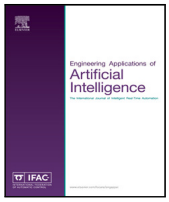
Engineering Applications of Artificial Intelligence 126 (2023) 106942

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Survey paper

Instance segmentation of stack composed of unknown objects

Michał Czubenko [a,b,*], Artur Chrzanowski [a,b], Rafał Okuński [a]

[a] Intema Sp. z o. o., Siennicka 25a, Gdańsk, Poland
[b] Department of Decision Systems and Robotics, Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Narutowicza 11/12 Gdańsk, Poland

ARTICLE INFO

MSC:
62H35
62M45

Keywords:
Segmentation techniques
Pick-and-place operation
Deep neural networks

ABSTRACT

The article reviews neural network architectures designed for the segmentation task. It focuses mainly on instance segmentation of stacked objects. The main assumption is that segmentation is based on a color image with an additional depth layer. The paper also introduces the Stacked Bricks Dataset based on three cameras: RealSense L515, ZED2, and a synthetic one. Selected architectures: DeepLab, Mask RCNN, DEtection TRansformer, Geometry-Aware Instance Segmentation, Shapemask, Synthetic Depth Mask RCNN, Synthetic Fusion Mask RCNN (SF-Mask), Unseen Object Instance Segmentation (UOIS), Unseen Object Clustering (UOC), and You Look Only At Coefficients, have been tested on various datasets. The results show that the best architectures for stacked elements segmentation are UOIS, SF-Mask, and UOC.

1. Introduction

Progress in machine vision has been developing for a long time due to the increasing resolution and frequency of image acquisition. Currently, the highest values which can be found on the websites of machine vision stores, e.g. *edmundoptics*, are 31.4 Mpx — 6480 × 4860 pixels, and 750 fps at 640 × 480 pixels resolution. Additionally, vision processors are integrated with cameras. Their possible tasks are to pre-process, denoise and undistort the captured images from the sensor on the fly. Currently, more specialized integrated circuits – the so-called neural processing units – may also be included in cameras. They allow for running specified neural networks (e.g. for the image recognition task or stitching together images from multiple sources). Moreover, optical systems are multiplied on a single device. This leads to stereo-vision (with appropriate processing), quick change of focal point and sharpening of selected image elements.

At the same time, image processing algorithms, especially those powered by artificial neural networks (ANNs), have also evolved. Among the many applications of (single/static) image processing, the following image processing issues related to artificial neural networks can be distinguished:

- object recognition/image classification — usually, recognition is processed on a cropped image with a single object; another option is to classify the whole image to a certain category (Czubenko et al., 2022; Krizhevsky et al., 2012)
- (specific) object identification – recognition of certain instances of the same category of objects – such as human faces (Coşkun et al., 2017)
- object inspection/anomaly detection — searching for small, minor defects in the image of a certain object (Staar et al., 2019)
- image segmentation — dividing an image into certain segments (e.g. background/foreground) (Liu et al., 2019; Minaee et al., 2021)
- image restoration — e.g. noise removal, super-resolution (Park et al., 2018)
- image generation — from, for example, noise, semantic description, or sketches of a completely new image can be created e.g. of a non-existing face/landscape (Viazovetskyi et al., 2020; Qu et al., 2019)
- image/scene reconstruction — fixing minor defects (Popov et al., 2020)
- style transfer — applying a certain style (e.g. artistic such as van Gogh) into an image (Gatys et al., 2016)
- semantic description — adding a valid caption in a natural language (Zhang et al., 2018; Hossain et al., 2019).

1.1. Image segmentation

This article mainly discusses the segmentation issue, i.e. determining the pixel membership to a certain segment (Kirillov et al., 2019).

* Corresponding author at: Department of Decision Systems and Robotics, Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Narutowicza 11/12 Gdańsk, Poland.
E-mail address: micczube@pg.edu.pl (M. Czubenko).

https://doi.org/10.1016/j.engappai.2023.106942
Received 2 August 2022; Received in revised form 27 February 2023; Accepted 3 August 2023
Available online 23 August 2023

(a) Original image. (b) Instance segmentation (c) Panoptic segmentation

Fig. 1. Example of image segmentation using Mask RCNN (Region-Based Convolutional Neural Network) from Detectron 2 library (Wu et al., 2019) framework on image created by Tamanoeconomico, CC BY-SA 4.0, via Wikimedia Commons. It should be noted that the used image does not belong to the COCO dataset on which the Detectron 2 network was trained.

The concept of image segmentation has been known for a long time (Yu et al., 2018; Ohta et al., 1978). An example of the formal definition of classical image segmentation is presented in Serra (2006). However, a shorted definition can be applied to the problem; following Garcia-Garcia et al. (2017), we may define the label space: $\mathcal{L} = \{l_1, l_2, \ldots, l_k\}$ whose interpretation depends on the segmentation type. Note that, in certain segmentation cases, the number of different labels (k) may be unknown, and we may add l_0, which corresponds to the background. We also assume that the existing set of variables is the pixel matrix — image $\mathcal{X} = \{x_{1,1}, \ldots, x_{w,h}\}$ where w and h correspond to the width and height of the image. The problem may be defined as finding a way to assign a label to each of the variables (Garcia-Garcia et al., 2017).

Currently, three types of segmentation task can be discussed: semantic (pixels belonging to a certain object type), instance (pixels belonging to individual instances of one, specific object type), and panoptic segmentation (pixels belonging to a certain object type with a distinction between the instances). The difference between instance and panoptic segmentation is shown in Fig. 1, while semantic segmentation would classify each person in Fig. 1(c) as belonging to the same category.

It should be noted that the issue of segmentation is most often applied to images of the surrounding environment created from a certain perspective. One of the most popular examples is the case of semantic segmentation applied to the road image by an autonomous driver (Kaymak and Uçar, 2019; Treml et al., 2016). A very similar case is the segmentation of mobile robots' environment (Li and Birchfield, 2010; Li et al., 2020) for navigation purposes. Automatic segmentation of satellite/aerial images is also being developed (Chai et al., 2020). The least studied case is segmentation of small objects picked up by production robots — the so called Pick&Place task (Schnieders et al., 2019; Ainetter and Fraundorfer, 2021).

On the other hand, the above-described cases can be divided according to the type of the processed image. In most cases, regular color – RGB (Red Green Blue) – images are the basis for segmentation. However, in the case of an autonomous car and object, much better efficiency can be achieved by using the spatial depth obtained by fusing the LIDAR (Laser Imaging, Detection, and Ranging) data or simply by using cameras with a depth channel (RGBD cameras), which still counts as image segmentation. While, another approach, also currently being developed, is to segment the points from the point cloud obtained by the appropriate sensor (Zhang et al., 2019a).

1.2. Motivation

Nowadays, we see a growing trend regarding the automation of production stations. In many companies, robots are replacing humans in the workplace. Increasingly, entrepreneurs are also starting to make use of cooperative robots (so-called cobots). In simple terms, these are robots that, among other things, detect approaching humans in the workspace or even collisions and react accordingly — they may automatically stop, or reduce speed and change the trajectory (Czubenko and Kowalczuk, 2021). Meeting the technical specification ISO/TS 15066, they allow for cooperation with a human at a workstation (Masłowski and Czubenko, 2019). So far, for the robotization of the station to be profitable, the robot had to produce/transfer the same/similar objects in a much shorter time than a human, 24 h a day. Robotization of stations producing short series was not profitable. This situation has begun to change thanks to the emergence of dynamic robotic stations with the possibility of quick and automatic reconfiguration (Rossi et al., 2020; Kousi et al., 2021).

The robot must be equipped with several sensors and technologies for detecting and recognizing objects in its working space, to ensure the reconfiguration ability of the station. One of the key technologies is the segmentation of the image of the working space. The use of RGBD sensors will make it possible to determine the spatial coordinates of objects near the robot. Thanks to the combination of both technologies, the robot can pick up detected objects.

The main goal of the work related to this article is to develop an autonomous gripping station equipped with a 6-DoF (Degrees of Freedom) robot. Based on the image from RGBD cameras and using neural networks, the robot should be able to find and pick up previously unknown objects located on a heap of objects. In turn, based on the algorithms developed as part of further work, the system should be able to sort/cluster these objects. Enabling picking up unknown objects and then their clustering will ultimately enable dynamic adaptation to the changing conditions of the robotic station.

From a bibliometric point of view, Keyword *image segmentation* is mentioned in ≈150,000 articles according to the Web of Science portal. The number of articles per year has increased from 500 items around 2010 to approximately 1000. In the case of *instance segmentation*, a similar trend is also observed, but on a smaller scale, from about 10 articles/year in 2010 to about 50 currently. So, you can see that the topic is getting more and more popular, although definitely not yet as popular as pure semantic segmentation. Currently, about 3200 articles on the subject have been published.

1.3. Contribution

This article aims to compare and review artificial neural networks for the purpose of instance segmentation using images taken from depth cameras. The goal set for the presented neural networks is to segment a stack of unknown objects. The stacks will be mainly heterogeneous (different instances of different object types). Note that, with regard to industrial automation, there are some closed solutions (such as PickIt, Photoneo, Zivid, and Solomon3D), composed of very advanced and expensive 3D cameras, for the task of picking up homogeneous objects from a stack. In our article, open solutions will be evaluated. Different

Table 1
Comparison of stack datasets. Note that the column named *objects* represents a number of different objects used in the dataset, while the *scenes* column shows how many different backgrounds were used. w (width) × h (height) represents the resolution of the RGB image, while *depth* shows the resolution of the depth map. *Items* are the number of samples, divided into the *train*, validation (*val)* and *test* parts of the dataset.

dataset name	*sensor*	*objects*	*scenes*	$w \times h$	*depth*	*items*	*train*	*val*	*test*
OCID	ASUS Pro Xtion	4	96	640 × 480	640 × 480	2390	1434	239	717
WISDOM real	Photoneo PhoXi	50	1	1032 × 772	1032 × 772	400	240	40	120
OSD	Kinect equivalent	Many	~2	640 × 480	640 × 480	111	66	11	34
TOD	synthetic	Many	Many	640 × 480	640 × 480	220 000	132 000	22 000	66 000
STIOS	Stereolabs ZED	15	8	2208 × 1242	2208 × 1242	192	115	19	58
SBD-S	virtual camera	Many	9	1000 × 750	1000 × 750	8427	5056	842	2529
SBD-L	RealSense L515	70	1	1100 × 540	1100 × 540	290	174	29	87
SBD-Z	Stereolabs ZED2	70	1	1250 × 620	2208 × 1242	280	168	28	84

neural networks which require D (Depth — the distance from the sensor), RGB, RGBD (colors and depth), or XYZ (measurements on a 3D axis) channels at the input will be presented and evaluated. With regard to the above, a similar survey does not currently exist.

This article presents an experimental overview and comparison of neural networks that allow the segmentation of a heap of objects. In detail, we show an overview of the datasets used for the segmentation task. We focus on sets based on RGBD images, especially heaps of objects that can be picked up by a production robot. From the presented datasets, five sets were selected for testing purposes. At the same time, our own three datasets based on Intel RealSense L515, Streolabs ZED2, and synthetic cameras were presented. Then, ten different architectures were selected and tested using the provided (by the authors) weights (without additional training). Based on these tests, three outstanding architectures were selected, and one with low indications (based on RGB images) for further testing. The selected architectures were properly trained using portions of the selected datasets, using multiple balancing methods for the training data, and appropriately tested. Complementarity tests of selected architectures were also carried out, with an introduction to complementarity metrics.

In short, new RGBD stacked objects datasets have been created and proposed. Ten selected neural segmenting architectures have been tested, and four of them with additional learning. Different methods for coping with the unbalanced mix of datasets have also been tested. Complementarity metrics have also been proposed.

1.4. Structure

Firstly, available datasets in the context of segmentation are presented. The dataset review shows the most commonly used 2D (RGB) and 2.5D/RGBD segmentation datasets, especially the datasets used during the tests. After that, the theoretical background showing the state of the art in the panoptic segmentation task can be found. The metrics used in the evaluation are also presented in detail. A brief characterization of the tested neural networks is included in the next section. The tests which were performed with the weights proposed by the authors, as well as those carried out on parts of the datasets after additional training sessions, are presented. Additionally, evaluation of the way in which one of the datasets copes with the domination is carried out. The conclusions can be found at the end of the paper.

2. Segmentation datasets

An extensive review of the classical (RGB) segmentation datasets can be found in Garcia-Garcia et al. (2017) and Minaee et al. (2021), while the most comprehensive review of RGBD/2.5D datasets is presented in Firman (2016). With regard to the source, the datasets can be divided as 2D — color (RGB) only, 2.5D — RGBD, and 3D — point clouds (PCL). On the other hand, with regard to the context, we can distinguish generic objects, piles/stacks of objects, indoor, outdoor, urban (driving) environments, aerial/satellite images, and synthetic scenes/objects. Here, we present the most commonly used 2D, 2.5D datasets, and the sets that are used in tasks performed on stacks of objects.

2.1. Color-only datasets

The COCO dataset (Common Objects in COntext) (Lin et al., 2014) "is a large-scale object detection, segmentation, and captioning dataset". COCO has several purposes: object segmentation, image description, and poselet estimation. It contains more than 200k (200,000) labeled images of variable resolution.

PASCAL VOC (Pattern Analysis, Statistical modeling and ComputAtional Learning, Visual Object Classes challenge) (Everingham et al., 2015) is one of the most popular datasets in the context of image segmentation. The dataset consists of a few other datasets (namely: TU Darmsad, UIUC, VOC, Caltech, MIT, TU Graz, 101 Objects) and contains about 2.9k images. Most of the images are provided with an annotation, a bounding box, and a pixel segmentation mask.

The Cityscapes Dataset (Cordts et al., 2016) is "a large-scale dataset which contains a diverse set of stereo video sequences recorded in street scenes from 50 different cities, with high quality pixel-level annotations of 5k frames in addition to a larger set of 20k weakly annotated frames". The Cityscapes Dataset is designed primarily for the urban segmentation task. The newest version of Cityscapes also contains 3D bounding boxes (Gählert et al., 2020).

Another dataset whose influence is currently growing is the Open Images Dataset — OID (version 6) (Benenson et al., 2019). Version 5 of this dataset also contains "segmentation masks for 2.8M object instances in 350 classes". Currently, the whole dataset contains 1.9M images (Krasin et al., 2017).

Among the numerous other datasets, we can indicate the following sets used in segmentation: PASCAL Context, ADE20K/MIT Scene Parsing (SceneParse150), SiftFlow, Stanford Background, Berkeley Segmentation Dataset (BSD), Youtube-Objects, CamVid, Semantic Boundaries Dataset (SBD), SYNTHIA, and Adobe's Portrait Segmentation.

2.2. RGBD datasets

Mapillary, a company that creates street maps, has several datasets of its own (Porzi et al., 2021). The Metropolis dataset (RGB with LIDAR) was taken in an urban environment for the purpose of "object recognition (2d/3d and tracking), 3d reconstruction, SLAM, image-based depth estimation, relocalization, image- and patch-based matching, image retrieval, depth estimation, etc.". It contains about 27k images. The Mapillary Planet-Scale Depth (MPSD) is a dataset which provides depth and color images of certain landscapes (about 750k annotated images). The Mapillary Vistas Dataset is "a large-scale street-level image dataset containing 25k high-resolution images annotated into 66/124 object categories, of which 37/70 classes are instance-specific labels".

The NYU-Depth V2 dataset was recorded by a Microsoft Kinect camera (Bousmalis et al., 2017). It consists of sequences from "a variety of indoor scenes", and contains about 1.5k images in 450 different scenes. All the objects have their semantic labels and instance numbers.

The SUN RGB-d (Scene UNderstanding Benchmark Suite) provides 10k of RGBD images (Song et al., 2015) of the indoor environment. The scale of objects is similar to that in PASCAL VOC. It is dedicated to segmentation and 3D bounding box prediction.

(a) SBD-L (RealSense L515). (b) SBD-S (virtual). (c) SBD-Z (ZED2). (d) OCID (ASUS pro Xtion).

(e) OSD (Kinect equivalent). (f) STIOS (ZED). (g) TOD (synthetic). (h) WISDOM (Photoneo PhoXi).

Fig. 2. Color image samples from all datasets used in our tests. The image ratio (width/height) is the original. To use the images for inference with neural networks, we should resize them and change the ratio. The names of the datasets are presented in the sub-captions as well as the camera used for image acquisition.

Among other datasets, we should mention ScanNet (Dai et al., 2017) — an RGBD video dataset. It contains about 1.5k scans of indoor scenes and about 2500k different RGBD views. It has been designed for "3d scene understanding tasks, 3d object classification, semantic voxel labeling, and CAD model retrieval".

KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute) also has several different datasets (Alhaija et al., 2018). They are designed for different tasks, such as semantic and instance evaluation, tracking, depth completion, and prediction, etc. KITTI's semantic segmentation dataset consists of 200 well-annotated images.

The Stanford 2D-3D-S (Armeni et al., 2017) dataset presents 6 large-scale indoor environments in different modalities. It contains 70k RGBD images with their semantic and geometric annotations.

Berkeley B3DO (Janoch, 2012) is a dataset where all the images were acquired using a Microsoft Kinect sensor. The dataset contains about 850 images with both the color and depth of the indoor environment, with different perspectives.

The RGB-D Object/Scenes Dataset (Lai et al., 2011) consists of RGBD scans of 300 "common household objects" obtained by MS Kinect. It also includes 22 annotated video sequences from different indoor environments.

2.3. RGBD stack/heap/pile datasets

All the above datasets are valid from both a segmentation and depth processing point of view. However, most of them present internal or external environments from the robot's/car's perspective. Thus they are not sufficient for the task of stack segmentation. With regard to the subject of this article, a few more datasets will be presented.

The Object Clutter Indoor Dataset (OCID[1]) (Suchi et al., 2019) presents 96 cluttered scenes. Each scene contains RGBD images with pixel-wise annotations. The scenes are acquired from two different perspectives (top/bottom), and for two different grounds (table/floor). Even though the OCID dataset was designed for an indoor mobile robot, it also contains images created by "adding one object after the other" into a disorder similar to a stack.

Another dataset which contains certain objects on a table from one perspective is the Object Segmentation Database (OSD[2]) (Richtsfeld et al., 2012). It contains images of stacked boxes and other cylindrical objects with multiple occlusions. The images were acquired by a Microsoft Kinect or Asus XtionPRO (the authors did not clearly specify the device). Note that, in the original paper (Richtsfeld et al., 2012), the authors present a surface segmentation method of unknown objects, which uses surface patches and support vector machines (SVM).

The STereo Instances on Surfaces Dataset (STIOS[3]) (Durner et al., 2021) is a dataset designed for robotic applications, especially object manipulation. The pictures were obtained with the use of two cameras: an rc_visard and a ZED stereo camera, in 8 different environments: office carpet, workbench, white table, wooden table, conveyor belt, lab floor, wooden plank, and tool cabinet. The original paper of the authors presents a novel stereo-based approach for the Unknown Object Instance Segmentation (UOIS) architecture for a robotic vision called the Instance Stereo Transformer (INSTR).

The Tabletop Object Dataset (TOD[4]) (Xie et al., 2020) contains 40k synthetic scenes with some objects seen from different points of view. Each image was generated by the PyBullet engine using different objects (5–25 instances), randomly placed on some surface (a table or another object) from the ShapeNet dataset. The perspective of the view is random, at a distance between [0.5 and 1.2] meters from the table, and rotated at a certain random angle within the range of [−12°, 12°].

The Warehouse Instance Segmentation Dataset for Object Manipulation (WISDOM[5]) (Danielczuk et al., 2019) has two parts — a simulated and a real one. The WISDOM-Real part was prepared using 50 different objects to create 400 different stacks, while the simulated one contains 50k images with 1.6k random objects. These datasets are perfect for our processing, since the view perspective is strictly above the stack.

The heap dataset list should also include our own dataset called the Stack Bricks Dataset (SBD). It has been divided into three parts according to the type of camera used: Stereolabs ZED2 (Z), Intel Realsense L515 (L), and synthetic (S), generated by zpy (Ponte et al., 2021). Details of all the used stack datasets are shown in Table 1. Note

[1] https://www.acin.tuwien.ac.at/en/vision-for-robotics/software-tools/object-clutter-indoor-dataset.

[2] https://www.acin.tuwien.ac.at/en/vision-for-robotics/software-tools/osd.

[3] https://www.dlr.de/rm/en/desktopdefault.aspx/tabid-17628.

[4] https://github.com/NVlabs/UnseenObjectClustering.

[5] https://sites.google.com/view/wisdom-dataset/home.

that one item/image set (here and after) is understood as a single slice of data (depth image, color image/images, point cloud, ground truth). Sample random images from the datasets presented in this section are shown in Fig. 2.

2.4. Datasets used in tests

All the datasets presented in the previous section were used in the tests we conducted. However, they differ significantly in size. For example, the OSD dataset, with no more than one hundred samples is eclipsed by the TOD dataset, where tens of thousands of samples are present. That fact was taken into consideration in order to obtain an unbiased analytic result. Consequently, the maximum evaluation size was set at 1000 samples for each dataset.

All the used datasets were divided into training, validation, and test parts with the same proportion: 60:10:30%. The detailed numbers are shown in Table 1.

It should be mentioned that some datasets contain significantly more labeled objects than others. The SDB datasets was created with a robot grasping small elements from a cluttered stack in mind. Therefore, there are dozens of objects in each sample. The SDB datasets may compensate for their small number of samples with the number of objects in a single item.

3. Theoretical background

In general, among the existing methods of semantic/panoptic image segmentation, we can distinguish (Yu et al., 2018):

- classical image processing, basic feature-based methods and contextual models
- weakly- and semi-supervised methods (image annotations or bounding boxes with annotations)
- fully (deeply) supervised methods (trained on datasets with full pixel membership).

Among the classical segmentation methods based on image analysis, methods/algorithms such as thresholding (both manual and automatic), random walker, active contour, watershed, means-shift, normalized cut, Turbopixel, SLIC (Simple Linear Iterative Clustering), Chan-Vase, and Felzenszwaib (graph-based) can be mentioned (van der Walt et al., 2014). Such methods are generally used in the case of non-semantic segmentation tasks, especially those without labeling. As a rule, however, these methods are quite insufficient for complex images. Nevertheless, they make it possible to define/obtain so-called superpixels, i.e. similar and usually interconnected pixels (Yu et al., 2018). At the same time, one can also list classifiers (e.g. Bayes, Random Decision Forests, Support Vector Machine, etc.) based on the visual features of certain regions (e.g. histogram) or descriptors (SIFT — Scale Invariant Feature Transform, HOG — Histogram of Oriented Gradients, FAST — Features from Accelerated Segment Test, BRIEF — Binary Robust Independent Elementary Features, ORB — Oriented FAST and Rotated BRIEF, etc.)

At a slightly higher level of abstraction, methods can be found that use contextual models such as CRF (Conditional Random Field) or MRF (Markov Random Field), and their modifications. Both methods are based on an undirected graph created from pixels. These methods may also use superpixels, classically designated features or descriptors in some way. On the other hand, there are also methods based on the mixture of CRF and Convolutional Neural Networks (CNN) (Shen et al., 2017). In both cases, classical and contextual, the implementation of a specific task requires knowledge of what specific features/descriptors should be applied to a given case (Hu et al., 2018).

Weakly- and semi-supervised segmentation methods rely on particular bounding boxes or categories assigned to an image. Thus, the segmentation problem can be decomposed to separate the background from pixels belonging to the object. One of the first approaches to the reformulated problem can be found in Vezhnevets and Buhmann (2010), where the authors combined Multiple Instance Learning (MIL) with Semantic Texton Forest (STF) — a feature-based method for semantic image segmentation. Classical methods such as SVM (Andrews et al., 2002) can also be used as an extension of MIL. The approach which uses the Expectation–Maximization (EM) method to train the convolutional network (Papandreou et al., 2015) should also be mentioned here. These algorithm types were created some time ago and are currently coming back into vogue (Toldo et al., 2020). Note that detailed overviews of the weakly- and semi-supervised methods can be found in Zhang et al. (2020a) and Yu et al. (2018) papers.

Deep neural networks are usually used in the fully supervised learning methods. In the last decade, we have been able to observe a boom in research into neural networks and algorithms related to them. An essential aspect of such methods is a large, well-annotated dataset, which is not always available. However, deep learning methods may be far better than the classical ones (in generalized cases). In the next paragraph, we present a brief description of the most common neural network structures/architectures used for segmentation tasks.

Classically, neural networks can be divided into feed-forward and recurrent ones (Jozefowicz et al., 2015; Goodfellow et al., 2016). Currently, we should also add to this category networks based on convolution layers (Murphy, 2016; Sakib et al., 2019). In recent years, many different neural structures and blocks have been developed. These include residual blocks, inception blocks, Long Short-Term Memory cells (LSTM), Gated Recurrent Unit (GRU) cells, and classical Recurrent Neural Network (RNN) cells. In turn, the types of neural architectures used in segmentation include: Fully Convolutional Networks (FCN), Regional Convolutional Neural Networks (R-CNN), Encoder–Decoder architecture, Generative Adversarial Networks (GAN), Multi-scale Networks (MsNN), U-net, and many other architectures. According to Zhang et al. (2020b), the most popular backbones in the semantic segmentation task are: VGG16 (a network created by Visual Geometry Group) (Simonyan and Zisserman, 2014), ResNet (a residual neural network) (He et al., 2016), Inception (a neural network that consists of Inception modules) (Szegedy et al., 2016), MobileNet (a neural network designed for mobile and embedded vision applications) (Sandler et al., 2018), and Xception (a CNN with 71 layers) (Chollet, 2017).

3.1. Latest segmentation reviews

There are numerous reviews of different segmentation methods used for various purposes. A review of remote sensing (e.g. satellite imagery) using deep learning approaches can be found in Yuan et al. (2021). They present general neural architectures such as AlexNet, VGG, GoogLeNet, U-Net, and some approaches specific for the task, such as SegNet and DeepLab. They also show the test of twelve particular solutions with different datasets. The highest accuracy (98.45%) is achieved with the method presented in Sellami et al. (2019) on the Pavia University dataset (hyperspectral images over the city of Pavia in Italy).

The paper (Taghanaki et al., 2021) shows different contributions, such as: "architectural improvements, optimization function-based improvements, data synthesis-based improvements, weakly supervised models, sequenced models, and multi-task models", to the segmentation of the medical images from particular modalities such as EM (Electro-Magnetic), PET (Positron Emission Tomography), MRI (Magnetic Resonance Imaging), histology, fundus, CT (Computed Tomography), microscopy, US (Ultra Sound), X-ray, and using deep neural networks.

Minaee et al. (2021) shows segmentation methods using deep neural networks for images with large-scale objects, such as from the PASCAL VOC or MS COCO datasets. The authors processed more than 100 architectures, divided based on the type of neural networks. They distinguish several types of neural networks such as: fully convolutional (FCN),

Table 2
Neural architectures summary. *Pretrained dataset* shows the original dataset on which the architectures were trained. *Input type* denotes the type of the input of the neural network: RGB (color image), Disparity map, DDD (three channels of Depth), RGBD (color image with Depth channel), XYZ (Cartesian coordinates of the pixels). The *input resolution* is based on the default configurations suggested by the authors (some can be changed via a new configuration or network training). *Time* measured in milliseconds shows the average processing time on our test machine (for 80 different items). Note that, in the last column (*achievement*), the highest achieved metric is presented with a test dataset in parentheses. Note that mIoU is the mean Intersection over Union metric, AP stands for Average Precision, while mAP is mean AP, and $f1$ is an f1-score metric.

Architecture	*Pretrained dataset*	*Input type*	*Input resolution*	*Time* [ms]	*Achievement*
DeepLab	Cityscapes	RGB	513 × 513 × 3	156.111	mIoU = 85.7 (PASCAL VOC)
Mask RCNN	COCO	RGB	Any	105.685	AJI = mIoU = 48.6 (other)
DETR	COCO	RGB	800 × 800 × 3	169.610	AP = 31.1 (COCO)
GAIS Net	Cityscapes	RGB, Disparity	Any	498.887	mAP = 37.1 (Cityscapes)
Shapemask	COCO	RGB	1024 × 1024 × 3	549.075	AP = 33.3 (PASCAL VOC)
SD-Mask RCNN	Wisdom-real	DDD	512 × 512 × 3	785.577 (on CPU)	AP = 51.6 (WISDOM)
SF-Mask RCNN	Wisdom-real	RGBD	640 × 480 × 4	155.324	AP = 57.7 (other)
UOIS	TOD	RGB XYZ	640 × 480 × 6	248.936	$f1$ = 80.8 (OCID & OSD)
UOC	TOD	RGB XYZ	640 × 480 × 6	425.97	$f1$ = 88.9 (OCID & OSD)
YOLACT	COCO	RGB	Any	99.733	mAP_{50} = 72.3 (PASCAL SBD)

encoder–decoder (E–D), multiscale, feature pyramid (FPN), region-based convolutional (R-CNN), dilated convolutional, recurrent (RNN), attention-based, generative adversarial (GAN), convolutional with active contour, and other networks. It is worth mentioning that the best quality indicators on the PASCAL VOC dataset were achieved by such architectures as (Minaee et al., 2021): EfficientNet+NAS-FPN (Zoph et al., 2020): 90.5 mIoU (mean Intersection over Union), DeeplabV3+ with Xception-71 backbone (Chen et al., 2018a): 89.0 mIoU, and EMANet with ResNet152 backbone (Li et al., 2019): 88.2 mIoU.

The aspect of domain change/adaptation for semantic segmentation systems is described in Toldo et al. (2020). To put it simply, the domain adaptation task may be described as a change of the dataset after final training of the segmentation system. The most popular case of domain adaptation corresponds to changing the synthetic training set into the real environment. The highest metric (mean Intersection over Union — mIoU) has been achieved by a model based on the ResNet38 backbone for the conversion from the synthetic GTA5 (Grand Theft Auto) dataset to Cityscapes (Zou et al., 2019). In general, a broad review of the domain adaptation aspect in the visual context may also be found in Wang and Deng (2018).

The semi- and weakly supervised methods of semantic segmentation are described in Zhang et al. (2020a). The authors present a comparison of standard neural architectures such as CNN, R-CNN, FCN, and GAN used for semantic segmentation. According to them, the highest score (mIoU — 72.9) on the PASCAL VOC dataset was achieved with the method presented in Tang et al. (2018), based on ResNet. A comparison of similar methods can be found in Yu et al. (2018). Note that the authors also included there a review of some neural architectures. In addition, they provide the state-of-the-art on the PASCAL VOC dataset, where the highest mIoU was achieved by DeepLabv3 based on the ResNet backbone.

A broad overview of deep neural networks for semantic segmentation can be found in Lateef and Ruichek (2019) and Garcia-Garcia et al. (2017). Lateef and Ruichek (2019) present 18 known architectures along with their accuracy achieved on specific datasets. However, since the architectures were tested on different datasets, they cannot be directly compared. Garcia-Garcia et al. (2017) also presents tests on different datasets with different methods. Once again, the highest IoU was achieved by DeepLab on the PASCAL VOC dataset.

4. Tested neural architectures

Ten different neural architectures were selected during the literature review (in the context of instance segmentation of a heap of objects). Each of the architectures allows the use of weights pre-trained on a certain dataset. Some of the presented neural networks only use color images (RGB), some only use a depth map, and some use both of the above. It is not possible to accurately determine the computing power of each neural architecture due to their use of different core libraries and their versions (tensorflow/torch). However, the average processing time (for 80 different items) was measured on a machine based on an Intel i7-11700F (2.5 GHz) with 16 GB of RAM equipped with an NVIDIA GTX 3060 and 12 GB of graphic memory. The detailed characteristics of the presented neural architectures are presented in Table 2.

4.1. DeepLab v3

"DeepLab is a state-of-art deep learning model for semantic image segmentation, where the goal is to assign semantic labels (e.g., person, dog, cat, and so on) to every pixel in the input image" (Chen et al., 2022). In other words, DeepLab is a series of architectures which introduced innovations in segmentation using neural networks such as atrous convolution, combining it with CRF, atrous spatial pyramid pooling (ASPP), and combining ASPP with the existing backbones (such as MobileNet and Xception) (Chen et al., 2017, 2018b). Note that both backbone architectures use depthwise separable convolution.

The version tested in this work (v3+) is based on the encoder–decoder architecture. The encoder module consists of preprocessing operations, a MobileNet block in version 3 (trained on data from the Cityscapes dataset), and an atrous spatial pyramid pooling block, whereas the decoder part consists of convolution and upsampling layers. A sketch of the Deeplab architecture is presented in Fig. 3(a). It is worth mentioning that Deeplab was designed for outdoor image segmentation. The network was trained on the PASCAL VOC dataset. For this dataset, it achieved a mean IoU of about 85.7 (in version 3).

Note that the result of the DeepLab architecture is a fully segmented image — each pixel has an assigned label. In consequence, DeepLab can also segment a background and, e.g., designate walls and a table separately. This operation is very useful when the input image is cropped only to the working space. However, in our tests, we assume that the only preprocessing operation is scaling. Thus the metrics which describe this particular architecture may be understated.

4.2. Mask RCNN from Detectron2

Detectron2 is a model library created by the FAIR (Facebook AI Research) group (Wu et al., 2019). It includes many models for object detection, instance/panoptic segmentation, and human keypoints identification. Most of them are trained by the COCO dataset. Among them, we can find the Mask RCNN (Region-Based Convolutional Neural Network) R-50-FPN-3x architecture used in our test (Zhao et al., 2020). The architecture, shown in Fig. 3(b), uses ResNet50 (ResNet50 — a network with 50 residual blocks), with FPN (Feature Pyramid Network) as a backbone (trained on ImageNet — one of the biggest image datasets) for feature extraction. Features and proposed regions are aligned and processed by fully connected convolutional layers in order to obtain a result (bounding box, class probability, or mask). The architecture was compared to classical U-Net architectures in Sarker et al. (2021), and achieved 48.6 on the aggregated Jaccard Index (AJI) tested on the ICOS IHC (Inducible T-cell COStimulator immunohistochemistry) — a cell identification dataset.

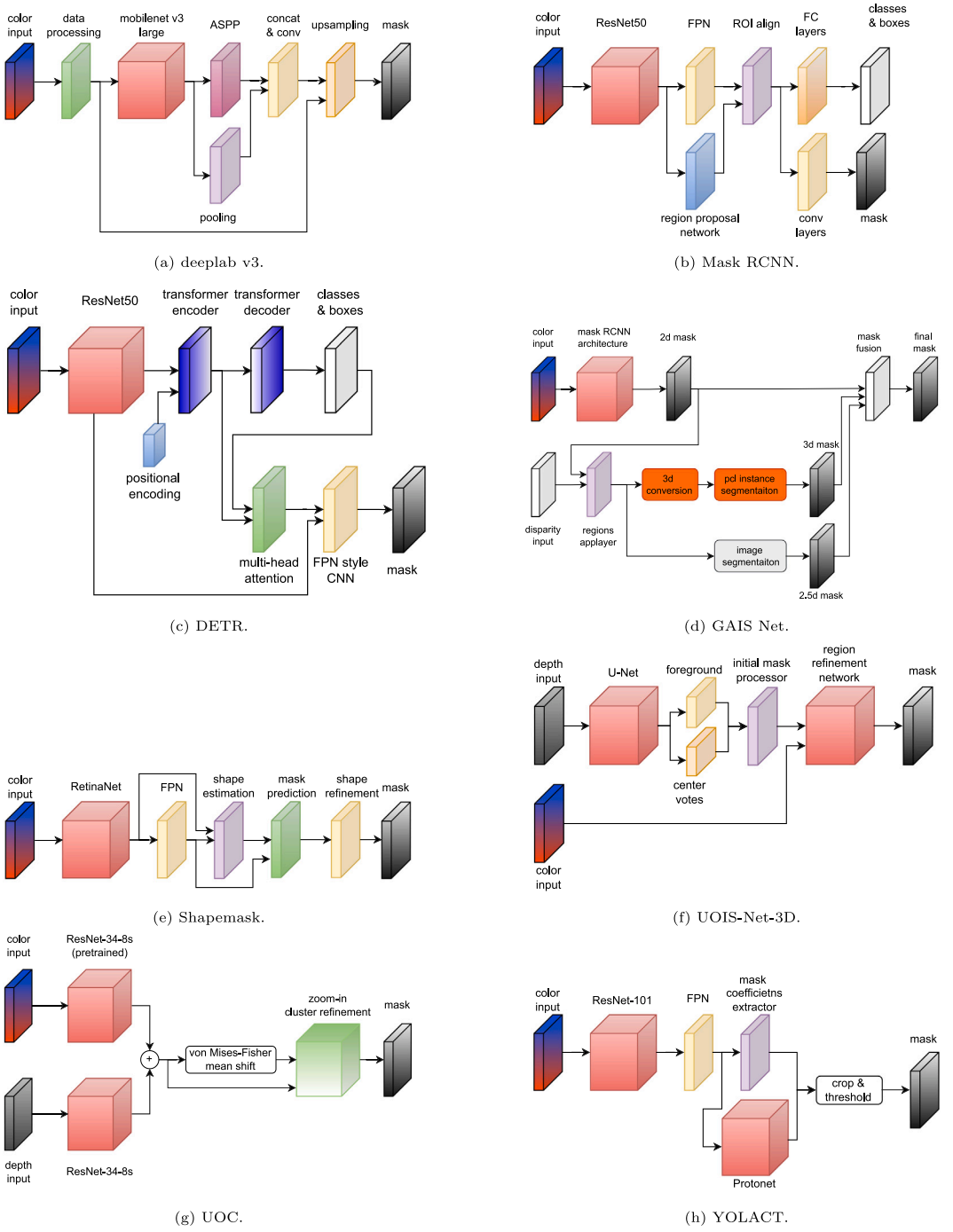


Fig. 3. Sketches of segmentation neural architectures. Note that the color input is always depicted as a blue–red gradient, while the depth input is white–gray, a backbone is shown as red–white, FPN or CNN layers are yellow, while other used algorithms are depicted as boxes.

4.3. DEtection TRansformer (DETR)

Another architecture available on the Facebook AI research repository is the DEtection TRansformer (Carion et al., 2020). The architecture also uses ResNet50 as a backbone for feature extraction. After that, the model uses the transformer architecture — a variation of an encoder–decoder with the use of multi-head attention and feed-forward layers (Vaswani et al., 2017). The output of the transformer can be converted to the prediction of bounding boxes. Adding the extension of "mask head on top of the decoder outputs" changes the purpose of the architecture to panoptic segmentation. The DETR was trained on a part of the COCO dataset (133 categories). It achieved an average precision (AP, described in detail in the next section) of 31.1. A sketch of the architecture can be found in Fig. 3(c).

4.4. Geometry-Aware Instance Segmentation (GAIS) net

GAIS Net is an architecture also based on ResNet50-FPN as a backbone (Wu et al., 2020). Part of the architecture's structure is very similar to Mask RCNN. Moreover, the code uses the part of the Detectron library mentioned earlier. The novelty of this architecture is to use disparity maps parallel to standard color images. The architecture was initially trained on the CityScapes dataset. It achieved a mean AP (mean average precision) at the level of 37.1, with training on COCO and CityScapes. A sketch of the architecture can be found in Fig. 3(d).

Note that GAIS Net uses a disparity map as an input. Since the datasets only provide depth maps, the usage of GAIS net is not advisable. Thus, we will omit this architecture in our tests.

4.5. Shapemask

Shapemask is another approach to the segmentation task, which is based on detection (Kuo et al., 2019). It uses RetinaNet (ResNet with feature pyramid) as a backbone; it can also use ResNet-101 with FPN to detect bounding boxes and feature extraction. Using the features and bounding boxes, the architecture estimates the shape of the objects and roughly predicts the masks. The shapes of the objects are refined by instance embedding. The authors trained the architecture on the COCO dataset divided into subsets: the objects available in the PASCAL VOC dataset and other objects. The architecture achieved, in particular, 30.2 (VOC trained, non-VOC tested) and 33.3 (non-VOC trained) of average precision. A brief sketch of Shapemask is shown in Fig. 3(e).

4.6. Synthetic Depth Mask R-CNN (SD-Mask)

The SD Mask R-CNN is an adaptation of the standard Mask R-CNN (described earlier and shown in Fig. 3(b)), with slight modifications (Danielczuk et al., 2019). Firstly, the authors changed the input of the architecture — from a color RGB image to a grayscale Depth image on all of the three channels. The authors of the architecture also changed the backbone to ResNet35 (35 residual layers). Note that the initial weights used in the backbone were pre-trained by the COCO dataset. The most important innovation here is that the authors used generated/simulated environments to train the architecture. They presented and used the WISDOM dataset (described earlier). The architecture was trained on simulation data. The tests were performed on the real part of the dataset. The architecture achieved about 51.6 of average precision.

4.7. Synthetic RGBD Fusion Mask R-CNN (SF-Mask)

The next step achieved by the researchers was to fuse the Depth image with RGB. The article (Back et al., 2020) shows such operations with different input modalities. They also used Mask R-CNN with ResNet50 as the main backbone architecture. The data fusion takes place before the FPN module. The color data, raw depth, and confidence map estimator are fused on four different convolution levels in the "ResNet50 backbone of each branch" (Back et al., 2020). As in the SD Mask R-CNN, the architecture was trained on synthetic data with the use of pretrained ImageNet weights (in ResNet50), and tested on real examples. The authors achieved 57.7 of average precision on their dataset collected by an Intel RealSense D415 RGBD camera.

4.8. Unseen Object Instance Segmentation (UOIS)

The UOIS net is an architecture which consists of three modules: Depth Seeding Network (DSN), Initial Mask Processor (IMP), and Region Refinement Network (RRN) (Xie et al., 2020). It is one of the few architectures which use a 3-channel organized point cloud as an input with a fusion of RGB images (in the RRN module). The DSN module can be described as an encoder–decoder architecture. The authors use a "U-Net architecture, where each of the 3×3 convolutional layers is followed by a GroupNorm layer and ReLU" to produce initial masks (background, table plane, table objects) and directions of object centers. Such an output is processed via the "Hough voting layer" to produce initial masks. The IMP module consists of image processing operations, mainly morphological, applied to the initial masks separately. After that, the masks are fused with a color image and processed by another U-Net architecture. The authors used the TOD dataset to train the architecture, and tested it on the OCID and OSD datasets. The architecture achieves f1-score metrics of about 80.8 on both datasets. The authors extend their architecture in Xie et al. (2021a). Namely, in the DSN module, they used dilated convolutions and changed the directions of the object centers from 3D offsets to object centers. The mean f1-score was higher by 3 points.

4.9. Unseen Object Clustering (UOC)

The UOC architecture is the next stage of depth and color processing architecture (Xiang et al., 2020). The architecture (presented in Fig. 3(g)) consists of two parallel backbones for color and depth processing. ResNet34 with 8 strides is used as both backbones. The color one is pretrained by ImageNet. After the backbone processing, the features are summed up and processed by two-stage clustering: the von Mises–Fisher mean shift (vMF-MS) and a zoom in cluster refinement. The second stage of clustering uses cropped RGB+D regions of interest, a neural network, and one more vMF-MS with thresholds. The algorithm is complex, but it achieves a mean f1-score at the level of 88.9 (for the OCID and OSD datasets).

4.10. YOLACT (You Only Look At CoefficienTs)

In general, most of the previously described architectures depend on a sequential process of four steps: feature extraction, bounding box localization, realigning features to the boxes, and mask prediction. The YOLACT architecture tries to remove the "explicit localization step" (Bolya et al., 2019). It consists of four steps: feature extraction (ResNet101 with FPN as a backbone), in parallel: mask prototype generalization via FPN and Protonet, calculation of mask coefficients and, at the end, mask assembly. The architecture was trained on the COCO dataset (29.8% of average precision) and the PASCAL SBD (72.3 of mAP_{50}). The main goal of the authors was to minimize the processing time, which was achieved — YOLACT works about three times faster than the standard Mask R-CNN (Bolya et al., 2019). The YOLACT architecture is shown in Fig. 3(h).

4.11. Summary

To sum up, a lot of architectures use only RGB images for segmentation purposes. Most of them use ResNet with Feature Pyramid Network as a backbone. Sometimes, they have other backbones such as Unet, MobileNet or RetinaNet. Newer architectures may also use Mask RCNN as a part of their structure. The UOC architecture seems to be the most valid architecture for our task. However, the various datasets and metrics used by the authors do not allow for direct comparison of the architectures.

5. Tests & their results

The first test was designed to evaluate all the networks with weights supplied from the source (the stock ones). Note that the stock training datasets were also included in the tests of several networks (such as the UOC, UOIS, SD-Mask RCNN, and SF-Mask RCNN). For example, the weights of the UOC network by default are trained on the TOD dataset. The results obtained on these datasets are much better than the other results. In this respect, the other networks are at a disadvantage. Our showcase performance is calculated on each of the datasets separately.

The second test was aimed at evaluating the four networks selected in the first test, retrained on all the RGBD stack datasets (additional learning). During the test, some additional remarks were made about the training, which was also described.

5.1. Initial conditions

All the tests were evaluated on the test part of the datasets, with a limitation of 100 samples (or less if the test part has fewer samples) from the dataset. The same machine as mentioned earlier was used. All the used metrics are described in the next subsection.

5.2. Metrics used in tests

For each framework/neural network ($f \in \mathcal{F}$) presented in this article, all the RGBD stack datasets ($D \in D$) were tested according to the following methodology. The items from the datasets were scaled and unified (e.g. XYZ axes in the required orientation), appropriately to the input of the selected neural network. Note that, for each item (one set of images) from the dataset ($m \in D$), there exists a ground truth matrix (GT). It usually has the form of a grayscale image, for simplicity converted into a matrix of binary masks $\mathbf{GT}_m \in \{0,1\}$, with the size of $w \times h \times N_{GT}$, where N_{GT} is the number of original instances, w is the width and h is the height of the image. The obtained results at the fth neural network output are also transformed into a similar matrix ($\mathbf{R}_m^f \in \{0,1\}$) with the appropriate resolution ($w \times h \times N_d$, where N_d represents the number of detected instances).

In general, the Jaccard index/IoU (Intersection over Union) for two binary masks: A, B of size $w \times h$ may be described as their binary conjunction divided by their binary sum:

$$IoU(A,B) = \sum_{i=1}^{w}\sum_{j=1}^{h} \frac{(A[i,j] \wedge B[i,j])}{(A[i,j] \vee B[i,j])}. \tag{1}$$

For two matrices: the original ground truth – $\mathbf{GT}_m$ from the dataset, and the result – $\mathbf{R}_m^f$, the Jaccard index matrix – $\mathbf{IoU}_m$ in the shape of $N_d \times N_{GT}$ may be calculated for each pair of the detected instance ($n_d \in \{1,\dots,N_d\}$) and the original instance ($n_{GT} \in \{1,\dots,N_{GT}\}$):

$$\mathbf{IoU}_m[n_d, n_{GT}] = IoU(\mathbf{R}_m^f[n_d], \mathbf{GT}_m[n_{GT}]) \tag{2}$$

Based on such a matrix, it is possible to determine the corresponding original instances. Note that the IoU for a certain instance (n_{GT}) evaluated for a network f is a maximum of the n_{GT}-th column from the matrix $\mathbf{IoU}_m$:

$$IoU^f(n_{GT}) = max_{i=1}^{N_d}(\mathbf{IoU}_m[i,n]). \tag{3}$$

Since one item contains N_{GT} different objects/instances, the metric for it may be based on an average of all the segmentations (for each instance), or for all the segmentations at once (pixel-wise). For a single instance with the matching ground truth/all the segmentations at once, we can describe the values of pixels as:

- true positive (TP) value is a sum of correctly detected pixels (belonging to the origin and detected instances)/correctly detected pixels
- false positive (FP) value is a sum of redundantly detected pixels (belonging only to the detected instance and not to the original one)/the pixels which are not present in the ground truth
- false negative (FN) value is the sum of undetected pixels (belonging only to the original instance)/the undetected pixels from objects
- true negative (TN) value is an irrelevant (0)/the background without any detections.

Note that, in some cases, the background can also be treated as a single object. However, there is no need to calculate another metric just for the background, since the metric can be calculated for the whole image – pixel-wise – IoU_{px}.

A single instance (object) may also be characterized by standard metrics calculated from a contingency matrix such as:

- $P = \frac{TP}{TP+FP}$ — precision
- $R = \frac{TP}{TP+FN}$ — recall
- $R_i = \frac{\sum TP}{FN+\sum TP}$ – recall calculated for one ground truth instance and all the detections – in short, we call it a coverage
- $F1 = \frac{FN}{FN+TP}$ — f1-score.

For all the segmentations on an image, we can calculate averages of the metrics mentioned earlier and metrics such as:

Table 3
Average results for used datasets from all tested networks. Note that the highest values are shown in bold.

Dataset	Metrics				
	$mIoU$	mP	mR	$mF1$	mAP
OCID	**0.3613**	0.4171	**0.4621**	**0.4029**	**0.2677**
STIOS	0.1922	0.2476	0.2950	0.2306	0.0943
TOD	0.3313	0.4084	0.3967	0.3669	0.2403
WISDOM	0.3219	**0.4358**	0.4143	0.3654	0.1937
OSD	0.3090	0.3742	0.4359	0.3537	0.2175
SBD-L	0.1538	0.1778	0.2320	0.1812	0.0766
SBD-Z	0.0671	0.0851	0.1161	0.0867	0.0124
SBD-S	0.2231	0.2717	0.2685	0.2550	0.1250

- $mIoU$ — average of $IoU^f(n)$ (only one instance for one ground truth) with negative detections
- $mIoU_{max}$ — average of $IoU^f(n)$ (only one instance for one ground truth and only positive detections)
- $mIoU_a$ — average IoU calculated for all detections
- IoU_{px} — pixel-wise IoU calculated for one item.

For one item, another two metrics – based on numbers with the standard threshold, not pixels – will also be calculated:

- TPR – True Positive Rate – number of positive detections divided by the number of objects
- PPV – Positive Predictive Value – number of positive detections divided by the number of all detections.

To present the architectures correctly, we also need to calculate averages over the datasets as a metric for network evaluation. For each threshold (th) of IoU from 0.5 to 0.95, with a step of 0.05, the confusion matrix can be calculated for all/selected datasets. The mAP_{th} metrics at a fixed threshold can be calculated as follows:

$$mAP = \frac{1}{10}\sum_{th=0.5}^{0.95} AP_{th} \tag{4}$$

where AP_{th} describes an average precision calculated from the confusion matrix at a fixed threshold (Everingham and Winn, 2011):

$$AP_{th} = \frac{1}{11}\sum_{r\in\{0,0.1,\dots,1\}} P_{interp}(R) \tag{5}$$

$$P_{interp}(R) = max_{\hat{R}:\hat{R}\geq R} P(\hat{R}) \tag{6}$$

where $P(\hat{R})$ is the measured precision at recall $\hat{R}$.

5.3. Stock weights

Using the weights provided by the authors of the presented neural networks (stock weights), all the networks (except GAIS Net — since it takes disparity as an input) were evaluated separately on all the eight previously presented heap datasets (Section 2.3). The average metrics for each dataset was calculated. As was mentioned earlier, all the items were scaled to fit the input of the networks. Note that, in some cases, this operation may be advantageous (Mask RCNN, GAIS Net, YOLACT), especially for the datasets that contain large RGB images (e.g. STIOS). Sample results are presented in the supplementary materials.

5.3.1. Datasets

For each dataset, the maximum 100 samples (or fewer, depending on the items in the dataset) were evaluated from the test subset (Table 1). A summary of the average performance of all the networks tested on the given datasets is presented in Table 3.

The OCID dataset seems to be the most *suitable* one for the networks. As can be seen in Fig. 2, the OCID dataset contains the most separated objects (a small number of occlusions), seen from a certain perspective.

Table 4
Details of neural networks inference results on given datasets.

Datasets	Mask RCNN	YOLACT	DETR	SF-Mask	Shapemask	DeepLab	UOC	SD-Mask	UOIS
mIoU									
OCID	0.2187	0.1079	0.2432	0.3951	0.4243	0.0352	**0.7054**	0.5381	0.6175
STIOS	0.3131	0.2278	0.2304	0.2487	**0.4120**	0.0438	0.0358	0.0716	0.1543
TOD	0.1477	0.0988	0.0792	0.3728	0.2007	0.0651	0.7287	0.4276	**0.7329**
WISDOM	0.2552	0.1684	0.0378	0.6052	0.3839	0.0599	**0.6602**	0.5075	0.0146
OSD	0.2565	0.1798	0.2006	0.2851	0.2906	0.0434	**0.6586**	0.3852	0.6288
SBD-L	0.0017	0.0022	0.0047	0.3412	0.0013	0.0128	0.4039	**0.5139**	0.0111
SBD-Z	0.0021	0.0001	0.0008	**0.2922**	0.0032	0.0182	0.0793	0.0772	0.0575
SBD-S	0.0536	0.0220	0.0119	0.4737	0.0634	0.0165	0.5659	**0.5789**	0.0469
mP									
OCID	0.2368	0.1147	0.2594	0.5076	0.4665	0.1060	**0.7845**	0.5791	0.6666
STIOS	0.3279	0.2354	0.2343	0.3525	**0.4284**	0.1243	0.0364	0.1100	0.2371
TOD	0.1571	0.1032	0.0887	0.4821	0.2159	0.1650	0.8552	0.5049	**0.8998**
WISDOM	0.2777	0.1801	0.0383	**0.8784**	0.4117	0.1903	0.8491	0.6983	0.0229
OSD	0.2778	0.1883	0.2309	0.3690	0.3271	0.1159	**0.7859**	0.4618	0.7352
SBD-L	0.0017	0.0023	0.0050	0.3751	0.0013	0.0312	0.5021	**0.5630**	0.0118
SBD-Z	0.0021	0.0001	0.0008	**0.3492**	0.0033	0.0454	0.0921	0.1086	0.0775
SBD-S	0.0563	0.0226	0.0123	0.5963	0.0676	0.0469	0.6462	**0.7122**	0.0697
mR									
OCID	0.2818	0.2036	0.3142	0.5210	0.5447	0.1493	**0.7745**	0.6368	0.7327
STIOS	0.4389	0.3537	0.3338	0.3809	**0.5348**	0.2131	0.0443	0.1416	0.1871
TOD	0.1642	0.1280	0.1437	0.4586	0.2650	0.2274	0.7562	0.4874	**0.7637**
WISDOM	0.3619	0.2810	0.1719	0.6561	0.5884	0.2049	**0.7142**	0.5350	0.0153
OSD	0.3674	0.2963	0.3379	0.3779	0.5652	0.2076	0.7181	0.4520	**0.7201**
SBD-L	0.0447	0.0687	0.0219	0.5168	0.0342	0.1137	0.5243	**0.6134**	0.0187
SBD-Z	0.0037	0.0101	0.0016	**0.4539**	0.0047	0.1106	0.1225	0.1473	0.0764
SBD-S	0.0611	0.0268	0.0187	0.5881	0.0800	0.0879	**0.6595**	0.6286	0.0546
*mF*1									
OCID	0.2359	0.1191	0.2642	0.4692	0.4603	0.0568	**0.7593**	0.5867	0.6786
STIOS	0.3434	0.2517	0.2569	0.3188	**0.4425**	0.0738	0.0390	0.0999	0.1874
TOD	0.1557	0.1049	0.0890	0.4323	0.2144	0.1032	0.7734	0.4721	**0.7870**
WISDOM	0.2744	0.1820	0.0472	0.7025	0.4122	0.0957	**0.7308**	0.5727	0.0174
OSD	0.2791	0.1964	0.2216	0.3477	0.3266	0.0738	**0.7211**	0.4434	0.6967
SBD-L	0.0024	0.0034	0.0061	0.4212	0.0018	0.0218	0.4710	**0.5764**	0.0141
SBD-Z	0.0026	0.0003	0.0011	**0.3735**	0.0035	0.0301	0.0956	0.1083	0.0716
SBD-S	0.0579	0.0240	0.0132	0.5629	0.0695	0.0261	0.6318	**0.6481**	0.0569
mAP									
OCID	0.1996	0.0772	0.2513	0.2739	0.3665	0.0016	**0.7512**	0.5459	0.6947
STIOS	0.2934	0.2016	0.2501	0.1198	**0.3409**	0.0000	0.0300	0.0309	0.1430
TOD	0.1367	0.0866	0.0642	0.2318	0.1198	0.0080	0.7392	0.4422	**0.7748**
WISDOM	0.2414	0.1557	0.0254	0.5268	0.2477	0.0035	**0.7148**	0.5246	0.0030
OSD	0.2142	0.1194	0.1576	0.1067	0.1931	0.0000	**0.7546**	0.4676	0.6961
SBD-L	0.0000	0.0004	0.0039	0.2928	0.0006	0.0000	0.4292	**0.6050**	0.0065
SBD-Z	0.0027	0.0000	0.0000	**0.2076**	0.0029	0.0007	0.0612	0.0232	0.0415
SBD-S	0.0573	0.0240	0.0113	0.4305	0.0614	0.0005	0.6082	**0.6400**	0.0362

The WISDOM dataset, which has the highest precision, is placed a little behind the front. Probably the main reason for such high metrics is the centralized location of objects and the dark background.

Note that our datasets proposed in this paper – SBD-L and SBD-Z – were the most difficult ones to achieve good network performance. Naturally, the primary reason for such a situation is the lack of training. Another reason for this may be the natural noise caused by the unclean production environment (where the datasets were created). However, SDB-Z is based on the same view perspective and the same camera as WISDOM. It would seem that a similar environmental setup would bring comparable results. The main difference between those datasets lies in the type of objects. The SDB contains quite simple objects in one color, while different shapes and colors are used in the WISDOM dataset. Note that stack objects with similar textures, or even those cast from a single material, would be the main goal for a bin-picking robot. Such an assumption was made during the preparation of the SDB datasets, even though the object seems to be simple. Thus the SDB datasets may be quite important in our tests.

All the metrics split on each of the datasets are presented in Table 4. As can be observed, four architectures, namely SF-mask, UOC, SD-mask, and UOIS (all using a depth channel), are ahead of the rest. Among them, the SF-mask architecture presents the most evenly distributed metrics across all the datasets. On the other hand, the UOIS network is clearly well trained only for the TOD, OCID, and OSD datasets. Looking at the end of the list, the DeepLab architecture presents quite low metrics. The assumption of the architecture, namely that each pixel should be assigned to a certain class (panoptic segmentation) may be the main reason for this.

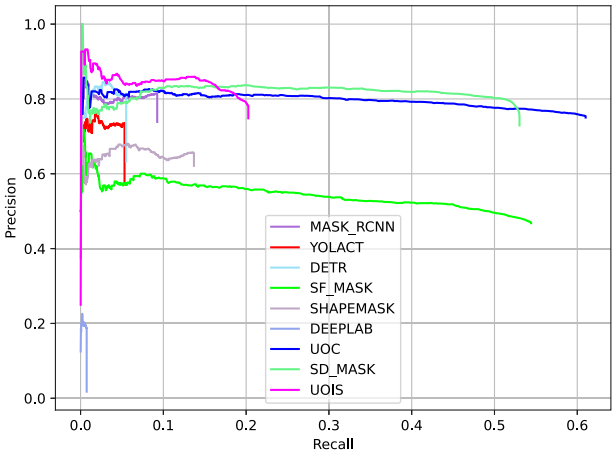
5.3.2. Networks

The mAP chart of the tested architectures is presented in Fig. 4(a). The four previously mentioned architectures (UOIS, UOC, SD-mask, SF-mask) seem to have the most valuable characteristics. The averaged metrics of the evaluated networks, described in Section 5.2, are presented in Table 5. Remember that the most general metric — $mIoU_a$ is calculated for all the instances, $mIoU$ shows the average for one matching detection — the other matches are ignored, while $mIoU_{max}$ ignores all the negative detections. $mIoU_{px}$ is a metric that includes the background.

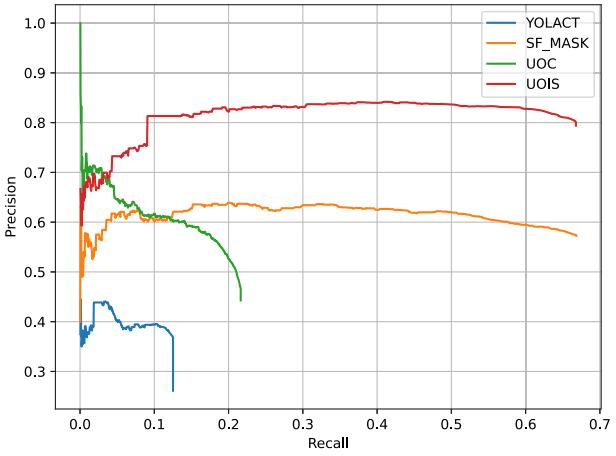
Since $mIoU_a$ takes into account all the detections, it is the lowest metric. However, the UOC architecture value is about 0.4607, which means that the detections are rather positive. Note that all the UOC metrics are rather high.

Table 5
Average metrics for evaluated neural networks without division into certain datasets.

Networks	MASK-RCNN	YOLACT	DETR	SF-MASK	SHAPEMASK	DEEPLAB	UOC	SD-MASK	UOIS
$mIoU$	0.6978	0.5387	0.5672	0.5388	0.6603	0.0973	0.6815	0.6696	**0.7017**
$mIoU_a$	0.0847	0.0481	0.0479	0.3871	0.1343	0.0288	**0.4607**	0.4068	0.1779
$mIoU_{max}$	0.7389	0.5849	0.5766	0.6942	0.7521	0.1202	**0.7774**	0.7301	0.7614
$mIoU_{px}$	0.1613	0.1125	0.1117	0.5397	0.1617	0.1160	**0.6412**	0.5465	0.3785
mP	0.1464	0.0867	0.0793	0.4184	0.1193	0.0447	0.8460	0.4564	**0.8612**
mR	0.3002	0.2538	0.1840	0.4743	0.4313	0.2466	0.2660	**0.5228**	0.3923
$mF1$	0.1968	0.1292	0.1109	0.4446	0.1869	0.0756	0.4048	0.4873	**0.5390**
mR_i	0.1224	0.0884	0.0849	**0.7670**	0.2126	0.3905	0.6553	0.5790	0.2385
TPR	0.0911	0.0516	0.0536	0.5676	0.1351	0.0063	**0.5703**	0.5187	0.2009
PPV	0.0896	0.0511	0.0534	0.3362	0.1264	0.0052	**0.5071**	0.4430	0.1897



(a) mAP before training.



(b) mAP after additional training.

Fig. 4. mAP chart for the tested neural networks.

The UOIS architecture also seems to be doing well with the average metrics. However, it does not cope with the SBD datasets.

Despite the fact that, on average, the SF-mask is not the best architecture, it constitutes a valuable contribution, since its average coverage (mR_i) is the highest. It means that there can be multiple detections of one instance, but they cover the ground truth pretty well.

5.3.3. The aspect of complementarity

During the tests of a number of neural networks, with regard to segmentation on many different datasets, it is worth considering their complementarity — i.e. which network is the most complementary to another one. Assuming that two segmentation networks are used in parallel or that a boosting algorithm is used (Baig et al., 2017), the complementary metric should provide the information about which network will probably work well with another one. Note that a highly complementary network should segment the object which the other network is having problems with.

Let us consider a complementary metric for network A with another network B defined as:

$$cpl_{AB} = \frac{|\{x : (\forall_{n\in D})(IoU^A(n) < 0.5 \wedge IoU^B(n) > 0.5)\}|}{|\{x : \forall_{m\in D} IoU^A(n) < 0.5\}|} \tag{7}$$

where the $|\{\}|$ operation is a cardinality of the set, while $IoU^A(n)$ is defined in (3).

The complementary relations for all tested networks are presented in Table 6. In general, the best network for complementing the others is the SF-mask, while for the SF-mask network, the best network to combine it with is the UOC with a score of above 0.47.

Note that the computed relation is not symmetrical. While the relation complementarity between the UOC and the SF-mask is 0.58, the relation between the SF-mask and the UOC is 0.47. As a result, when the difference between two complementarity metrics is relatively big, one network could be used as a kind of validator for the other. When both complementarity scores are high, the segmented objects are in a common pool. In the situation of low complementarity, this aspect may be omitted. In such a case, there is a possibility of changing the initial configuration of the networks — e.g. by introducing additional parameters such as the size of the objects.

5.3.4. Size of objects

The object size impacts the metrics, therefore this report searches for a relationship which helps to determine how well networks deal with objects of different sizes. Since the shapes of the images are different in various datasets, the object size is measured in relation to the image area. The size of a single object is presented as the total number of pixels that are occupied by the corresponding label mask, or the detection divided by the image resolution as a percentage.

The graphs of the relation between the number of objects and their size are shown in Fig. 5. Note that the bold black dashed line shows the ground truth — all the dataset statistics. For positive detections (Fig. 5(a)), UOC, SD-mask, and SF-mask show (shape) characteristics similar to the ground truth, especially for smaller objects. However, the number of objects correctly detected by the networks leaves much to be desired. Note that in the highest peak (~0.135), the best network, SD-mask, correctly detects about 70% of objects.

Note that the characteristics for positive detections of UOC, SD-mask, and SF-mask have similar shapes, but the object size is below the average. This means that the networks detect the object shape quite correctly, but some objects are not detected.

On the other hand, Fig. 5(b) shows that small objects are responsible for the majority of false detections in the case of the UOIS network. The characteristics of Shapemask, YOLACT, and Mask-RCNN are quite similar. All those networks falsely detect objects of similar sizes.

Table 6
Complementarity of tested neural networks according to (7). Each row shows the complementarity of all networks for the network shown in the first column.

Networks	MASK RCNN	YOLACT	DETR	SF-mask	Shapemask	DeepLab	UOC	SD-MASK	UOIS
MASK RCNN	X	0.0433	0.0422	**0.4504**	0.1224	0.0061	0.4196	0.3762	0.1500
YOLACT	0.0821	X	0.0459	**0.5004**	0.1352	0.0057	0.4661	0.4178	0.1659
DETR	0.0741	0.0427	X	**0.4484**	0.1216	0.0057	0.4177	0.3745	0.1490
SF-mask	0.1689	0.1412	0.1402	X	0.2109	0.1084	**0.4727**	0.4345	0.2352
Shapemask	0.0770	0.0455	0.0444	**0.4521**	X	0.0084	0.4214	0.3780	0.1521
DeepLab	0.1026	0.0698	0.0687	**0.4936**	0.1522	X	0.4615	0.4164	0.1809
UOC	0.1267	0.0885	0.0872	**0.5823**	0.1845	0.0435	X	0.4923	0.2179
SD-MASK	0.1226	0.0882	0.0870	**0.5337**	0.1748	0.0475	0.5000	X	0.2050
UOIS	0.0975	0.0627	0.0615	**0.5121**	0.1501	0.0217	0.4782	0.4303	X

Table 7
$F1-score$ for each dataset and each type of training using SF-Mask architecture after 28 epochs. The data for the evaluation was from the test parts of the datasets.

	$F1$ of datasets								
	OCID	STIOS	TOD	WISDOM	OSD	SBD-L	SBD-Z	SBD-S	$mF1$
528 items balanced	0.6954	0.5641	0.5766	0.6655	0.6697	0.7438	0.6715	0.6428	0.6545
528 items random	0.6414	0.4902	0.7298	0.6673	0.3098	0.4827	0.4275	0.6043	0.5526
528 items sequential	0.6887	0.5175	0.5278	0.7188	0.7198	0.7392	0.6763	0.6432	0.6486
1000 items proportional	0.628	0.4843	**0.7423**	0.6720	0.3795	0.5285	0.4957	0.6259	0.5809
1000 items filled	**0.7419**	**0.6495**	0.6424	**0.7988**	**0.7507**	**0.8451**	**0.8338**	**0.6914**	**0.7444**

5.4. Training procedure

In general, the number of training samples depends on the dataset and the hyper-parameters (batch size). The standard training procedure uses random data from a single dataset. In our case (using eight different datasets, each with a different number of objects), there is a possibility of *over*-fitting to the more numerous datasets. To optimize the final tuning, a pre-training test was performed. Four different types of cropped datasets were proposed: proportional (1000 samples), filled (1000 samples), balanced (528 samples), and sequential (528 samples).

In the proportional approach, the samples are randomly taken from each dataset for training, according to their maximum number. The more samples the dataset contains, the more are selected for the cropped dataset. Thus, the TOD dataset dominates in this approach. The samples randomly feed the SF-mask architecture in training.

On the other hand, the filled approach takes a maximum number of samples from the less numerous datasets (66 from OSD, 115 from STIOS, and 136 samples from the remaining datasets). The prepared mixed dataset is still unbalanced. However, the numbers are quite similar.

The sequential and balanced approaches take exactly 66 samples from each dataset.

The sequential approach relies on the dataset sequence. During one epoch of the training, the samples are fed from one dataset until they are exhausted. Afterwards, the dataset is switched to the next one. It may be presented as: [TOD, ..., TOD, OSD, ..., OSD, STIOS, ..., STIOS, etc.] In contrast, the balanced approach feeds the training with random samples from each of the datasets. An example of such training may be presented as [TOD, OSD, STIOS, ..., TOD, OSD, STIOS, ..., etc.]

To show the difference in the pre-training procedure, the results from SF-mask training are presented in Table 7. The training was limited to 28 epochs with the loss functions and other hyper-parameters of the SF-mask architecture (Back et al., 2020), originally proposed by the authors. Note that after pre-training, the selected network was evaluated on all the datasets (test parts). During the training, the average loss function decreased.

As can be supposed, the random and proportional approaches, which reward the TOD dataset (the most numerous one), have the lowest average score. The balanced and sequential approaches (an equal number of samples) are not much different. Surprisingly, even when the unbalanced approach is used, the F1 metric grows, even for the least numerous datasets (STIOS and OSD). Thus the main conclusion from the test is to use more samples than in the smallest dataset, but with some kind of upper limit. One dataset should not be allowed to dominate the others.

Table 8
Hyper-parameters of training. Note that 'U' marks the parameter as tuned by Optuna, while 'D' is a default parameter used by the authors of the architecture.

Architecture	Yolact	UOC	UOIS	SF-mask
Learning rate	0.00010 U	0.00001 D	0.00010 D	0.00010 D
Gamma		0.1 D		0.1 U
Weight decay	0.0005 U	0.0005 D		0.0001 D
Momentum	0.9 U	0.9 D		0.9 D
Delta			0.1 D	
Sigma			0.2 D	

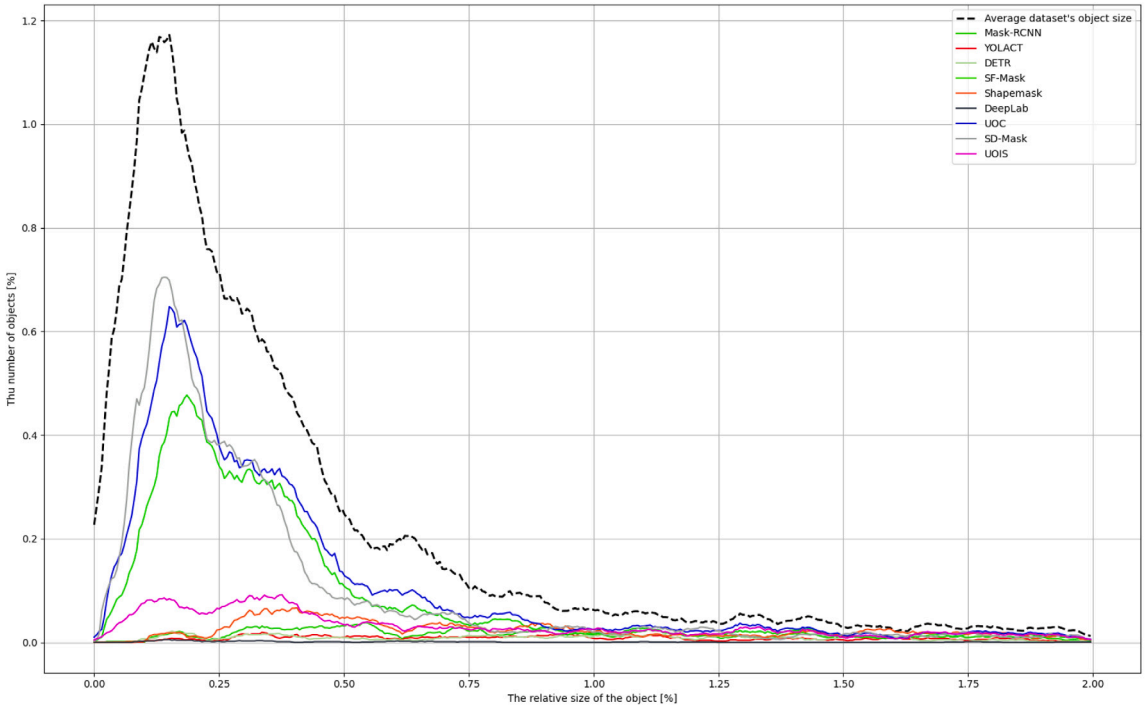
5.5. Additional training

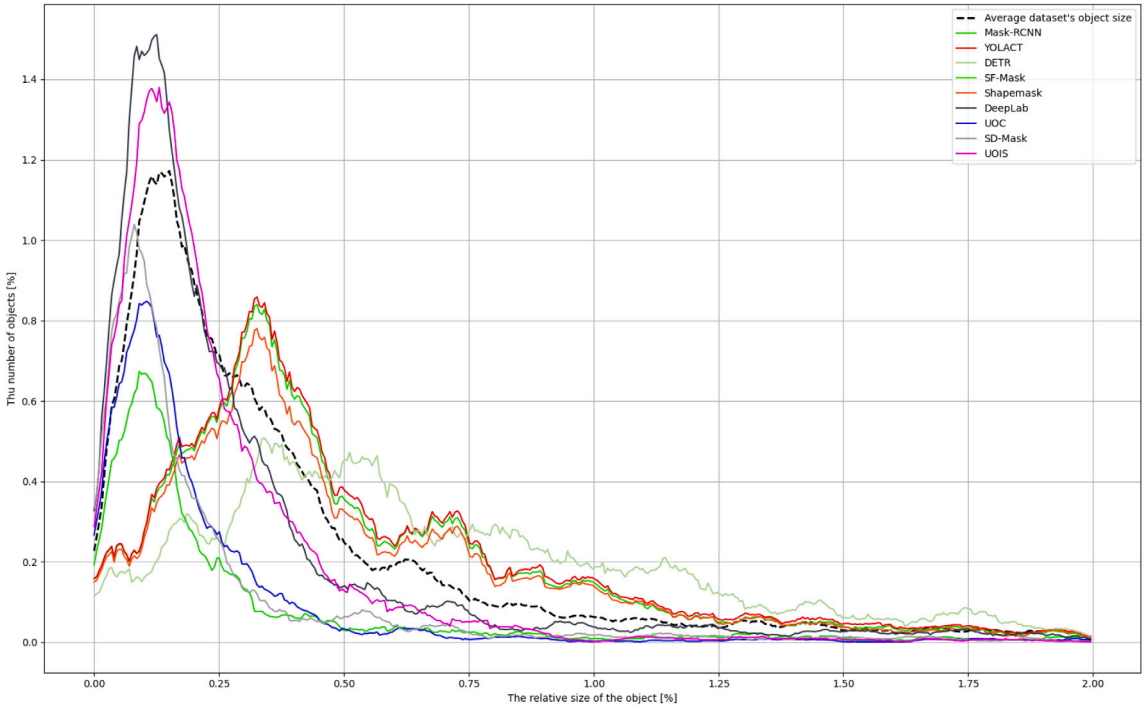
Based on the results presented above, the YOLACT, SF-MASK, UOC, and UOIS networks were chosen for further training. The main reason for this selection was the achieved levels of the evaluated metrics. However, in one case (YOLACT), we made an exception. YOLACT processes only the RGB input — it may be valuable to test it for further architecture development.

In the training procedure, the number of parameter classes for all the networks was set to one (sometimes two, if the background must be included). Thus all the classes are treated as one type of object. This simplification is necessary, due to the difference in datasets and network configurations. Since the best results in pre-training were achieved when using the filled 1000 dataset method, as described in the previous section, it was also used in the final network training.

Hyper-parameters such as the learning rate, momentum, decay, and optimizer were inspired by the suggestions made by the authors with regard to a given network, but slightly adjusted, for better results, by the Optuna framework (Akiba et al., 2019). They are presented in Table 8.

The evaluation of networks after additional training is presented in a similar form as before. The additional training was also limited to 28 epochs, like the test with the different training datasets. The mAP curve is shown in Fig. 4(b), the metric average values are presented in Table 9, while the detailed metrics split by the dataset are presented in Table 10. Note that we also computed complementarity metrics for the trained networks (Table 11).





(b) False detections ($mIoU < 0.15$).

Fig. 5. Approximation of the relation between the number of object instances to their size. The X axis shows the relative size of the object in % according to the image resolution. It is divided into 400 intervals. While on the Y axis, the number of detected objects (in a certain interval) divided into all objects is shown. The dashed black line shows the ground truth, and its integral is equal to 100%.

In general, after additional training, the UOIS architecture achieved better results than with the original weights (Table 4), especially on the SBD datasets. The decreased precision and the $mIoU_{max}$ may lead to the conclusion that more objects can be detected, but less precisely. Before the training, the architecture was basically reliable only on the TOD, OCID and OSD primary datasets, while the metrics for the other sets were near zero. After the training, the mAP scores for the primary datasets changed: TOD 0.77 → 0.67 (decreased), OCID 0.72 → 0.81, and OSD 0.69 → 0.81. At the same time, major improvement is noticeable in the other datasets, especially with regard to WISDOM 0.003 → 0.77. Unfortunately, the results for the SBD-Z and STIOS datasets are still not satisfying: mAP < 0.35, to use the network consistently on them. The camera parameters are probably somehow influencing the network results, since both datasets were created with a ZED camera.

Table 9
Average metrics after additional training for all datasets.

Networks	YOLACT	SF-Mask	UOC	UOIS
$mIoU$	0.2690	0.6088	0.5862	**0.7095**
$mIoU_a$	0.0952	0.4742	**0.5127**	0.5086
$mIoU_{max}$	0.3604	0.7452	**0.7533**	0.7518
$mIoU_{px}$	0.1210	0.6917	0.7612	**0.8041**
mP	0.0647	0.6308	**0.8740**	0.8238
mR	0.3649	0.5747	0.6006	**0.6631**
$mF1$	0.1099	0.6014	0.7120	**0.7348**
mR_i	0.2982	0.7705	**0.8358**	0.6998
TPR	0.1050	0.6567	**0.7195**	0.6118
PPV	0.0925	0.4458	0.5450	**0.5691**

Note that the UOIS architecture is described as "Unseen Object Instance Segmentation". The architecture should detect an unknown object without additional training. However, the metrics do not indicate that this is the case.

The average metrics of the UOC architecture show that it has learned quite well. The architecture has particularly improved its recall and F1-score (Table 9). Note that the detailed metrics show great improvement on the SBD datasets. However, the network still does not learn well on the STIOS dataset (an improvement on the mAP form 0.03 → 0.14), while its improvement on the SBD-Z is quite significant (0.06 → 0.42). Such behavior may be caused by the view perspective of the images in the STIOS dataset. However, a similar perspective is present in the OSD and TOD datasets.

The SF-mask architecture achieved significant improvements across all the datasets, excluding the WISDOM dataset (Table 10). Similarly to UOIS, the architecture tries to adapt to all the datasets at the expense of the primary learning dataset. The SF-mask architecture achieved the highest metrics on the STIOS and SBD-Z datasets. Thus it can be used as a complementary architecture to UOIS. This fact is proven by the complementarity metrics presented in Table 11.

The YOLACT architecture did not achieve any significant improvements. The training slightly corrected the metrics on the WISDOM dataset. Since the YOLACT operates only on RGB images, we assumed that the architecture would improve metrics on the SBD datasets. Unfortunately, this did not happen.

The mAP curves also show improvement of the architectures. In this respect, the UOIS seems to be the best architecture after additional training. However, the statistics on the SBD-Z and STIOS datasets do not indicate this. It would be better to use the UOC architecture to cope with different situations.

Looking at Table 11, it seems that the best combination for our purpose is to use the SF-Mask with the UOC architecture.

The approximation of the relation to object size may be found in Fig. 6. The dotted lines and the solid lines on the chart show, respectively, the results without and after training. According to Fig. 6(a), all the architectures improved themselves.

5.6. Formal comparison

Based on the data presented in Table 9, it can be concluded that the UOIS architecture shows the highest efficiency. Looking more closely at the data from individual datasets, we might assume that this phenomenon depends on its advantage in the TOD, WISDOM, and OSD datasets. On the other hand, for the datasets presented by us, especially those based on depth cameras (SDB-l, SDB-Z), the UOC architecture would be the best choice. However, we chose the SF-Mask architecture for further work, due to the method of implementation and the ease of replacing backbones. Taking into account the data on complementarity, the combination of SF-Mask with UOC should also be used for further consideration.

Further, we tested the SF-Mask architecture with different backbones. The default backbone of this architecture is ResNet50. A description of ResNet50 can be found in Shabbir et al. (2021). The literature review allowed us to select the following (potentially better) backbones: ResNet152 (more residual layers), RegNet32, and EfficientNet-B4 (potentially better) (Xu et al., 2022a; Tan and Le, 2019). The results of inference on the test set are presented in Table 12. Note that, in this particular case, the training time was not limited to 28 epochs and we used a confidence threshold on the output of the architecture. Thus, all detections with confidence below 0.5 were rejected (marked as false negatives). This effect may be observed in the mR metric.

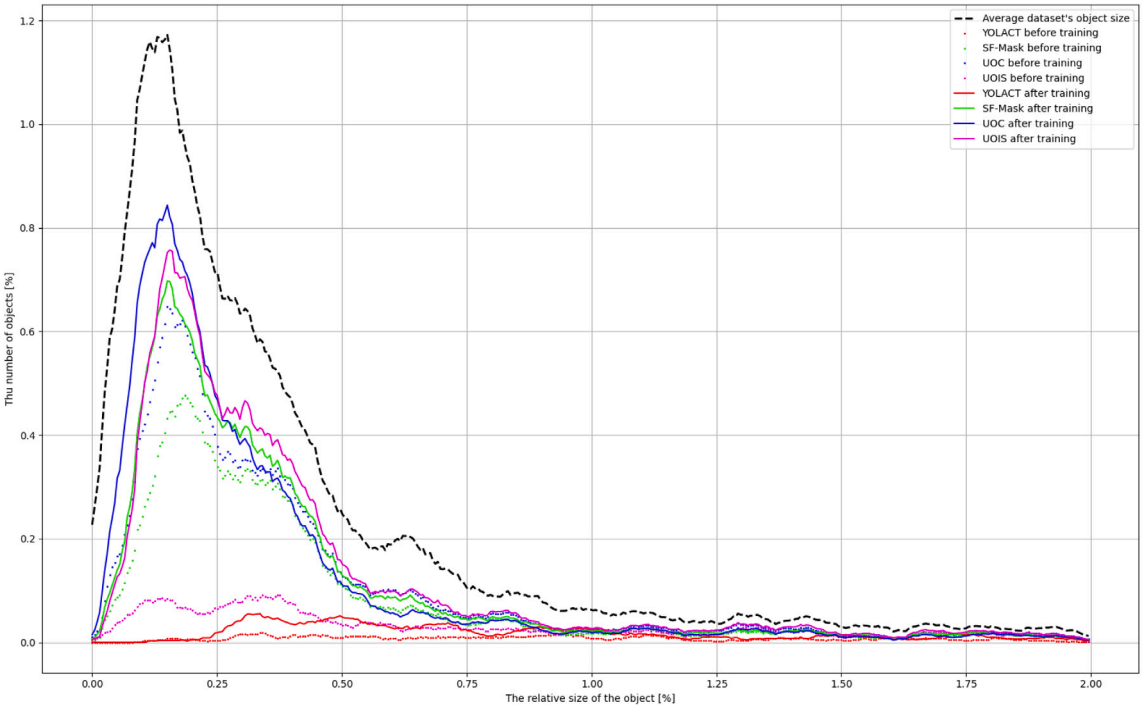
This particular test showed that similar results can be obtained by ResNet50 and RegNet32. However, the network which should give the highest results (EfficientNet-B4) did not perform as we expected.

Table 10
Network results on datasets after training.

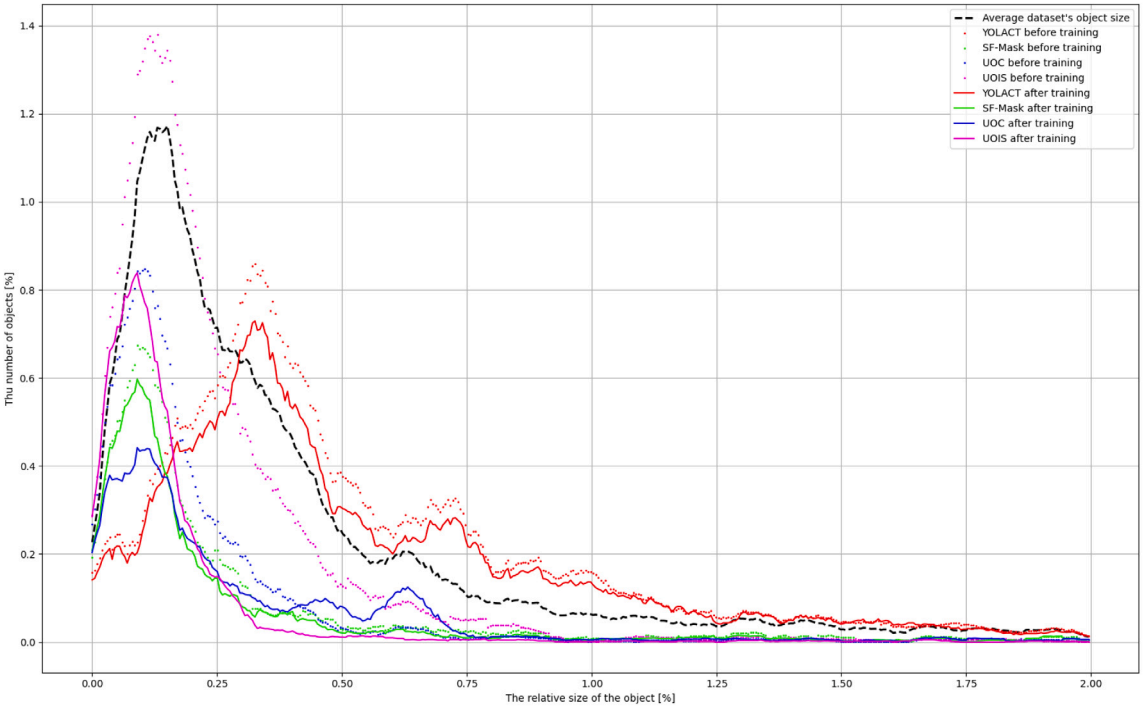
Datasets	YOLACT	SF-Mask	UOC	UOIS
$mIoU$				
OCID	0.1295	0.5187	0.5885	**0.7039**
STIOS	0.1013	**0.3768**	0.2631	0.2625
TOD	0.1003	0.6237	0.5725	**0.6487**
WISDOM	0.3273	0.5495	0.5945	**0.6907**
OSD	0.1518	0.2544	0.6662	**0.7101**
SBD-L	0.0397	0.3868	**0.5724**	0.5191
SBD-Z	0.0082	0.3558	**0.4239**	0.3195
SBD-S	0.0991	**0.5605**	0.5180	0.5244
mP				
OCID	0.1858	0.6794	0.7551	**0.7665**
STIOS	0.2001	**0.5488**	0.5488	0.4249
TOD	0.1402	0.7662	0.7015	**0.8893**
WISDOM	0.4748	0.8500	**0.8792**	0.8179
OSD	0.2990	0.3445	**0.8609**	0.8578
SBD-L	0.0490	0.4817	**0.7447**	0.5488
SBD-Z	0.0113	0.4592	**0.5499**	0.3545
SBD-S	0.1178	0.6678	**0.7331**	0.5548
mR				
OCID	0.3263	0.6381	0.6459	**0.8105**
STIOS	0.3396	**0.4660**	0.3069	0.3139
TOD	0.2573	0.6648	0.6016	**0.6809**
WISDOM	0.4915	0.6059	0.6360	**0.7835**
OSD	0.3362	0.3167	0.7085	**0.7681**
SBD-L	0.1913	0.5112	0.6579	**0.6846**
SBD-Z	0.0964	0.4714	**0.5207**	0.4536
SBD-S	0.2097	**0.6528**	0.5851	0.6464
$mF1$				
OCID	0.1720	0.6088	0.6595	**0.7707**
STIOS	0.1491	**0.4589**	0.3283	0.3188
TOD	0.1361	0.6797	0.6226	**0.7166**
WISDOM	0.4022	0.6452	0.6755	**0.7710**
OSD	0.2188	0.3049	0.7420	**0.7831**
SBD-L	0.0482	0.4747	**0.6601**	0.5988
SBD-Z	0.0126	0.4375	**0.5033**	0.3840
SBD-S	0.1173	**0.6400**	0.5963	0.5834
mAP				
OCID	0.0689	0.4761	0.6179	**0.8128**
STIOS	0.0426	**0.2580**	0.1411	0.2302
TOD	0.0422	0.5785	0.5735	**0.6774**
WISDOM	0.3207	0.5357	0.6436	**0.7674**
OSD	0.0590	0.1388	0.7644	**0.8174**
SBD-L	0.0169	0.3407	**0.6493**	0.6481
SBD-Z	0.0011	0.2778	**0.4178**	0.3306
SBD-S	0.0612	0.5550	0.5417	**0.5964**

Table 11
Network complementarity after training. The row shows the complementarity of all the networks for the networks listed in the column.

Networks	YOLACT	SF-Mask	UOC	UOIS
YOLACT	X	**0.5261**	0.5255	0.4458
SF-mask	0.1497	X	**0.5585**	0.4834
UOC	0.1494	**0.6095**	X	0.5244
UOIS	0.1162	**0.5739**	0.5732	X



(a) True positive detections ($mIoU > 0.5$).



(b) False detections – ($mIoU < 0.15$).

Fig. 6. Approximation of the relation between the number of object instances to their size for the networks after additional training. The X axis shows the relative size of the object in % according to the image resolution. It is divided into 400 intervals. On the Y axis, the number of detected objects (in a certain interval) divided by the total number of objects is shown. The dashed black line shows the ground truth, and its integral is equal to 100%.

6. Discussion

The presented results show that there is still much room for improvement in terms of architecture universality. The conducted tests show that the architectures created by NVIDIA Labs (UOC and UOIS) suit the stack segmentation task best. Our final test indicates that merging of UOC and the SF-mask would better meet the expectations of a picking robot. The type of camera also affects the effectiveness of architectures. It seems that pure stereo-vision (ZED) cameras are inadequate for the heap segmentation task. Better results are obtained by datasets with additional active sensors (such as the Intel RealSense L-515).

Our future goal is to apply the selected architectures to the heap segmentation task. We would probably like to test a boosting algorithm

Table 12
Different backbones test for SF-Mask.

Networks	ResNet50	ResNet152	RegNet32	EfficientNet-B4
$mIoU$	0.8334	0.7203	0.8111	0.7397
$mIoU_a$	0.7244	0.5630	0.6892	0.6192
$mIoU_{max}$	0.8555	0.7813	0.8452	0.8143
$mIoU_{px}$	0.7419	0.7164	0.7312	0.7061
mP	0.8749	0.5976	0.5383	0.5765
mR	0.3181	0.4829	0.5430	0.4972
$mF1$	0.4666	0.5342	0.5406	0.5339
mR_i	0.8202	0.7330	0.8217	0.8068
TPR	0.8365	0.6955	0.8113	0.7752
PPV	0.8047	0.6237	0.7695	0.6711

with the SF-mask and UOC architectures. Adding another input/layer to the network (such as normals) may also improve the depth information. The change of camera to a high-quality one, such as the ZIVID two, is also planned.

6.1. Future works

Our superior project goal related to this article is to build a station for automatically picking unknown objects from the heap. For this reason, further work related to the project concerns fine-tuning the effectiveness of the selected segmentation architecture and tests related to its improvement. Next, we will deal with the aspect of determining the place of the grip and the clustering of unknown objects due to the features obtained based on the image. It is also possible to predict the shape of objects as in Landgraf et al. (2021).

It should also be noted that there are already solutions that merge instance segmentation with the determination of the pick-up point, e.g. for a suction cup (Fu et al., 2022; Zhao et al., 2022). However, our assumption is to determine the pick-up point for both a suction cup and a gripper equipped with two or three fingers. Therefore, we cannot use existing solutions.

Among future works, we also plan to expand the comparison of architectures with newly created ones. Currently, these include Back et al. (2022) and Cheng et al. (2021). Unfortunately, most of the emerging segmentation architectures are dedicated to the external outdoor environment, and not to taking objects from a stack.

6.2. Future trends

Let us take into account current trends in artificial intelligence. A number of methods can be distinguished to increase the effectiveness of heap segmentation. Among them, several primary aspects should be denoted: changing the construction of the selected architecture or creating a new one, as suggested by Zhou et al. (2022), and the use of more advanced learning methods. Unfortunately, looking at the same trends, architectures for semantic segmentation of both indoor and outdoor environments are gaining much more popularity, rather than heaps of unknown objects (Jiang et al., 2022; Guo and Yang, 2022; Seth et al., 2022; Mo et al., 2022).

Recently, neural networks based on (vision) transformers (such as ChatGPT3 — a chat based on Generative Pre-trained Transformer 3) are gaining more and more popularity (Gu et al., 2022). Among the many applications of vision transformers, one can distinguish image recognition, object detection, and segmentation, especially semantic ones. For example, an architecture called Segmenter scored mIoU 10 points higher than DeepLabv3+ in a semantic segmentation task for an outdoor environment in tests using the PASCAL Context (Strudel et al., 2021). TopFormer, on the other hand, has comparable performance to DeepLabv3+ when tested on the ADE20K dataset based on an indoor environment (Zhang et al., 2022). There are even video transformers that allow the use of multimodal data — depth or thermal imaging (Wang et al., 2022; Liu et al., 2021b). Unfortunately, according to our knowledge, there is no architecture yet that uses a transformer to segment the instance heap of objects. It would be worth creating such one.

At the same time, the generative approach to various aspects of image processing has been developing more and more recently. Text-to-image architectures already exist, such as DALLE-2 (Patel et al., 2022). Artificial face generators have also been developed. There also exists generative architectures for landscapes generated using pseudo paint, where a landscape scheme is drawn in different colors (Park et al., 2019; Jo and Park, 2019). The reverse approach would also work in our case. For this purpose, a Generative Adversarial Network (GAN) should be used to color various objects. This approach is already used in medical imaging (Cirillo et al., 2021; Dong et al., 2019). Moreover, a segmentation architecture based on GAN also exists, but for outdoor scenes (Liu et al., 2021a).

Another aspect that can be used in our case is the use of graph convolutional neural networks currently used for biological images segmentation (Zhang et al., 2019b; Xu et al., 2022b) or just graph neural networks (Xie et al., 2021b; Wang et al., 2019; Gao et al., 2022). For semantic segmentation, they have had pretty good success (Johnander et al., 2020). In the case of graph networks, there is already some approach to segmenting the object heap (Xie et al., 2022). RICE: Refining Instance Masks in Cluttered Environments with Graph Neural Networks allows you to specify the relationship between objects on the heap.

On the other hand, instead of interfering with the architecture, we can change the method of training neural networks. One of the newer trends in such a case is the use of zero/one/few-shot learning (Wang et al., 2019; Kang and Cho, 2022). In addition, switching to meta-learning methods may also prove effective in our case (Luo et al., 2022).

CRediT authorship contribution statement

Michał Czubenko: Conceptualization, Methodology, Formal analysis, Writing – original draft, Writing – review & editing, Supervision. **Artur Chrzanowski:** Data curation, Software. **Rafał Okuński:** Software, Visualization, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: The company reports financial support was provided by The National Centre for Research and Development.

Data availability

Data will be made available on request.

Acknowledgments

The project is co-funded by the European Union from the European Regional Development Fund as part of the Smart Growth Program. The project is implemented as part of the National Center for Research and Development (NCBR): Fast Track, activities: R+D projects for enterprises, sub-activity: Industrial research and development works carried out by enterprises; POIR.01.01.01-00-1685/20.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.engappai.2023.106942.

References

Ainetter, S., Fraundorfer, F., 2021. End-to-end trainable deep neural network for robotic grasp detection and semantic segmentation from rgb. In: 2021 IEEE International Conference on Robotics and Automation. ICRA, IEEE, pp. 13452–13458.

Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M., 2019. Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

Alhaija, H.A., Mustikovela, S.K., Mescheder, L., Geiger, A., Rother, C., 2018. Augmented reality meets computer vision: Efficient data generation for urban driving scenes. Int. J. Comput. Vis. 126 (9), 961–972.

Andrews, S., Tsochantaridis, I., Hofmann, T., 2002. Support vector machines for multiple-instance learning. In: NIPS. p. 7.

Armeni, I., Sax, S., Zamir, A.R., Savarese, S., 2017. Joint 2d-3d-semantic data for indoor scene understanding. arXiv preprint arXiv:1702.01105.

Back, S., Kim, J., Kang, R., Choi, S., Lee, K., 2020. Segmenting unseen industrial components in a heavy clutter using rgb-d fusion and synthetic data. In: 2020 IEEE International Conference on Image Processing. ICIP, IEEE, pp. 828–832.

Back, S., Lee, J., Kim, T., Noh, S., Kang, R., Bak, S., Lee, K., 2022. Unseen object amodal instance segmentation via hierarchical occlusion modeling. In: 2022 International Conference on Robotics and Automation. ICRA, IEEE, pp. 5085–5092.

Baig, M.M., Awais, M.M., El-Alfy, E.-S.M., 2017. AdaBoost-based artificial neural network learning. Neurocomputing 248, 120–126.

Benenson, R., Popov, S., Ferrari, V., 2019. Large-scale interactive object segmentation with human annotators. In: CVPR.

Bolya, D., Zhou, C., Xiao, F., Lee, Y.J., 2019. YOLACT: Real-time instance segmentation. In: ICCV.

Bousmalis, K., Silberman, N., Dohan, D., Erhan, D., Krishnan, D., 2017. Unsupervised pixel-level domain adaptation with generative adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3722–3731.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S., 2020. End-to-end object detection with transformers. In: European Conference on Computer Vision. Springer, pp. 213–229.

Chai, D., Newsam, S., Huang, J., 2020. Aerial image semantic segmentation using DCNN predicted distance maps. ISPRS J. Photogramm. Remote Sens. 161, 309–322.

Chen, L.-C., Papandreou, G., Schroff, F., Adam, H., 2017. Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H., 2018a. Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European Conference on Computer Vision. ECCV, pp. 801–818.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H., 2018b. Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H., 2022. DeepLab: Deep labelling for semantic image segmentation. https://github.com/tensorflow/models/tree/master/research/deeplab.

Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R., 2021. Masked-attention mask transformer for universal image segmentation. arXiv.

Chollet, F., 2017. Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1251–1258.

Cirillo, M.D., Abramian, D., Eklund, A., 2021. Vox2Vox: 3D-GAN for brain tumour segmentation. In: Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 6th International Workshop, BrainLes 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4, 2020, Revised Selected Papers, Part I 6. Springer, pp. 274–284.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B., 2016. The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3213–3223.

Coşkun, M., Uçar, A., Yildirim, Ö., Demir, Y., 2017. Face recognition based on convolutional neural network. In: 2017 International Conference on Modern Electrical and Energy Systems. MEES, IEEE, pp. 376–379.

Czubenko, M., Kowalczuk, Z., 2021. A simple neural network for collision detection of collaborative robots. Sensors 21 (12), 4235.

Czubenko, M., Kowalczuk, Z., Żmuda-Trzebiatowska, W., 2022. Recognition of emotions in dog behavior based on deep neural network. Int. J. Comput. Intell. 38 (6), 2116–2133.

Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M., 2017. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In: Proc. Computer Vision and Pattern Recognition (CVPR). IEEE.

Danielczuk, M., Matl, M., Gupta, S., Li, A., Lee, A., Mahler, J., Goldberg, K., 2019. Segmenting unknown 3d objects from real depth images using mask r-cnn trained on synthetic data. In: 2019 International Conference on Robotics and Automation. ICRA, IEEE, pp. 7283–7290.

Dong, X., Lei, Y., Wang, T., Thomas, M., Tang, L., Curran, W.J., Liu, T., Yang, X., 2019. Automatic multiorgan segmentation in thorax CT images using U-net-GAN. Med. Phys. 46 (5), 2157–2168.

Durner, M., Boerdijk, W., Sundermeyer, M., Friedl, W., Marton, Z.-C., Triebel, R., 2021. Unknown object segmentation from stereo images. arXiv:2103.06796.

Everingham, M., Eslami, S.A., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A., 2015. The pascal visual object classes challenge: A retrospective. Int. J. Comput. Vis. 111 (1), 98–136.

Everingham, M., Winn, J., 2011. The pascal visual object classes challenge 2012 (voc2012) development kit. In: Pattern Analysis, Statistical Modelling and Computational Learning, Vol. 8. Tech. Rep, p. 5.

Firman, M., 2016. RGBD datasets: Past, present and future. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 19–31.

Fu, Y., Sun, T., Wang, L., Yu, S., Deng, L., Chen, B., Li, L., Xie, Y., Deng, S., Yin, H., 2022. RGB-D instance segmentation-based suction point detection for grasping. In: 2022 IEEE International Conference on Robotics and Biomimetics. ROBIO, IEEE, pp. 1643–1650.

Gählert, N., Jourdan, N., Cordts, M., Franke, U., Denzler, J., 2020. Cityscapes 3d: Dataset and benchmark for 9 dof vehicle detection. arXiv preprint arXiv:2006.07864.

Gao, H., Xiao, J., Yin, Y., Liu, T., Shi, J., 2022. A mutually supervised graph attention network for few-shot segmentation: the perspective of fully utilizing limited samples. IEEE Trans. Neural Netw. Learn. Syst..

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Garcia-Rodriguez, J., 2017. A review on deep learning techniques applied to semantic segmentation. arXiv:1704.06857.

Gatys, L.A., Ecker, A.S., Bethge, M., 2016. Image style transfer using convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2414–2423.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

Gu, J., Kwon, H., Wang, D., Ye, W., Li, M., Chen, Y.-H., Lai, L., Chandra, V., Pan, D.Z., 2022. Multi-scale high-resolution vision transformer for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12094–12103.

Guo, Y., Yang, B., 2022. A survey of semantic segmentation methods in traffic scenarios. In: 2022 International Conference on Machine Learning, Cloud Computing and Intelligent Mining. MLCCIM, IEEE, pp. 452–457.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

Hossain, M.Z., Sohel, F., Shiratuddin, M.F., Laga, H., 2019. A comprehensive survey of deep learning for image captioning. ACM Comput. Surv. (CsUR) 51 (6), 1–36.

Hu, Y., Chen, Z., Lin, W., 2018. RGB-D semantic segmentation: a review. In: 2018 IEEE International Conference on Multimedia & Expo Workshops. ICMEW, IEEE, pp. 1–6.

Janoch, A., 2012. In: Darrell, T., Abbeel, P., Malik, J. (Eds.), The Berkeley 3D Object Dataset (Master's thesis). EECS Department, University of California, Berkeley, URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-85.html.

Jiang, X., Han, L., Liu, H., Nie, S., Wang, S., Wen, Y., 2022. Deep learning based 3D object detection in indoor environments: A review. In: 2022 6th CAA International Conference on Vehicular Control and Intelligence. CVCI, IEEE, pp. 1–6.

Jo, Y., Park, J., 2019. Sc-fegan: Face editing generative adversarial network with user's sketch and color. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1745–1753.

Johnander, J., Brissman, E., Danelljan, M., Felsberg, M., 2020. Learning video instance segmentation with recurrent graph neural networks. arXiv preprint arXiv:2012.03911.

Jozefowicz, R., Zaremba, W., Sutskever, I., 2015. An empirical exploration of recurrent network architectures. In: International Conference on Machine Learning. PMLR, pp. 2342–2350.

Kang, D., Cho, M., 2022. Integrative few-shot learning for classification and segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9979–9990.

Kaymak, Ç., Uçar, A., 2019. A brief survey and an application of semantic image segmentation for autonomous driving. In: Handbook of Deep Learning Applications. Springer, pp. 161–200.

Kirillov, A., He, K., Girshick, R., Rother, C., Dollár, P., 2019. Panoptic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9404–9413.

Kousi, N., Gkournelos, C., Aivaliotis, S., Lotsaris, K., Bavelos, A.C., Baris, P., Michalos, G., Makris, S., 2021. Digital twin for designing and reconfiguring human–robot collaborative assembly lines. Appl. Sci. 11 (10), 4620.

Krasin, I., Duerig, T., Alldrin, N., Ferrari, V., Abu-El-Haija, S., Kuznetsova, A., Rom, H., Uijlings, J., Popov, S., Kamali, S., Malloci, M., Pont-Tuset, J., Veit, A., Belongie, S., Gomes, V., Gupta, A., Sun, C., Chechik, G., Cai, D., Feng, Z., Narayanan, D., Murphy, K., 2017. OpenImages: A public dataset for large-scale multi-label and multi-class image classification. Dataset available from https://storage.googleapis.com/openimages/web/index.html.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. 25, 1097–1105.

Kuo, W., Angelova, A., Malik, J., Lin, T.-Y., 2019. Shapemask: Learning to segment novel objects by refining shape priors. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9207–9216.

Lai, K., Bo, L., Ren, X., Fox, D., 2011. A large-scale hierarchical multi-view rgb-d object dataset. In: 2011 IEEE International Conference on Robotics and Automation. IEEE, pp. 1817–1824.

Landgraf, Z., Scona, R., Laidlow, T., James, S., Leutenegger, S., Davison, A.J., 2021. Simstack: A generative shape and instance model for unordered object stacks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13012–13022.

Lateef, F., Ruichek, Y., 2019. Survey on semantic segmentation using deep learning techniques. Neurocomputing 338, 321–348.

Li, Y., Birchfield, S.T., 2010. Image-based segmentation of indoor corridor floors for a mobile robot. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 837–843.

Li, F., Chen, W., Xu, W., Huang, L., Li, D., Cai, S., Yang, M., Xiong, X., Liu, Y., Li, W., 2020. A mobile robot visual SLAM system with enhanced semantics segmentation. IEEE Access 8, 25442–25458.

Li, X., Zhong, Z., Wu, J., Yang, Y., Lin, Z., Liu, H., 2019. Expectation-maximization attention networks for semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9167–9176.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context. In: European Conference on Computer Vision. Springer, pp. 740–755.

Liu, C., Chen, L.-C., Schroff, F., Adam, H., Hua, W., Yuille, A.L., Fei-Fei, L., 2019. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 82–92.

Liu, Z., Tan, Y., He, Q., Xiao, Y., 2021b. SwinNet: Swin transformer drives edge-aware RGB-D and RGB-T salient object detection. IEEE Trans. Circuits Syst. Video Technol. 32 (7), 4486–4497.

Liu, K., Ye, Z., Guo, H., Cao, D., Chen, L., Wang, F.-Y., 2021a. FISS GAN: A generative adversarial network for foggy image semantic segmentation. IEEE/CAA J. Autom. Sin. 8 (8), 1428–1439.

Luo, S., Li, Y., Gao, P., Wang, Y., Serikawa, S., 2022. Meta-seg: A survey of meta-learning for image segmentation. Pattern Recognit. 108586.

Masłowski, D., Czubenko, M., 2019. Safety system for an industrial cooperating robot based on depth cameras. Pomiary Autom. Robot. 23, in Polish.

Minaee, S., Boykov, Y.Y., Porikli, F., Plaza, A.J., Kehtarnavaz, N., Terzopoulos, D., 2021. Image segmentation using deep learning: A survey. IEEE Trans. Pattern Anal. Mach. Intell..

Mo, Y., Wu, Y., Yang, X., Liu, F., Liao, Y., 2022. Review the state-of-the-art technologies of semantic segmentation based on deep learning. Neurocomputing 493, 626–646.

Murphy, J., 2016. An Overview of Convolutional Neural Network Architectures for Deep Learning. Microway Inc.

Ohta, Y.-i., Kanade, T., Sakai, T., 1978. An analysis system for scenes containing objects with substructures. In: Proceedings of the Fourth International Joint Conference on Pattern Recognitions. pp. 752–754.

Papandreou, G., Chen, L.-C., Murphy, K.P., Yuille, A.L., 2015. Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1742–1750.

Park, J., Hwang, D., Kim, K.Y., Kang, S.K., Kim, Y.K., Lee, J.S., 2018. Computed tomography super-resolution using deep convolutional neural network. Phys. Med. Biol. 63 (14), 145011.

Park, T., Liu, M.-Y., Wang, T.-C., Zhu, J.-Y., 2019. Gaugan: semantic image synthesis with spatially adaptive normalization. In: ACM SIGGRAPH 2019 Real-Time Live!. p. 1.

Patel, M., Fatangare, S., Nasare, A., Pachpute, A., 2022. Image-dev: An advance text to image AI model. In: 2022 IEEE Pune Section International Conference (PuneCon). IEEE, pp. 1–6.

Ponte, H., Ponte, N., Crowder, S., 2021. zpy: Synthetic Data for Blender, Vol. 1. GitHub, Note: https://github.com/ZumoLabs/zpy.

Popov, S., Bauszat, P., Ferrari, V., 2020. CoReNet: Coherent 3D scene reconstruction from a single RGB image. In: European Conference on Computer Vision. Springer, pp. 366–383.

Porzi, L., Bulo, S.R., Kontschieder, P., 2021. Improving panoptic segmentation at all scales. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7302–7311.

Qu, Y., Chen, Y., Huang, J., Xie, Y., 2019. Enhanced pix2pix dehazing network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8160–8168.

Richtsfeld, A., Mörwald, T., Prankl, J., Zillich, M., Vincze, M., 2012. Segmentation of unknown objects in indoor environments. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 4791–4796.

Rossi, F., Pini, F., Carlesimo, A., Dalpadulo, E., Blumetti, F., Gherardini, F., Leali, F., 2020. Effective integration of Cobots and additive manufacturing for reconfigurable assembly solutions of biomedical products. Int. J. Interact. Des. Manuf. (IJIDeM) 14, 1085–1089.

Sakib, S., Ahmed, N., Kabir, A.J., Ahmed, H., 2019. An overview of convolutional neural network: its architecture and applications.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C., 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4510–4520.

Sarker, M.M.K., Makhlouf, Y., Craig, S.G., Humphries, M.P., Loughrey, M., James, J.A., Salto-Tellez, M., O'Reilly, P., Maxwell, P., 2021. A means of assessing deep learning-based detection of ICOS protein expression in colon cancer. Cancers 13 (15), 3825.

Schnieders, B., Palmer, G., Luo, S., Tuyls, K., 2019. Fully convolutional one-shot object segmentation for industrial robotics. In: Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, Vol. 2.

Sellami, A., Farah, M., Farah, I.R., Solaiman, B., 2019. Hyperspectral imagery classification based on semi-supervised 3-D deep neural network and adaptive band selection. Expert Syst. Appl. 129, 246–259.

Serra, J., 2006. A lattice approach to image segmentation. J. Math. Imaging Vision 24 (1), 83–130.

Seth, A., Sharma, S., et al., 2022. State of the art techniques to advance deep networks for semantic segmentation: A systematic review. U. Porto J. Eng. 8 (6), 132–159.

Shabbir, A., Ali, N., Ahmed, J., Zafar, B., Rasheed, A., Sajid, M., Ahmed, A., Dar, S.H., 2021. Satellite and scene image classification based on transfer learning and fine tuning of ResNet50. Math. Probl. Eng. 2021, 1–18.

Shen, F., Gan, R., Yan, S., Zeng, G., 2017. Semantic segmentation via structured patch prediction, context crf and guidance crf. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1953–1961.

Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Song, S., Lichtenberg, S.P., Xiao, J., 2015. Sun rgb-d: A rgb-d scene understanding benchmark suite. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 567–576.

Staar, B., Lütjen, M., Freitag, M., 2019. Anomaly detection with convolutional neural networks for industrial surface inspection. Procedia CIRP 79, 484–489.

Strudel, R., Garcia, R., Laptev, I., Schmid, C., 2021. Segmenter: Transformer for semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7262–7272.

Suchi, M., Patten, T., Fischinger, D., Vincze, M., 2019. EasyLabel: A semi-automatic pixel-wise object annotation tool for creating robotic RGB-D datasets. In: International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019. IEEE, pp. 6678–6684. http://dx.doi.org/10.1109/ICRA.2019.8793917.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2016. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2818–2826.

Taghanaki, S.A., Abhishek, K., Cohen, J.P., Cohen-Adad, J., Hamarneh, G., 2021. Deep semantic segmentation of natural and medical images: a review. Artif. Intell. Rev. 54 (1), 137–178.

Tan, M., Le, Q., 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning. PMLR, pp. 6105–6114.

Tang, M., Djelouah, A., Perazzi, F., Boykov, Y., Schroers, C., 2018. Normalized cut loss for weakly-supervised cnn segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1818–1827.

Toldo, M., Maracani, A., Michieli, U., Zanuttigh, P., 2020. Unsupervised domain adaptation in semantic segmentation: a review. Technologies 8 (2), 35.

Treml, M., Arjona-Medina, J., Unterthiner, T., Durgesh, R., Friedmann, F., Schuberth, P., Mayr, A., Heusel, M., Hofmarcher, M., Widrich, M., et al., 2016. Speeding up semantic segmentation for autonomous driving. In: 29th Conference on Neural Information Processing Systems (NIPS 2016). Barcelona, Spain.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Adv. Neural Inf. Process. Syst. 30.

Vezhnevets, A., Buhmann, J.M., 2010. Towards weakly supervised semantic segmentation by means of multiple instance and multitask learning. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE, pp. 3249–3256.

Viazovetskyi, Y., Ivashkin, V., Kashin, E., 2020. Stylegan2 distillation for feed-forward image manipulation. In: European Conference on Computer Vision. Springer, pp. 170–186.

van der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E., Yu, T., the scikit-image contributors, 2014. scikit-image: image processing in Python. PeerJ 2, e453. http://dx.doi.org/10.7717/peerj.453.

Wang, Y., Chen, X., Cao, L., Huang, W., Sun, F., Wang, Y., 2022. Multimodal token fusion for vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12186–12195.

Wang, M., Deng, W., 2018. Deep visual domain adaptation: A survey. Neurocomputing 312, 135–153.

Wang, W., Lu, X., Shen, J., Crandall, D.J., Shao, L., 2019. Zero-shot video object segmentation via attentive graph neural networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9236–9245.

Wu, C.-Y., Hu, X., Happold, M., Xu, Q., Neumann, U., 2020. Geometry-aware instance segmentation with disparity maps. In: CVPR Workshop on Scability in Autonomous Driving.

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., Girshick, R., 2019. Detectron2. https://github.com/facebookresearch/detectron2.

Xiang, Y., Xie, C., Mousavian, A., Fox, D., 2020. Learning rgb-d feature embeddings for unseen object instance segmentation. arXiv preprint arXiv:2007.15157.

Xie, G.-S., Liu, J., Xiong, H., Shao, L., 2021b. Scale-aware graph neural network for few-shot semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5475–5484.

Xie, C., Mousavian, A., Xiang, Y., Fox, D., 2022. Rice: Refining instance masks in cluttered environments with graph neural networks. In: Conference on Robot Learning. PMLR, pp. 1655–1665.

Xie, C., Xiang, Y., Mousavian, A., Fox, D., 2020. The best of both modes: Separately leveraging rgb and depth for unseen object instance segmentation. In: Conference on Robot Learning. PMLR, pp. 1369–1378.

Xie, C., Xiang, Y., Mousavian, A., Fox, D., 2021a. Unseen object instance segmentation for robotic environments. IEEE Trans. Robot..

Xu, R., Li, Y., Wang, C., Xu, S., Meng, W., Zhang, X., 2022b. Instance segmentation of biological images using graph convolutional network. Eng. Appl. Artif. Intell. 110, 104739.

Xu, J., Pan, Y., Pan, X., Hoi, S., Yi, Z., Xu, Z., 2022a. RegNet: self-regulated network for image classification. IEEE Trans. Neural Netw. Learn. Syst..

Yu, H., Yang, Z., Tan, L., Wang, Y., Sun, W., Sun, M., Tang, Y., 2018. Methods and datasets on semantic segmentation: A review. Neurocomputing 304, 82–103.

Yuan, X., Shi, J., Gu, L., 2021. A review of deep learning methods for semantic segmentation of remote sensing imagery. Expert Syst. Appl. 169, 114417.

Zhang, R., Du, L., Xiao, Q., Liu, J., 2020b. Comparison of backbones for semantic segmentation network. In: Journal of Physics: Conference Series. IOP Publishing, 012196.

Zhang, W., Huang, Z., Luo, G., Chen, T., Wang, X., Liu, W., Yu, G., Shen, C., 2022. TopFormer: Token pyramid transformer for mobile semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12083–12093.

Zhang, S., Tong, H., Xu, J., Maciejewski, R., 2019b. Graph convolutional networks: a comprehensive review. Comput. Soc. Netw. 6 (1), 1–23.

Zhang, Z., Wu, Q., Wang, Y., Chen, F., 2018. High-quality image captioning with fine-grained and semantic-guided visual attention. IEEE Trans. Multimed. 21 (7), 1681–1693.

Zhang, J., Zhao, X., Chen, Z., Lu, Z., 2019a. A review of deep learning-based semantic segmentation for point cloud. IEEE Access 7, 179118–179133.

Zhang, M., Zhou, Y., Zhao, J., Man, Y., Liu, B., Yao, R., 2020a. A survey of semi- and weakly supervised semantic segmentation of images. Artif. Intell. Rev. 53 (6), 4259–4288.

Zhao, X., Xie, P., Wang, M., Li, W., Pickhardt, P.J., Xia, W., Xiong, F., Zhang, R., Xie, Y., Jian, J., et al., 2020. Deep learning–based fully automated detection and segmentation of lymph nodes on multiparametric-mri for rectal cancer: A multicentre study. EBioMedicine 56, 102780.

Zhao, Y., Yang, J., Wang, S., Li, X., 2022. Towards one shot & pick all: 3d-oas, an end-to-end framework for vision guided top-down parcel bin-picking using 3d overlapping-aware instance segmentation and gnn. Preprint submitted to Robotic and automatic system.

Zhou, T., Wang, W., Konukoglu, E., Van Gool, L., 2022. Rethinking semantic segmentation: A prototype view. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2582–2593.

Zoph, B., Ghiasi, G., Lin, T.-Y., Cui, Y., Liu, H., Cubuk, E.D., Le, Q.V., 2020. Rethinking pre-training and self-training. arXiv preprint arXiv:2006.06882.

Zou, Y., Yu, Z., Liu, X., Kumar, B., Wang, J., 2019. Confidence regularized self-training. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5982–5991.