applied sciences

MDPI

Article

# Deep Instance Segmentation of Laboratory Animals in Thermal Images

**Magdalena Mazur-Milecka *, Tomasz Kocejko * and Jacek Ruminski ***

Department of Biomedical Engineering, Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technology, 80-233 Gdansk, Poland

* Correspondence: magdalena.milecka@pg.edu.pl (M.M.-M.); tomasz.kocejko@pg.edu.pl (T.K.); jacek.ruminski@pg.edu.pl (J.R.)

Received: 9 July 2020; Accepted: 25 August 2020; Published: 28 August 2020


**Abstract:** In this paper we focus on the role of deep instance segmentation of laboratory rodents in thermal images. Thermal imaging is very suitable to observe the behaviour of laboratory animals, especially in low light conditions. It is an non-intrusive method allowing to monitor the activity of animals and potentially observe some physiological changes expressed in dynamic thermal patterns. The analysis of the recorded sequence of thermal images requires smart algorithms for automatic processing of millions of thermal frames. Instance image segmentation allows to extract each animal from a frame and track its activity and thermal patterns. In this work, we adopted two instance segmentation algorithms, i.e., Mask R-CNN and TensorMask. Both methods in different configurations were applied to a set of thermal sequences, and both achieved high results. The best results were obtained for the TensorMask model, initially pre-trained on visible light images and finally trained on thermal images of rodents. The achieved mean average precision was above 90 percent, which proves that model pre-training on visible images can improve results of thermal image segmentation.

**Keywords:** instance segmentation; deep learning; computer vision

## 1. Introduction

Laboratory animals' behavior analysis is often used in studies of stress, anxiety, depression or neurodegenerative diseases [1]. The automation of this analysis has undoubtedly many advantages, among which objectivity and standardization are one of the most desirable. As far as the position and motion analysis are easy and commonly used parameters [2], the action or behavior recognition are quite difficult to automatize. Existing systems for laboratory animals' behavior analysis are restricted by, for example, the number or the color of the objects [2]. The spectrum of behavior analysis performed by the available systems is usually limited to exploration, rest or grooming. All these behaviors are detected based on simple object parameters representing the position, speed, direction of the movement or the shape parameters of the observed rodent [3]. Objective detection and analysis of more complex behaviors would open up new possibilities. An important parameter of animals' health and well-being is its social behavior. It ensures survival of the species and is the specific characteristics of a single individual. Futhermore, deviations and abnormalities of social actions may be related to stress, fear or illness [1]. Social behaviors are divided into three categories: aggressive, defensive and neutral [4]. An example of aggressive behavior is an attack, bite or aggressive grooming. However, gentle grooming can also be a form of defense, it can also be confused with other behavior—climbing. All these insignificant differences make the classification of complex behaviors an extremely difficult task, mainly performed by human observers.

The presence of a man who, after all, is a rodent's natural enemy, probably adversely affects the results of behavior research [5]. This can be solved by using camcorders and analyze the behavior from the recordings. An additional advantage of this solution is the ability to slow down or stop the action, certainly helpful during fast animal movements, e.g., fights. Nevertheless, the need for a good lighting for cameras is a very unfavorable and stressful factor for nocturnal animals such as rodents [6,7]. This is where technology that offers alternative imaging comes to the rescue. Thermal camcorders record the surface temperature and can work in limited light and even in complete darkness. In addition, they provide data of the object's surface temperature distribution, proved to be useful in early disease diagnosis [8,9]. Changes in the rodent's surface temperature can be an indicator of his health condition, behavioral changes, anxiety or stress [10,11]. In paper [12] authors use mouse surface temperature distribution as an object identifier to recover tracking algorithm identity swap after close contact.

The automatic analysis of rodents from video recordings requires the identification and tracking of each animal in a scene. The distinction between the object and the background in animal tracking visual systems is most often based on difference between frames, color threshold or color matching [13]. A clear temperature difference in thermal imaging greatly simplifies foreground-background segmentation. Usually, the Otsu's thresholding supported by morphology is enough [2,14]. Figure 1b demonstrates the result of Otsu's thresholding on thermal image. In a cluttered environment there is also a need to segment the object from other parts of the foreground or to distinguish individuals from each other. For small connections of the animals' bodies, methods such as marker-controlled watershed segmentation technique are sufficient [15]. However, the solution to the segmentation problem for the objects that overlap significantly is not easy and most image analysis methods fail. Figure 1c shows the results of rodent segmentation using watershed technique with markers build from the low gradient regions. There are no clear differences between the images of both objects and no border between them is visible. For these reasons a significant number of systems analyze only one individual at a time [2,13]. That is why the correct segmentation between objects is necessary for further analysis.

(**a**) (**b**) (**c**) (**d**)

**Figure 1.** Results of the most popular algorithms used for segmentation of (**a**) original image by: (**b**) Otsu's thresholding, (**c**) watershed with gradient regions markers, (**d**) U-Net—semantic segmentation.

Many segmentation techniques have been proposed in literature. However, the methods based on deep-learning have been recently found to provide the best segmentation results in many computer vision tasks. Segmentation techniques based on deep learning may be divided into two approaches: (i) semantic segmentation—pixel classification-based approach and (ii) instance segmentation–object detection and classification based approach.

Segmentation using (i) does not discriminate between instances. In paper [16] we have used popular methods of semantic segmentation—U-Net [17] and V-Net [18] to separate two objects. Algorithms correctly segmented animals in close contact, but they were not able to draw the boundary between the objects at the time of body overlapping. Figure 1d shows the result of U-Net architecture trained on 200 images on five epochs, 2000 steps each. The animals are not separated and are recognized together as one object.

In (ii) each object is detected, represented as a separate segment and labeled with the class name. Instance segmentation is one of the most challenging computer vision tasks. It was introduced by Hariharan [19] and then popularized by COCO [20]. Instance segmentation algorithms classify and

localize each object's bounding box while also precisely segment each instance. Both those subtasks can be performed as a single- or two-stage detection process.

### 1.1. Two-Stage Methods

Treating segmentation as an extension of the object detection is adopted by most of two-stage methods [21–24]. They are designed according to the dominant paradigm for instance segmentation which is: detect-then-segment. First, it detects an object with a box and then segments each object using the box as a guide.

Mask R-CNN [23] is one of a state of the art two-stage method of instance segmentation. It extends Fast [25] and Faster R-CNN [26] methods by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition [23]. This method detects object bounding boxes, and then crops and segments them to find the objects.

Many approaches based on Mask R-CNN also have dominated leaderboards of recent segmentation challenges.

In 2019 Mask Scoring R-CNN algorithm was introduced in [24]. This method focuses on scoring the predicted instance masks. The model learns an Intersection-over-Union (IoU) score for each mask and improves segmentation results by rewarding more accurate predictions during COCO AP evaluation.

The method named PANet presented in [27] improves Mask R-CNN in three steps by:

1. creating bottom-up path augmentation,
2. adaptive feature pooling,
3. augmenting mask prediction with fully connected layers.

The results outperform Mask R-CNN architecture but are re-implemented by the PANet authors.

Another extension of the Mask R-CNN model, this time with a cascade architecture, is the Hybrid Task Cascade (HTC) [28]. It improves the segmentation accuracy by incorporating cascade and multi-tasking at each stage and explores more contextual information from the spatial context.

State-of-the-art two-stage methods achieve outstanding results. However, due to the cascade strategy, these methods are usually time- and memory-consuming. That is why recently, performing instance detection and segmentation in a single-stage architecture [29–32] has become more and more popular, even though it does not yet perform as good as the two-stage methods.

### 1.2. Single-Stage Methods

Though single-stage methods simplify the procedure by removing the re-pooling step, usually they cannot produce masks that are as accurate as the two-stage methods. Some of those methods works in a per-pixel prediction way, similar to the semantic segmentation.

The algorithm Fully Convolutional One-Stage detection (FCOS) [33] has fully eliminated pre-defined anchor boxes, thus avoids complicated computation such as calculating boxes overlapping during training.

EmbedMask proposed in [32] unifies both segmentation- and proposal-based methods and takes the advantages of both. It is built on top of the detection models and thus has strong detection capabilities, just like the proposal-based methods. It applies extra embedding modules to generate embeddings for pixels and proposals, what enables EmbedMask to generate high-resolution masks without missing details from repooling, like segmentation-based methods.

TensorMask is an example of a dense sliding-window instance segmentation method [34]. The idea is to use structured 4D tensors to represent masks over a spatial domain. It performs multiclass classification in parallel to mask prediction. Authors show that this algorithm yields similar results to Mask R-CNN.

In this paper, we are mainly interested in the detection of each rodent in reference to each other and in reference to the background objects. Therefore, we focus on instance segmentation approach and

more specifically deep instance segmentation architectures represented by two different approaches: one- and two-stage methods. In particular, we addressed the problem of animals instance segmentation in close physical contact on thermal images. The aim of this study was also to verify whether thermal data bit depth reduction or re-scaling has an effect on the results.

The paper is organized as follows: Section 2 presents the general description of methods and materials. Results are introduced in Section 3 and discussed in Section 4. The last section concludes the work.

## 2. Methods

In this section we describe the methods for instance segmentation adopted by us for thermal image of laboratory animals. For this purpose we use thermal database of rats introduced in [35]. The database consists of 300 min of social behavior tests recordings. Every single test was carried out on two healthy male rats of the Wistar strain at the age of 12–16 weeks kept in a plexiglass cage (dimensions: 35 length $\times$ 45 width $\times$ 46 cm height) for about 17 min in dimly illuminated room with temperature about 22 Celsius degrees. Animals were not accustomed to the cage earlier, it was an unfamiliar environment. There were no food, drink nor bedding material in the cage. The image sequences were recorded by FLIR A320G camera situated 120 cm above the cage with the spatial resolution of 320 $\times$ 240 pixels, 60 fps and 16-bit image representation. The results of semantic segmentation on the same dataset are presented in paper [16].

The principles for the care and use of laboratory animals in research, as outlined by the Local Ethical Committee, were strictly followed and all the protocols were reviewed and approved by the Committee.

### *2.1. Data Preprocessing*

Thermal data were measured and stored with a resolution of 16-bit. The reduction of bit depth to 256-level standard images caused data loss. However, some data, e.g., background temperature, were redundant. In order to minimize data loss and investigate the effect of a temperature range selection, we proposed thermal frames processing to achieve five different images. This procedure is described in details in paper [16] and presented in Figure 2 in brief. First, the entire range of recorded raw thermal data (Figure 2a) was scaled to 256-level gray image (Figure 2b) and called orig image, as these were the original raw thermal data re-scaled to 8 bits. Then the gray-scale image was created from only three selected thermal ranges: animal body range (Figure 2c) and two thermal ranges (Figure 2d,e), all marked as ch1, ch2 and ch3 respectively. Minimal temperature value of the object was assumed to be equal to the threshold temperature between the background and the object calculated by the Otsu method, marked as a red line in the Figure 2a and called $maxOtsu_{thres}$. The green and blue dashed lines named as $T_2$ and $T_3$ respectively defined one- and two-thirds of the temperature interval from $maxOtsu_{thres}$ to the maximal value for all data. The ch1 range was selected between $maxOtsu_{thres}$ and the maximal temperature value. The ch2 image was created for the temperature range from $T_2$ to the maximal temperature value, and analogously, the lower limit of the ch3 image was $T_3$ value and the upper limit was the maximum temperature. Re-scaling 16-bit raw thermal data representation to 8-bit image caused loss of accuracy. Therefore, the last type of images was an image of the entire range of recorded data saved as a 16 bit image (16-bit). In [16] we proved that proper range selection improved the results of semantic segmentation.

### *2.2. Training Models*

In this study we chose one single- and one two-stage architectures to compare the instance segmentation methods on thermal images of laboratory rodent. The decision was made for the Mask R-CNN [23] and the TensorMask [34].

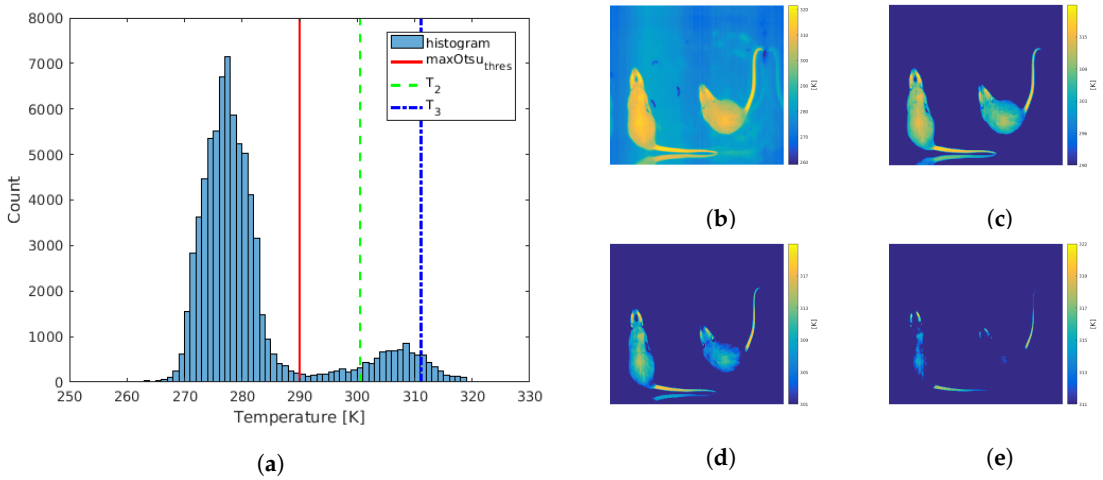


**Figure 2.** Range selection: (**a**) histogram of one frame raw thermal data with values of: minimal rodent's surface thermal value named $maxOtsu_{thres}$, marked with red solid line; the minimum values for images *ch2* and *ch3* marked as green ($T_2$) and blue ($T_3$) lines respectively, (**b**) orig image of the whole thermal range, (**c**) ch1 image of the first range: $maxOtsu_{thres}$ to $T_{max}$, (**d**) ch2 image of the second range: $T_2$ to $T_{max}$, (**e**) ch3 image of the third range: $T_3$ to $T_{max}$.

### *2.3. Learning Configurations*

We made two training and one testing manually segmented datasets. The only difference between training sets was their size: 200 and 500 images respectively. The smaller set was used for the transfer-learning training, where we used pre-trained network: Mask R-CNN with a ResNet-Feature Pyramid Network (FPN) backbone 50 layers deep trained on the COCO dataset [36], and similarly TensorMask with a ResNet-FPN backbone 50 layers deep [37], both implementations from Detectron2 v0.1.2.

The 500 images were used for training the whole architectures from random initialization. In this training the number of training iterations had to be increased so models could converge. However, even despite the very large number of iterations, the TensorMask architecture sometimes had convergence problems: the loss function increased over time. In such cases we have used normalization techniques according to [38], and replaced Frozen Batch Normalization with Group Normalization [39]. Group Normalization's accuracy was insensitive to batch sizes [39] and allowed for successful TensorMask model training from scratch.

The testing set consisted of 50 images not included in the training sets. Images in all data-sets depicted two rodents during physical contact of varying degrees: from a small part of the body contact (e.g., nose-to-nose) to the body overlapping and covering.

For every training model we performed three independent training sessions. Json files with ground-truth rat regions were manually created for every training and testing set using the VGG Image Annotator (VIA) [40]. The regions were marked on a zoomed image by a set of points forming a closed polygon.

To compare the trained and commonly available models, we tested the inference using weights pre-trained on MS COCO Dataset [20], Citiscapes [41] and LVIS [42] and not trained on our database. We have also used two different Python3 implementations of Mask R-CNN (Mask R-CNN2.1 [36], Detectron2 v0.1.2 [37]) and one for TensorMask Detectron2 v0.1.2 [37].

In total, we tested 24 different model configurations for pre-trained learning and 22 for training from scratch.

For both models and both training methods with the best segmentation results 3-fold cross-validation was performed. The first fold was made for all the model configurations. The second

and third folds used the whole testing data-set from the first fold in their training sets and replaced the training set with 50 different images selected from the training set of the first fold.

All experiments were performed using NVIDIA DGX-1 Station with Ubuntu 18.

### 2.4. Evaluation Metrics

To evaluate the results we are using three different detection metrics for bounding box- (bbox) and segmentation-level (segm): mean Average Precision (mAP), AP under Intersection-over-Union thresholds of 0.5 ($AP_{50}$) and 0.75 ($AP_{75}$).

The mean average precision (mAP) is actually the standard for the quantitative evaluation of object detection and instance segmentation methods [43,44]. The general definition for the Average Precision is finding the area under the precision-recall curve. The precision and recall are defined as follows:

$$Precision = \frac{TP}{TP + FP}; \quad Recall = \frac{TP}{TP + FN} \tag{1}$$

where: TP = True Positive, FP = False Positive and FN = False Negative.

The precision measures how accurate the prediction was, and recall measures how good all the positives were found.

In object detection and instance segmentation it is important to appropriately define a true positive and other types of results. As a current standard, the IoU metric was used. It was defined as the intersection between the predicted segment (bbox) and the actual segment (bbox) divided by their union:

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{2}$$

In this study the actual (ground truth) segment was manually drawn by the authors on each reference image.

A prediction is considered to be True Positive if the IoU value is higher than the threshold (e.g., IoU > 0.5), and False Positive otherwise. However, different thresholds could be used and the precision recall curve could be obtained. Then, the area under the curve could be calculated as:

$$AP = \int_0^1 p(r)dr, \tag{3}$$

where: p(r) is the precision-recall curve.

In practice, in a majority of current papers about instance segmentation, the interpolated curve is calculated in ranges of IoU threshold values from 0.5 to 0.95 with a step size of 0.05. This method was also used in this paper. The mAP metric calculates the mean of AP across all of the IoU thresholds. We also used AP at fixed IoUs: IoU = 0.5 ($AP_{50}$) and IoU = 0.75 ($AP_{75}$). Which means, that a prediction is considered as True Positive if its IoU is >0.5 and >0.75 respectively. Thus, the higher the threshold, the greater the requirement for the accuracy of matching the prediction to the ground-truth. The standard metrics used in this study allow direct comparison of future methods with the approach used in this paper.

We have evaluated bounding box AP for object detection and segmentation AP for instance segmentation. The metrics are measured for all five types of images (orig, ch1, ch2, ch3, 16-bit) and training parameters shown in Table 1.

**Table 1.** Training parameters that achieved the best results in training.

| Transfer Learning | | | | Training from Scratch | | | |
|---|---|---|---|---|---|---|---|
| Mask R-CNN | | TensorMask | | Mask R-CNN | | TensorMask | |
| **Batch** | **Epochs** | **Batch** | **Epochs** | **Batch** | **Epochs** | **Batch** | **Epochs** |
| 2 | 1000 | 2 | 2000 | 4 | 16,000 | **2** | **100,000** |
| 4 | 1000 | 4 | 1000 | 4 | 24,000 | 4 | 100,000 |
| **4** | **2000** | **4** | **2000** | 8 | 16,000 | | |
| | | 8 | 2000 | 8 | 24,000 | | |
| | | | | **2** | **100,000** | | |

## 3. Results

The best results among all the training parameter combinations were achieved by the values presented in Table 1.

The TensorMask's convergence problems during training from scratch forced the use of a large iteration number, much larger than for Mask R-CNN architecture. In order to compare the effectiveness of both architectures, we performed additional Mask R-CNN training with optimal parameters for TensorMask (2 batches, 100,000 epochs). Values in bold indicate parameters that have been selected for cross-validation.

In Tables 2 and 3 we compare Mask R-CNN testing results for pre-trained and from scratch learning respectively. Analogous measurements for the TensorMask architecture are presented in Tables 4 and 5.

**Table 2.** Testing results for Mask R-CNN pre-trained learning and 200 images.

| | **mAP** | **$AP_{50}$** | **$AP_{75}$** | **mAP** | **$AP_{50}$** | **$AP_{75}$** |
|---|---|---|---|---|---|---|
| **orig** | bbox | | | segm | | |
| 2 batch, 1000 epochs | 88.641 | 100 | 98.99 | 89.007 | 100 | 98.99 |
| 4 batch, 1000 epochs | **89.025** | 100 | **100** | **89.444** | 100 | **100** |
| 4 batch, 2000 epochs | 88.781 | 100 | 99 | 89.068 | 100 | 99 |
| **ch1** | bbox | | | segm | | |
| 2 batch, 1000 epochs | 87.332 | 100 | 100 | 89.289 | 100 | 100 |
| 4 batch, 1000 epochs | 86.942 | 100 | 100 | **89.715** | 100 | 100 |
| 4 batch, 2000 epochs | **90.205** | 100 | 100 | 89.655 | 100 | 100 |
| **ch2** | | | | | | |
| 2 batch, 1000 epochs | 85.252 | 100 | **99** | 87.323 | 100 | 100 |
| 4 batch, 1000 epochs | 85.847 | 100 | 98.98 | 88.256 | 100 | 100 |
| 4 batch, 2000 epochs | **87.915** | 100 | 98.98 | **88.893** | 100 | 100 |
| **ch3** | | | | | | |
| 2 batch, 1000 epochs | 63.719 | 99.114 | 73.409 | 52.901 | 93.649 | 59.722 |
| 4 batch, 1000 epochs | 69.028 | 99.417 | 80.98 | 55.181 | 97.284 | 60.683 |
| 4 batch, 2000 epochs | **73.908** | **99.952** | **87.824** | **65.687** | **98.902** | **83.376** |
| **16-bit** | bbox | | | segm | | |
| 2 batch, 1000 epochs | 88.128 | 100 | 99.01 | **89.604** | 100 | 100 |
| 4 batch, 1000 epochs | 89.965 | 100 | 100 | 89.194 | 100 | 100 |
| 4 batch, 2000 epochs | **90.549** | 100 | 100 | 89.417 | 100 | 100 |

**Table 3.** Training results for Mask R-CNN from scratch learning and 500 images.

| | **mAP** | **$AP_{50}$** | **$AP_{75}$** | **mAP** | **$AP_{50}$** | **$AP_{75}$** |
|---|---|---|---|---|---|---|
| **orig** | bbox | | | segm | | |
| 4 batch, 16,000 epochs | 81.811 | 100 | 98.307 | 85.586 | 100 | 100 |
| 4 batch, 24,000 epochs | 82.567 | 100 | 100 | 86.864 | 100 | 100 |
| 8 batch, 16,000 epochs | 81.658 | 100 | 100 | 84.728 | 100 | 98.584 |
| 8 batch, 24,000 epochs | 84.465 | 100 | 100 | 86.224 | 100 | 100 |
| 2 batch, 100,000 epochs | **86.695** | 100 | 100 | **89.569** | 100 | 100 |
| **ch1** | | | | | | |
| 4 batch, 16,000 epochs | 82.085 | 100 | 100 | 83.666 | 100 | 100 |
| 4 batch, 24,000 epochs | 84.368 | 100 | 100 | 85.821 | 100 | 100 |
| 8 batch, 16,000 epochs | 81.426 | 100 | 100 | 84.038 | 100 | 100 |
| 8 batch, 24,000 epochs | 85.153 | 100 | 98.772 | 86.389 | 100 | 100 |
| 2 batch, 100,000 epochs | **86.564** | 100 | 100 | **87.208** | 100 | 100 |
| **ch2** | | | | | | |
| 4 batch, 16,000 epochs | 80.629 | 100 | 98.95 | 80.896 | 100 | 100 |
| 4 batch, 24,000 epochs | 82.979 | 100 | 100 | **84.834** | 100 | 100 |
| 8 batch, 16,000 epochs | 80.983 | 100 | 100 | 79.931 | 100 | 100 |
| 8 batch, 24,000 epochs | **83.091** | 100 | 100 | 84.428 | 100 | 100 |
| 2 batch, 100,000 epochs | 82.55 | 100 | 97.55 | 82.397 | 100 | 100 |
| **ch3** | | | | | | |
| 4 batch, 16,000 epochs | 60.947 | 97.766 | 73.004 | 48.183 | 93.325 | 47.4 |
| 4 batch, 24,000 epochs | **67.739** | 98.113 | 79.816 | **58.782** | **96.882** | **69.92** |
| 8 batch, 16,000 epochs | 65.355 | 97.874 | 74.09 | 54.855 | 96.873 | 64.968 |
| 8 batch, 24,000 epochs | 66.014 | 98.4 | 78.156 | 57.452 | 95.887 | 63.699 |
| 2 batch, 100,000 epochs | 66.369 | **98.504** | **84.891** | 56.116 | 94.891 | 67.155 |
| **16-bit** | | | | | | |
| 4 batch, 16,000 epochs | 82.826 | 100 | 98.842 | 84.561 | 100 | 100 |
| 4 batch, 24,000 epochs | 84.29 | 100 | 100 | 86.111 | 100 | 100 |
| 8 batch, 16,000 epochs | 83.493 | 100 | 97.82 | 85.313 | 100 | 100 |
| 8 batch, 24,000 epochs | **85.596** | 100 | 100 | **86.607** | 100 | 100 |
| 2 batch, 100,000 epochs | 84.862 | 100 | 100 | 85.877 | 100 | 100 |

**Table 4.** Training results for TensorMask pre-trained learning and 200 images.

| | **mAP** | **$AP_{50}$** | **$AP_{75}$** | **mAP** | **$AP_{50}$** | **$AP_{75}$** |
|---|---|---|---|---|---|---|
| **orig** | bbox | | | segm | | |
| 2 batch, 2000 epochs | 90.646 | 100 | 98.01 | 89.82 | 100 | 100 |
| 4 batch, 1000 epochs | **91.264** | 100 | **99.01** | 89.877 | 100 | 99.01 |
| 4 batch, 2000 epochs | 90.916 | 100 | **99.01** | **90.158** | 100 | 100 |
| 8 batch, 2000 epochs | 90.672 | 100 | 98.96 | 89.824 | 100 | 100 |

**Table 4.** Cont.

| | mAP | $AP_{50}$ | $AP_{75}$ | mAP | $AP_{50}$ | $AP_{75}$ |
|---|---|---|---|---|---|---|
| **ch1** | | | | | | |
| 2 batch, 2000 epochs | 89.091 | 100 | 100 | 89.808 | 100 | **100** |
| 4 batch, 1000 epochs | 89.624 | 100 | 100 | 89.547 | 100 | 99.01 |
| 4 batch, 2000 epochs | **90.299** | 100 | 100 | **90.171** | 100 | 99.01 |
| 8 batch, 2000 epochs | 90.052 | 100 | 100 | 90.087 | 100 | 99.01 |
| **ch2** | | | | | | |
| 2 batch, 2000 epochs | 88.319 | 100 | 100 | **89.496** | 100 | 100 |
| 4 batch, 1000 epochs | 88.17 | 100 | 100 | 87.949 | 100 | 100 |
| 4 batch, 2000 epochs | 87.912 | 100 | 98.931 | 89.335 | 100 | 100 |
| 8 batch, 2000 epochs | **88.828** | 100 | 100 | 89.264 | 100 | 100 |
| **ch3** | | | | | | |
| 2 batch, 2000 epochs | 74.37 | 99.933 | 88.433 | 65.062 | **99.933** | 80.743 |
| 4 batch, 1000 epochs | 71.325 | 99.834 | 84.895 | 60.251 | 99.602 | 69.757 |
| 4 batch, 2000 epochs | 75.031 | 99.961 | **91.558** | 64.849 | 99.423 | 79.397 |
| 8 batch, 2000 epochs | **76.159** | **100** | 91.406 | **65.307** | 99 | **83.105** |
| **16-bit** | | | | | | |
| 2 batch, 2000 epochs | 91.248 | 100 | 100 | **90.249** | 100 | 100 |
| 4 batch, 1000 epochs | 90.443 | 100 | 98.951 | 89.857 | 100 | 98.951 |
| 4 batch, 2000 epochs | **91.609** | 100 | 100 | 90.199 | 100 | 100 |
| 8 batch, 2000 epochs | 90.939 | 100 | 100 | 90.171 | 100 | 98.99 |

**Table 5.** Training results for TensorMask from scratch learning and 500 images.

| | mAP | $AP_{50}$ | $AP_{75}$ | mAP | $AP_{50}$ | $AP_{75}$ |
|---|---|---|---|---|---|---|
| **orig** | bbox | | | segm | | |
| 2 batch, 100,000 epochs | **78.235** | **100** | 92.41 | 79.615 | **100** | **95.535** |
| 4 batch, 100,000 epochs | 76.413 | 99.99 | **95.435** | **80.371** | 99.99 | 91.678 |
| **ch1** | | | | | | |
| 2 batch, 100,000 epochs | **78.862** | 100 | **98.307** | 77.724 | 100 | 90.687 |
| 4 batch, 100,000 epochs | 77.725 | 100 | 94.535 | **81.259** | 100 | **95.188** |
| **ch2** | | | | | | |
| 2 batch, 100,000 epochs | 64.42 | 99.18 | 69.516 | 62.307 | 96.206 | 71.867 |
| 4 batch, 100,000 epochs | **75.275** | **99.833** | **90.342** | **74.431** | **99.833** | **89.674** |
| **ch3** | | | | | | |
| 2 batch, 100,000 epochs | 44.557 | 84.367 | 40.402 | 13.097 | 60.133 | 0.012 |
| 4 batch, 100,000 epochs | **55.982** | **95.072** | **57.009** | **32.752** | **89.025** | **16.88** |
| **16-bit** | | | | | | |
| 2 batch, 100,000 epochs | **78.112** | 99.99 | **97.29** | 80.198 | 99.99 | 94.435 |
| 4 batch, 100,000 epochs | 76.296 | **100** | 90.662 | **81.267** | **100** | **94.887** |

Figure 3 presents the exemplary testing results for two images (Figures 3a,b) which constituted quite a serious challenge for segmentation. For a better illustration, the contact area was zoomed and manually segmented with a yellow line in Figures 3c,d. The examples given include animals that overlapped one another. Figures 3e–h present the results of pre-trained learning for Mask R-CNN and TensorMask respectively. The instance segmentation made by those architectures trained from scratch are shown in Figures 3i–l.

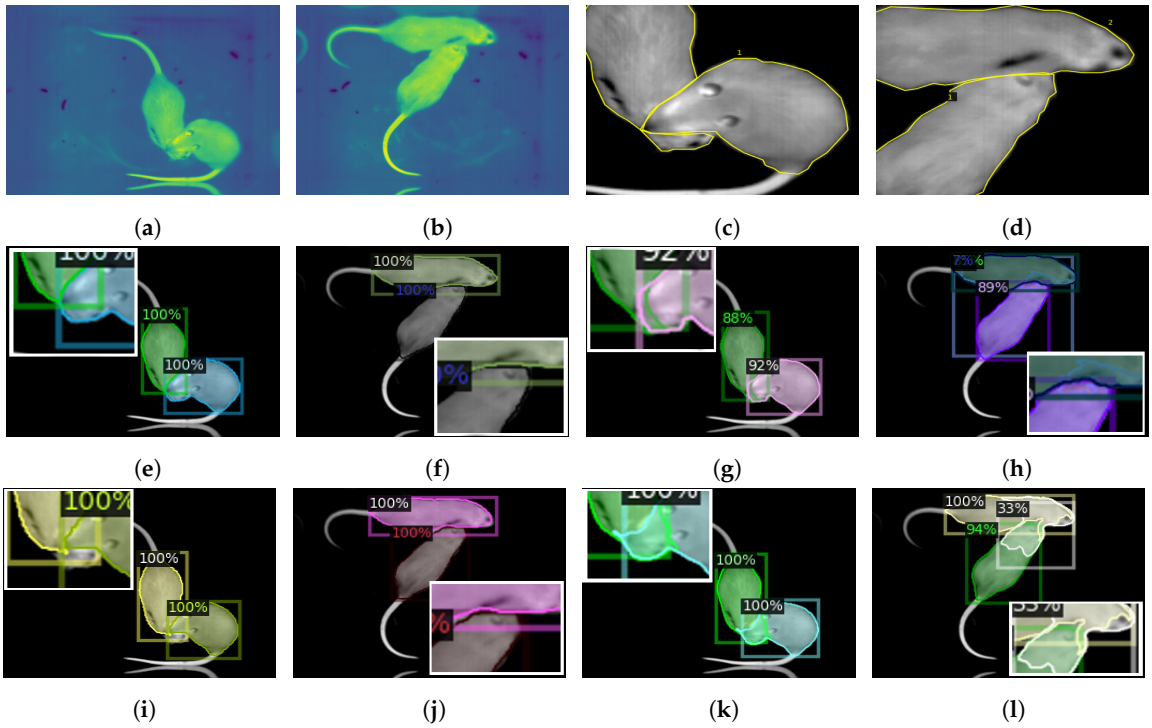


**Figure 3.** Examples of segmentation and detection prediction made for (**a**,**b**) the original images; (**c**,**d**) proper segmentation zoomed and marked with yellow lines. Results for: (**e**,**f**) pre-trained Mask R-CNN; (**g**,**h**) pre-trained TensorMask; (**i**,**j**) random initialized Mask R-CNN; (**k**,**l**) random initialized TensorMask.

Table 6 demonstrates the results of the inference made for *orig* images using commonly available models pre-trained only on MS COCO Dataset, Citiscapes and LVIS. We also used two different implementations of Mask R-CNN pre-trained on the COCO dataset.

**Table 6.** Segmentation results for commonly available models.

| | **mAP** | **$AP_{50}$** | **$AP_{75}$** | **mAP** | **$AP_{50}$** | **$AP_{75}$** |
|---|---|---|---|---|---|---|
| **Mask R-CNN** | bbox | | | segm | | |
| MS COCO—implementation [37] | 4.14 | 13.39 | 1.4 | 5.36 | 14.44 | 1.05 |
| MS COCO—implementation [36] | 7.62 | 13.5 | 7.62 | - | - | - |
| Citiscapes | 0.06 | 0.3 | 0 | 0.01 | 0.09 | 0 |
| LVIS | 0.71 | 3.45 | 0.04 | 0 | 0 | 0 |
| **TensorMask** | | | | | | |
| MS COCO—implementation [37] | 1.69 | 3.41 | 1.29 | 3.06 | 6.31 | 2.6 |

The results of 3-fold cross-validation for the parameters of four batch size with 2000 epochs and two batch size with 100,000 epochs for pre-trained and trained from random initialization models respectively are presented in Table 7. The results are presented in the form of box and segmentation

mAP. The folds of model training for three different training and testing datasets were repeated twice. Presented result values are the best selected repetition.

**Table 7.** 3-fold cross-validation mean Average Precision (mAP) results for pre-trained (four batch, 2000 epochs) and trained from scratch (two batch, 100,000 epochs) models of Mask R-CNN and TensorMask.

| | **Mask Pre-Trained** | | **Mask From Scratch** | | **TensorMask Pre-Trained** | | **TensorMask From Scratch** | |
|---|---|---|---|---|---|---|---|---|
| **mAP** | bbox | segm | bbox | segm | bbox | segm | bbox | segm |
| **orig** | | | | | | | | |
| 1. fold | 88.8 | 89.1 | 83.6 | 85.9 | 90.9 | 90.2 | 78.2 | 79.6 |
| 2. fold | **92.0** | 89.5 | 84.7 | 87.6 | 91.9 | **90.5** | 73.9 | 70.6 |
| 3. fold | 89.7 | 89.2 | 87.7 | 88.0 | 90.5 | 90.2 | 70.6 | 76.6 |
| **average** | 90.2 | 89.3 | ***85.3*** | ***87.2*** | **91.1** | **90.3** | 74.2 | 75.6 |
| **ch1** | | | | | | | | |
| 1. fold | 90.2 | 89.7 | 84.6 | 87.2 | 90.3 | 90.2 | 78.9 | 77.7 |
| 2. fold | 88.4 | 89.1 | 88.3 | 88.9 | **91.3** | **90.4** | 78.7 | 77.8 |
| 3. fold | 89.6 | 89.4 | 88.3 | 89.0 | 90.2 | 89.8 | 75 | 74.4 |
| **average** | 89.4 | 89.4 | ***87.1*** | ***88.4*** | **90.6** | **90.1** | 77.5 | 76.6 |
| **ch2** | | | | | | | | |
| 1. fold | 87.9 | 88.9 | 82.6 | 82.4 | 87.9 | **89.3** | 64.4 | 62.3 |
| 2. fold | 84.7 | 86.4 | 86.6 | 88.2 | 89.1 | 89.0 | 66 | 57.2 |
| 3. fold | **89.5** | 88.1 | 85.0 | 87.7 | 88.3 | 88.9 | 65.7 | 54.3 |
| **average** | 87.4 | 87.8 | ***84.7*** | ***86.1*** | **88.4** | **89.1** | 65.4 | 57.9 |
| **ch3** | | | | | | | | |
| 1. fold | 73.9 | **65.7** | 66.4 | 56.1 | **75.0** | 64.8 | 44.6 | 13.1 |
| 2. fold | 63.0 | 51.9 | 65.6 | 57 | 70.9 | 61.9 | 40.2 | 11.2 |
| 3. fold | 72.6 | 62.9 | 66.6 | 62.4 | 71.3 | 62.5 | 36.6 | 11.2 |
| **average** | 69.8 | 60.2 | ***66.2*** | ***58.5*** | **72.4** | **63.1** | 40.5 | 11.8 |
| **16-bit** | | | | | | | | |
| 1. fold | 90.5 | 89.4 | 84.9 | 85.9 | 91.6 | 90.2 | 78.1 | 80.2 |
| 2. fold | 89.2 | 89.4 | 85.8 | 88.5 | **92.5** | **90.8** | 76.8 | 77.6 |
| 3. fold | 90.0 | 89.1 | 86.6 | 87.8 | 90.6 | 89.9 | 71.5 | 73.9 |
| **average** | 89.9 | 89.3 | ***85.8*** | ***87.4*** | **91.6** | **90.3** | 75.5 | 77.2 |

## 4. Discussion

The objective of the study was to compare two different instance segmentation approaches and two different learning methods for the purpose of experimental animals segmentation on thermal images. Various methods for re-scaling thermal data to a standard image were also compared.

In paper [38] authors showed that the model trained from random initialization, if it only has proper training parameters, can get results similar to the pre-trained models, however, it needs more iterations to converge. In our experiments training Mask R-CNN from scratch needed about 16 times the number of epochs (16,000 or 24,000) (Tables 4 and 5) than for the pre-trained model (1000 or 2000) to achieve similar results (Tables 2 and 3). Increasing the number of epochs to 100,000 improved the results only for the orig and ch1 images, but not significantly. TensorMask architecture required a much larger number of epochs for the random initialization training (100,000), and still did not achieve results comparable to the pre-trained model (see Tables 4 and 5).

Mask R-CNN training from scratch neither is time nor data expensive. Only a slightly larger number of images and epochs allow it to achieve all-layers training results comparable to the pre-trained models.

The values in Tables 2 and 4 indicate that the best segmentation results for the pre-trained models can be obtained for the 16-bit and ch1 image followed by the orig for Mask R-CNN and also 16-bit with orig this time followed by ch1 using TensorMask. Ch2 images achieved only slightly lower results than the top ones. The bbox and segmentation mAP values for both architectures were close to 90 percent, with a slight TensorMask advantage. Detection for ch3 in both cases achieved the best mAP above 70, while the segmentation best mAP was equal to 65 percent. If we look at the results for all combinations of training parameters, not only the best ones, it can be clearly seen that TensorMask achieves better results for various training models.

The results of training from scratch show the opposite trend (see Tables 3 and 5). This time it is Mask R-CNN that achieves better results for all images (mAP above 80 percent for almost all images) under various criteria, suggesting that this model trained from scratch catch up not only by chance for a single metric. The differences were especially visible for the ch3 image, for which the best TensorMask segmentation mAP was equal only to 32.752 percent.

The pre-trained model shows generally similar results for instance segmentation and detection for all images except ch3. The image ch3 contains a very narrow range of only the highest animals' surface thermal values (Figure 2a,e), so it is deprived of a significant part of the body area. Such an object is detectable, but difficult to correctly segment if there is no information about the object's boundaries. That is why the mAP for segmentation is much smaller than for detection.

The data in Tables 3 and 5 show that models trained from scratch are likely to perform segmentation more accurately than the detection (except ch3). Detection and segmentation accuracy for the pre-trained models are more similar.

In the Figure 3 we have presented the results of the detection and segmentation for challenging views with the critical regions zoomed in white frames. The image in Figure 3c is difficult to segment, because the snout of one rodent covers the snout of the other one over the entire width of the body. However, there is a small element of the snout that belongs to the animal at the bottom and in this camera view is not connected to the rest of the body. One of animals in Figure 3d has its snout hidden under the body of the other rat, which, as a matter of fact, is not so rare. Here, the cooler water mark left on the fur creates a line, that can be mistaken as a continuation of the body boundary. In this particular way it was segmented by the pre-trained TensorMask (Figure 3h), as a result of which a small area of the body on the border was miss-assigned to the wrong object. It also marked both individuals as the object, however with low probability (7%). The Mask R-CNN architecture set the boundaries more precisely (Figure 3f,j) but left a few pixels' gap between the objects. TensorMask trained from scratch segmented the rodents' body areas similarly to the pre-trained model, (Figure 3l), and here also an additional object—a combination of both body parts—was detected.

The separated object from the Figure 3c caused the Mask R-CNN the most problems. The pre-trained model (Figure 3e) considered the separated part of the animal's mouth as an element of the other rat's body. In addition, as the only one, it inaccurately determined the bbox of the detected object. Mask R-CNN trained from scratch did not assign this small part of the snout to any object, but correctly detected the bbox (Figure 3i). The pre-trained TensorMask (Figure 3g) classified the problematic snout partly to both individuals—this line of segmentation seems to be the closest to the correct one. TensorMask architecture trained from scratch assigned both animal snouts to both objects at once (Figure 3k).

Although the segmentation results were not always satisfactory, it should be remembered that difficult cases were presented here. For the vast majority of images, the prediction was very similar to ground-truth images and succeeded in its mission of rodent segmentation, in contrast to the semantic segmentation algorithms presented in paper [16]. The results of U-Net segmentation (see Figure 1d) show that the boundary between objects disappears during close contact, although it was previously visible during small connection.

The results of segmentation made by the commonly available models (see Table 6) are much worse than those trained on the target images, and do not exceed 8 mAP. The Citiscapes and LVIS

datasets achieved extremely low values—bellow 0.75 mAP. During the evaluation, the correctness of class assignment is also taken into account. Although the COCO data-set has a "mouse" class [20], however, rat objects were not assigned to it, so in this case the assignment correctness was zero.

The 3-fold cross-validation results presented in Table 7 are very similar. The top segmentation and detection results (marked as bold in Table 7) on average was achieved by the pre-trained TensorMask. Both pre-trained models show greater accuracy than those trained from scratch, which is consistent with the previous results (see Tables 2–5). It is probably related to a much smaller training set comparing to the COCO set [20]. As far as a random initialized models are concerned, the difference between the both models is significant. Mask R-CNN (marked as bold italic in Table 7) obtains mAP even 10 percent higher than TensorMask. It is possible that TensorMask needs more data and/or epochs number to achieve a Mask R-CNN-like results.

## 5. Conclusions

The deep instance segmentation algorithms are able to distinguish between two individuals in close contact where overlaps may appear. The segmentation mAP almost reach value of 90 percent. The detection results are slightly higher and usually oscillate around 90 percent. The top results are achieved by the single-stage (TensorMask) pre-trained network, however, two-stage method (Mask R-CNN) works better than single-stage when trained from scratch. Training Mask R-CNN model from random initialization is neither time nor data consuming. It can be used for training non-standard images, however, keeping in mind that networks trained from scratch focuses more on segmentation than detection. In turn, the cost of training TensorMask is large, and still does not achieve results similar to the pre-trained version. The pre-trained TensorMask model shows the best performance. The research indicates that thermal images can be successfully analyzed by architectures pre-trained on standard images. However, some layers of the network must be trained on thermal images, because the models pre-trained only on publicly available databases achieve very poor segmentation results and even worse in detection.

The thermal data conversion from different thermal data range does not improve the quality of segmentation. The results for 16-bit (*16-bit*) and 8-bit (orig) image representation as well as for image deprived of the background (ch1) are comparable. Results of images with narrower thermal range (ch2) do not differ much from the others. Unlike the results for ch3 image, where only the information about the warmest parts of the body is visible. However, the bounding box mAP values of this images for pre-trained model suggest that detection for limited-data images is possible.

The main contributions of this paper are the following:

- the adopted deep instance segmentation algorithms have been experimentally verified for the laboratory rodents detection from thermal images,
- it was shown that laboratory rodents can be accurately detected (and separated from each other) from the thermal images using the Mask R-CNN and TensorMask models,
- the obtained results demonstrated that the adopted TensorMask model, pre-trained using visible light images and trained with thermal sequences gave the best results with the mean average precision (mAP) greater that 90,
- it was verified that thermal data conversion from narrower raw thermal range does not improve the quality of segmentation,
- single-stage pre-trained networks achieves better results than two-stage pre-trained models, however two-stage methods seem to work better than single-stage when trained from scratch,
- network pre-training using visible light images improves the segmentation results for thermal images.

The conclusion of this work is that segmentation algorithms can be used to segment experimental animals in thermal images. Depending on the needs, one can customize architectures, learning methods or image types for the best performance. Instance segmentation methods work better than the

semantic segmentation methods. The presented approach will work well in social behaviour tests, but not only that. It can be used wherever identification and tracking of experimental animals is required, especially in numerous groups.

In the future it is worth training both architectures with a more diverse thermal database. Increasing the amount of training data can also improve the results, especially for the TensorMask.

**Author Contributions:** Conceptualization, M.M.-M. and J.R.; data curation, M.M.-M.; formal analysis, M.M.-M. and J.R.; funding acquisition, J.R.; investigation, M.M.-M.; methodology, M.M.-M. and J.R.; project administration, M.M.-M. and J.R.; resources, M.M.-M.; software, M.M.-M. and T.K.; supervision, J.R.; validation, M.M.-M., J.R. and T.K.; visualization, M.M.-M.; writing—original draft preparation, M.M.-M.; writing—review and editing, J.R. and T.K. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work has been partially supported by Statutory Funds of Electronics, Telecommunications and Informatics Faculty, Gdansk University of Technology

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Lezak, K.; Missig, G.; Carlezon, W.A., Jr. Behavioral methods to study anxiety in rodents. *Dialogues Clin. Neurosci.* **2017**, *19*, 181–191. [PubMed]
2. Franco, N.H.; Gerós, A.; Oliveira, L.; Olsson, I.A.S.; Aguiar, P. ThermoLabAnimal—A high-throughput analysis software for non-invasive thermal assessment of laboratory mice. *Physiol. Behav.* **2019**, *207*, 113–121. [CrossRef] [PubMed]
3. Junior, C.F.C.; Pederiva, C.N.; Bose, R.C.; Garcia, V.A.; Lino-de-Oliveira, C.; Marino-Neto, J. ETHOWATCHER: Validation of a tool for behavioral and video-tracking analysis in laboratory animals. *Comput. Biol. Med.* **2012**, *42*, 257–264. [CrossRef] [PubMed]
4. Grant, E.; Mackintosh, J. A comparison of the social postures of some common laboratory rodents. *Behaviour* **1963**, *21*, 246–259.
5. Kask, A.; Nguyen, H.P.; Pabst, R.; von Hoorsten, S. Factors influencing behavior of group-housed male rats in the social interaction test—Focus on cohort removal. *Physiol. Behav.* **2001**, *74*, 277–282. [CrossRef]
6. Aslani, S.; Harb, M.; Costa, P.; Almeida, O.; Sousa, N.; Palha, J. Day and night: diurnal phase influences the response to chronic mild stress. *Front. Behav. Neurosci.* **2014**, *8*, 82. [CrossRef]
7. Roedel, A.; Storch, C.; Holsboer, F.; Ohl, F. Effects of light or dark phase testing on behavioural and cognitive performance in DBA mice. *Lab. Anim.* **2006**, *40*, 371–381. [CrossRef]
8. Manzano-Szalai, K.; Pali-Schöll, I.; Krishnamurthy, D.; Stremnitzer, C.; Flaschberger, I.; Jensen-Jarolim, E. Anaphylaxis Imaging: Non-Invasive Measurement of Surface Body Temperature and Physical Activity in Small Animals. *PLoS ONE* **2016**, *11*, e0150819. [CrossRef]
9. Etehadtavakol, M.; Emrani, Z.; Ng, E.Y.K. Rapid extraction of the hottest or coldest regions of medical thermographic images. *Med. Biol. Eng. Comput.* **2019**, *57*, 379–388. [CrossRef]
10. Jang, E.; Park, B.; Park, M.; Kim, S.; Sohn, J. Analysis of physiological signals for recognition of boredom, pain, and surprise emotions. *J. Phys. Anthropol.* **2015**, *34*, 1–12. [CrossRef]
11. Tan, C.; Knight, Z. Regulation of Body Temperature by the Nervous System. *Neuron* **2018**, *98*, 31–48. [CrossRef] [PubMed]
12. Sona, D.; Zanotto, M.; Papaleo, F.; Murino, V. Automated Discovery of Behavioural Patterns in Rodents. In Proceedings of the 9th International Conference on Methods and Techniques in Behavioral Research, Wageningen, The Netherlands, 27–29 August 2014.
13. Koniar, D.; Hargaš, L.; Loncová, Z.; Simonová, A.; Duchoň, F.; Beňo, P. Visual system-based object tracking using image segmentation for biomedical applications. *Electr. Eng.* **2017**, *99*, 1349–1366. [CrossRef]
14. Fleuret, J.; Ouellet, V.; Moura-Rocha, L.; Charbonneau, É.; Saucier, L.; Faucitano, L.; Maldague, X. A Real Time Animal Detection And Segmentation Algorithm For IRT Images In Indoor Environments. *Quant. InfraRed Thermogr.* **2016**, 265–274.
15. Kim, W.; Cho, Y.B.; Lee, S. Thermal Sensor-Based Multiple Object Tracking for Intelligent Livestock Breeding. *IEEE Access* **2017**, *5*, 27453–27463. [CrossRef]
16. Mazur-Milecka, M.; Ruminski, J. Deep learning based thermal image segmentation for laboratory animals tracking. *Quant. InfraRed Thermogr. J.* **2020**, 1–18. [CrossRef]

17. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*; Springer: Cham, Switzerland, 2015; pp. 234–241.
18. Milletari, F.; Navab, N.; Ahmadi, S. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. In Proceedings of the IEEE 2016 Fourth International Conference on 3D Vision (3DV), Stanford, CA, USA, 25–28 October 2016; pp. 565–571.
19. Hariharan, B.; Arbelaez, P.; Girshick, R.B.; Malik, J. Simultaneous Detection and Segmentation. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2014; pp. 297–312.
20. Lin, T.; Maire, M.; Belongie, S.J.; Bourdev, L.D.; Girshick, R.B.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2014; pp. 740–755.
21. Dai, J.; He, K.; Sun, J. Instance-aware Semantic Segmentation via Multi-task Network Cascades. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3150–3158.
22. Chen, L.; Hermans, A.; Papandreou, G.; Schroff, F.; Wang, P.; Adam, H. MaskLab: Instance Segmentation by Refining Object Detection with Semantic and Direction Features. *arXiv* **2017**, arXiv:1712.04837.
23. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R.B. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
24. Huang, Z.; Huang, L.; Gong, Y.; Huang, C.; Wang, X. Mask Scoring R-CNN. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 6409–6418.
25. Girshick, R.B. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
26. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv* **2015**, arXiv:1506.01497.
27. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
28. Chen, K.; Pang, J.; Wang, J.; Xiong, Y.; Li, X.; Sun, S.; Feng, W.; Liu, Z.; Shi, J.; Ouyang, W.; et al. Hybrid Task Cascade for Instance Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 4974–4983.
29. Yao, J.; Yu, Z.; Yu, J.; Tao, D. Single Pixel Reconstruction for One-stage Instance Segmentation. *arXiv* **2019**, arXiv:1904.07426.
30. Bolya, D.; Zhou, C.; Xiao, F.; Lee, Y.J. YOLACT: Real-time Instance Segmentation. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 9157–9166.
31. Xiang, C.; Tian, S.; Zou, W.; Xu, C. SAIS: Single-stage Anchor-free Instance Segmentation. *arXiv* **2019**, arXiv:cs.CV/1912.01176.
32. Ying, H.; Huang, Z.; Liu, S.; Shao, T.; Zhou, K. EmbedMask: Embedding Coupling for One-stage Instance Segmentation. *arXiv* **2019**, arXiv:cs.CV/1912.01954.
33. Tian, Z.; Shen, C.; Chen, H.; He, T. FCOS: Fully Convolutional One-Stage Object Detection. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 9627–9636.
34. Chen, X.; Girshick, R.B.; He, K.; Dollár, P. TensorMask: A Foundation for Dense Object Segmentation. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 2061–2069.
35. Mazur-Milecka, M.; Ruminski, J. Automatic analysis of the aggressive behavior of laboratory animals using thermal video processing. In Proceedings of the IEEE Conference of the Engineering in Medicine and Biology Society, EMBC, Seogwipo, Korea, 11–15 July 2017; pp. 3827–3830.
36. Abdulla, W. Mask R-CNN for Object Detection and Instance Segmentation on Keras and TensorFlow. 2017. Available online: https://github.com/matterport/Mask_RCNN (accessed on 1 February 2020).
37. Wu, Y.; Kirillov, A.; Massa, F.; Lo, W.Y.; Girshick, R. Detectron2, 2019. Available online: https://github.com/facebookresearch/detectron2 (accessed on 1 May 2020).

38. He, K.; Girshick, R.; Dollar, P. Rethinking ImageNet Pre-Training. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019; pp. 4917–4926.
39. Wu, Y.; He, K. Group Normalization. *Int. J. Comput. Vis.* **2019**. [CrossRef]
40. Dutta, A.; Zisserman, A. The VIA Annotation Software for Images, Audio and Video. In Proceedings of the 27th ACM International Conference on Multimedia, Nice, France, 21–25 October 2019; ACM: New York, NY, USA, 2019. [CrossRef]
41. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 3213–3223.
42. Gupta, A.; Dollár, P.; Girshick, R. LVIS: A Dataset for Large Vocabulary Instance Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 5356–5364.
43. Lu, Y.; Lu, C.; Tang, C. Online Video Object Detection Using Association LSTM. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2363–2371.
44. Oksuz, K.; Cam, B.C.; Akbas, E.; Kalkan, S. Localization Recall Precision (LRP): A New Performance Metric for Object Detection. In *Computer Vision—ECCV 2018*; Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 521–537.