. We just need to find or create a 3D model of the object which is being investigated and a 3D model of the environment and create the design. Then render the image and if that is enough, it's possible to move on to the next step. With 3D models it is easy to make masks, and with software calculations we can find mask boundaries and expose points. The coordinates are saved in a json file for further processing and saving the object boundaries. This is how annotations are created. After, the annotated images are made augmentation, bloating a sample of images, for example from 40 photos, it is possible to make 1000. This is done to improve neural network learning. The next step is to start training the neural network. And after several epochs of training obtain the results on which the neural network will recognize objects in real images.

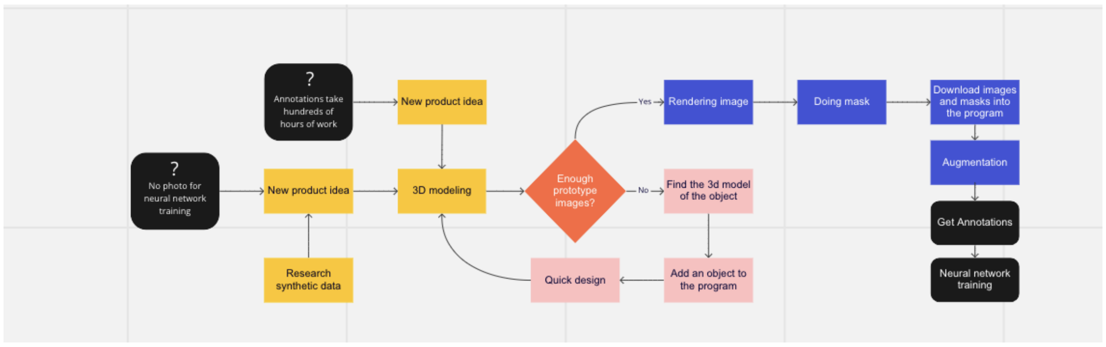


Figure 9: Process description diagram

We want to investigate the efficiency of the object detection neural network, after training on synthetic data. The process took about 44.2 seconds to annotate 1000 images, including also the augmentation of the whole dataset, which greatly accelerated the sample generation process.

The neural network was trained on a test set with 3D images. In the end, to train the neural network, we had 100 images created from the 3D models and another 912 images that were the result of augmentation. Fifteen images were selected for testing, most of which showed good prediction results. The neural network itself was run using a cloud-based code tool, Google Colab. This is a very convenient tool for working with code in Python, which is simple and easy to use. The main feature of Colab is the free powerful GPUs and TPUs that can be used not only for basic data analytics, but also for more complex machine learning research. What takes CPUs hours to compute, a GPU or TPU can do in minutes or even seconds. GPUs themselves are expensive, and not everyone can afford them. Google Colaboratory provides 12 hours of free, uninterrupted use. The only thing needed is to have a Google account.

6. Results

Mask-R-CNN neural network was trained on a dataset containing synthetic images and the object is highlighted by the contour using a mask. After training, it can detect objects at real images.

The neural network was trained with using cloud services, which allowed the use of Google's TPU processors. After training, the neural network finds objects in the image and highlights them using a mask. After testing the neural network, it turned out that out of 150 real photos, about 40 are not recognized accurately. All of the objects at Fig. 10 are found correctly. An inaccuracy appeared in Fig. 11-a where a part of the dog's paw was selected by the rescue mask, but it was not selected as a separate object. At Fig. 11-b several objects are highlighted that are not the subject of the study. This can be solved by increasing the training of the neural network by more steps. As previously investigated [17], the diversity of the 3D model sample could be significantly increased by 5.5 percent, which proves the gain from adding more shape variations to the training data.

In our work, we also evaluated the accuracy of the classification models on the dataset. For this purpose, we calculated the F-score [18], also known as F1-score. In this vein, we combined model accuracy and recall into a single metric. To do this, we calculated the harmonic mean of these two metrics

$$F = \frac{2}{\frac{1}{recall} \times \frac{1}{precision}} = 2 \times \frac{precision \times recall}{precision + recall} \qquad (17)$$

$$precision = \frac{tp}{tp + fp} \qquad (18)$$

$$recall = \frac{tp}{tp + fn} \qquad (19)$$

Precision refers to the ratio of true positive examples to the total number of examples classified as positive by the model. On the other hand, recall, also known as sensitivity, is the ratio of examples classified as positive to the total number of positive examples. The variables tp, fn, and fp refer to the number of true positives, false negatives, and false positives, respectively, classified by the model.

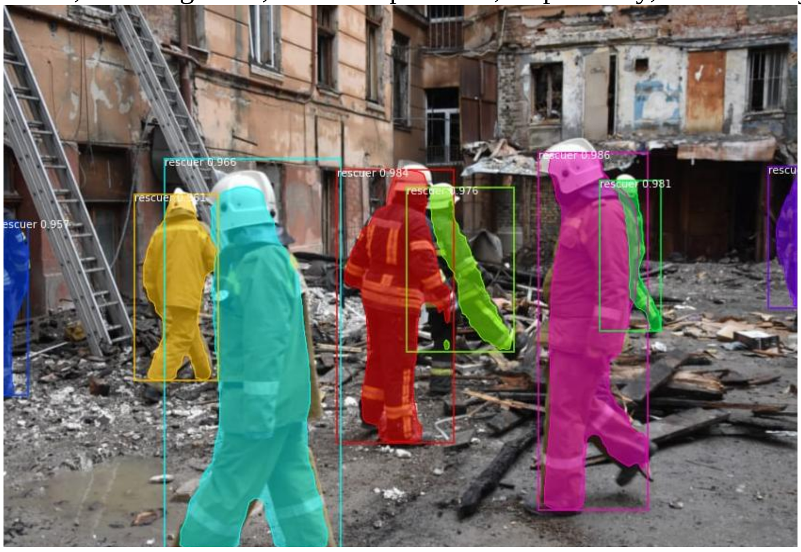
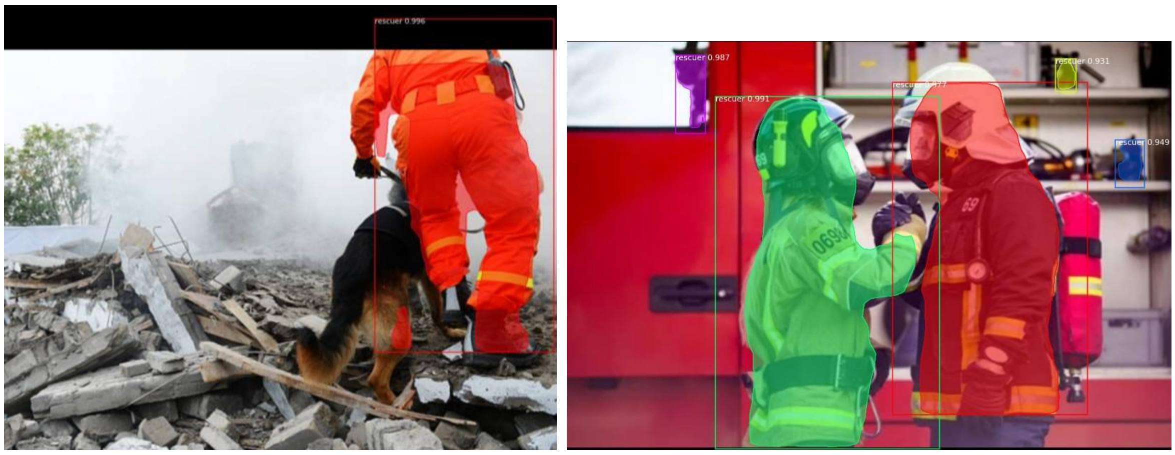


Figure 10: Result of training neural network with synthetic dataset



a) b)

Figure 11: Result of faulty object recognitions

Following formulas (17), (18) and (19), we can calculate an approximate F1-score for the sample to better understand the result. If we take the average values for the whole sample, precision is about 0.83 and recall about 0.75, resulting in a numerical value of F1 equal to 0.78 for one class.

One of the simplest indicators for model evaluation is accuracy, which is computed by dividing the number of correctly classified examples by the total number of examples. However, accuracy does not consider class imbalance and the varying costs of false negatives and false positives. Formula (20) can be used to calculate accuracy, which is helpful for understanding the experiment's methodology, although the value may not always reflect the full effectiveness of the approach.

$$accuracy = \frac{tp + tn}{tp + tn + fp + fn} \qquad (20)$$

As a result of the calculation we can say that the accuracy is 0.73, which corresponds to reality.

7. Conclusion

We used the Mask-R-CNN neural network because it is a fast and reliable neural network for detecting objects in the image. In a situation where there is no possibility to take real life pictures for the dataset, it makes sense to use synthetic data in programs such as Blender. The resulting images are needed to be annotated. For thousands of photos it can take days, weeks or even months, making points on the contour of the object is especially difficult if it does not have a perfectly straight shape. Each of the services listed in the example in section 5.1 has its own characteristics and disadvantages, so it all depends on the requirements for annotations and how accurate they should be. But working on the project and considering all the advantages of these services, we came to the conclusion that it is still very monotonous work, dotting the contour. Proposed approach allows developers to make annotations in just a couple of minutes. The resulting synthetic dataset was used to train Mask R-CNN neural network. The work has produced promising results in a short period of time. The pre-trained neural network is easily trained to recognize new objects. In 110 out of 150 photos the neural network is able to correctly find the object. In 73% of the photos all objects were recognized correctly, which is a promising result.

The proposed method can be used to recognize rescuers at disaster scenes. For example, if possible for neural networks that can recognize rescuers at house collapse areas. This research is so broad and relevant that it can be used in many different ways. It could be incorporated into the development of the virtual reality glasses industry. For example, the operator coordinating the rescuers' work would be allowed to see literally in front of him, in real time, the location and position of the rescuers. The work of rescuers is relevant during the war in Ukraine, but also after the earthquake in Turkey and other countries. The work of rescuers is dangerous and this project is aimed help them do their job more safely and save more lives.

8. Reference

[1] E. Wood, T. Baltrušaitis, C. Hewitt, S. Dziadzio, Thomas J. Cashman, J. Shotton, Fake It Till You Make It: Face Analysis in the Wild Using Synthetic Data Alone, Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 3681-3691. URL: https://microsoft.github.io/FaceSynthetics/

[2] Zhixiang Wang, Glauco Lorenzut, Zhen Zhang, Andre Dekker, Alberto Traverso,Applications of generative adversarial networks (GANs) in radiotherapy: narrative review, Precision Cancer Medicine, 2022, DOI: 10.1002/acm2.13359,URL: doi.org/10.1002/acm2.13359

[3] Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, Debbie Rankin, Synthetic data generation for tabular health records: A systematic review, Elsevier, 2022, 28-45, DOI: 10.1016/j.neucom.2022.04.053, URL: https://doi.org/10.1016/j.neucom.2022.04.053

[4] https://annotate.net/ , 2023 , URL: https://annotate.net/

[5] RectLabel, Rasterize, Inc, California, USA, 2019. URL: https://rectlabel.com/

[6] Labelbox, Labelbox, Inc, San Francisco, USA, 2018. URL: https://labelbox.com/

[7] T. Lin , LabelImg, Taipei, Taiwan, 2015. URL: https://github.com/tzutalin/labelImg

[8] SuperAnnotate, SuperAnnotate AI, Inc, California and Massachusetts, United States, 2019. URL: https://superannotate.com/

[9] J. Broks, COCO Annotator, released on GitHub, San Francisco, USA, 2017. URL: https://github.com/jsbroks/coco-annotator

[10] C. Dahms, Microcontrollers And More. URL: https://github.com/MicrocontrollersAndMore/OpenCV_3_License_Plate_Recognition_Python/blob/master/AnnotationTool.py

[11] W. Rasband, ImageJ, National Institutes of Health (NIH). URL: https://imagej.nih.gov/ij/

[12] P. Mattis, S. Kimball, GIMP ,University of California Berkeley. URL: https://www.gimp.org/

[13] Abhishek Dutta and Andrew Zisserman. 2019. The VIA Annotation Software for Images, Audio and Video. In Proceedings of the 27th ACM International Conference on Multimedia (MM ’19), October 21–25, 2019, Nice, France. ACM, New York, NY, USA, 4 pages. URL: https://doi.org/10.1145/3343031.3350535.

[14] ChatGPT, OpenAI, San Francisco, 2020. URL: https://chat.openai.com

[15] R. Girshick. Fast R-CNN. In ICCV, 2015. 1, 2, 3, 4, 6

[16] He K, Gkioxari G, Dollár P, Girshick R. Mask R-CNN. URL: https://arxiv.org/abs/1703.06870

[17] D. Zeiler, Rob Fergus,Learning Deep Object Detectors from 3D Models, European Conference on Computer Vision (ECCV), 2014.

[18] Y. Sasaki, The truth of the F-measure, 2007, URL: https://www.cs.odu.edu/~mukka/cs795sum09dm/Lecturenotes/Day3/F-measure-YS-26Oct07.pdf