scientific reports

OPEN

Generation of microbial colonies dataset with deep learning style transfer

Jarosław Pawłowski[1,2]✉, Sylwia Majchrowska[1,2] & Tomasz Golan[1]

We introduce an effective strategy to generate an annotated synthetic dataset of microbiological images of Petri dishes that can be used to train deep learning models in a fully supervised fashion. The developed generator employs traditional computer vision algorithms together with a neural style transfer method for data augmentation. We show that the method is able to synthesize a dataset of realistic looking images that can be used to train a neural network model capable of localising, segmenting, and classifying five different microbial species. Our method requires significantly fewer resources to obtain a useful dataset than collecting and labeling a whole large set of real images with annotations. We show that starting with only 100 real images, we can generate data to train a detector that achieves comparable results (detection mAP = 0.416, and counting MAE = 4.49) to the same detector but trained on a real, several dozen times bigger dataset (mAP = 0.520, MAE = 4.31), containing over 7 k images. We prove the usefulness of the method in microbe detection and segmentation, but we expect that it is general and flexible and can also be applicable in other domains of science and industry to detect various objects.

Deep learning (DL) has recently achieved outstanding results in several disciplines, including medicine, microbiology, and bioinformatics. DL-based automatic extraction of information from images has led to unprecedented progress in many fields. Neural networks were applied in numerous medical tasks, most extensively in radiology and pathology specialties[1–8]. Moreover, their performance is frequently comparable, or even better than in the case of human experts. Furthermore, it is possible that DL algorithms could be used to extract data that would not be apparent in human analysis[9]. The use of neural networks also makes it possible to automate industrial processes such as counting microbial colonies on Petri dishes, which is an important step in the microbiological laboratory to evaluate the cleanliness of the samples. Traditionally, the counting task is done manually or semi-automatically[10,11], but recent studies suggest that DL-based methodology will accelerate the process[12–16].

To evaluate the number of microbial colonies grown on a Petri dish, neural network models called detectors may be applied[12]. The most representative group of detectors are two-stage models. These architectures are characterised by high localization and classification accuracy. The concept is to separately resolve the localization and classification tasks in two stages. During the first stage, regions of interest are generated, and then in the second stage, a bounding box regression is made, and the achieved region proposals are assigned to the appropriate class with some level of confidence[17]. The baseline two-stage architecture is Faster R-CNN[18]. The Mask R-CNN[19] extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. However, object instance segmentation needs annotation at a pixel level, which makes the labeling process even more challenging. Estimation of the number of microbial colonies may also be based on density map estimation[14], similarly to the case of human crowd counting[20]. In this approach, the annotations for each object are in the form of a density map. Algorithms based on a Convolutional Neural Network (CNN) learn the mapping between the extracted features and their object density maps, to indirectly count objects by integration of the predicted map. In turn, in this type of research UNet[21]-based models are used.

However, despite the broad possibilities and the recent interest in the topic[12–14], the application of DL-based methods in medical, pharmaceutical, or food industries still faces some challenges. DL algorithms are quite efficient when learning from large amounts of data, whose collection and annotations require a lot of time and financial support. Moreover, the labelling process itself can be challenging. In our case, the problem of identifying a separate colony could be hard even for trained professionals, because some microbial colonies tend to agglomerate and overlap, thus becoming indistinguishable. On top of that, the identified colony can be atypical

[1]NeuroSYS, Rybacka 7, 53-656 Wrocław, Poland. [2]Faculty of Fundamental Problems of Technology, Wroclaw University of Science and Technology, Wybrzeże S. Wyspiańskiego 27, 50-372 Wrocław, Poland. ✉email: j.pawlowski@neurosys.com

or deformed, and the image itself is influenced by various uncontrolled natural conditions. Object diversity requires well-annotated data whose distribution should match the target industrial use case.

Related works. DL can also resolve this issue by using techniques to generate synthetic data confusingly similar to real samples. The idea is to train a model on a synthetically generated dataset with the intention of *transfer learning* to real data. The most popular models to generate synthetic data are Generative Adversarial Networks (GANs)[22], which are composed of two neural networks—a generator to generate new samples, and a discriminator to recognize fake samples from the real one. Moreover, a CycleGAN[23] composed of two GANs, or conditional GAN[24], is useful in the case of image-to-image translation.

Generative neural models are becoming more widely used in medicine[25,26]. GANs are used for generation of high quality radiology images, e.g. mammograms for radiology education[27], or realistic chest X-ray images[28]. In dermatology, for automated skin cancer classification[29], or to build a general skin condition classifier[30]. In ophthalmology, for eye's retinal image synthesis[31,32]. Image-to-image translation can be also applied in dermatology for melanoma detection[33], or to translate magnetic resonance images from computed tomography ones[34]. On the other hand, there is notable work in microbiology, related to our, on generating a dataset of blood plates on agar using GANs, then used for training a neural network model for semantic segmentation of bacterial colonies[35]. However, GANs, like other deep learning models, need a lot of real data for the training process.

It is also worth noting that there are very few useful datasets containing images of bacterial colonies that can be used to train DL models. The biggest ones are MicrobIA[36], AGAR[12], and DIBaS[37]. Collecting a large dataset is a great effort, not only to gather the images but also to label them. Therefore, creating synthetic datasets is a good alternative.

Synthetic microbial colonies dataset. In our study, we develop another strategy to generate annotated synthetic datasets without the need for a lot of input resources. We propose a method combining traditional computer vision and DL-based style transfer concepts for generating synthetic images of Petri dishes. We utilise the neural style transfer technique to change the style of an image in one domain (synthetic) to the style of an image in another domain (real), and thus improve the realness of the generated data. Photorealistic image stylization does not require much resources and typically uses a single style image during the stylization process[38]. The idea was introduced by Gatys[39] and further developed by Fei-Fei and others[40–43].

We used the *higher-resolution* subset of the recently introduced AGAR microbial colony dataset[12] to conduct a synthetic dataset generation experiment. We randomly selected 100 annotated images from this subset and used them to feed the colony extraction part. Additionally, 10 empty dishes were taken to serve as a background on which colonies were deposited. Using the extracted colonies, we generated a big synthetic dataset of Petri dishes augmented using style transfer with 20 different styles. For diverse styles, we used 20 fragments with different lighting conditions, selected from these 100 input images. Each generated image contains a selected number of colonies (corresponding to real data to be mimicked) and is of size 512×512 pixels, which corresponds to the size of fragments (patches) used for training colony detectors when evaluating AGAR dataset[12]. To cover the whole dish we would need approximately 8×8 patches which gives a similar resolution for the whole dish image as in the AGAR samples with 4000×4000 pixels (*higher-resolution* subset). Note, however, that during the generation process we synthesize individual patches, not the entire dish at once, but after simply adjusting the method, we could also generate the whole dish. Finally, a deep learning detector was trained entirely on this synthetic dataset and evaluated on the test part of the *higher-resolution* AGAR subset. This way, the obtained results are comparable with the ones from[12], obtained for the same DL detector architecture, but trained entirely on the real dataset, i.e. training part of the *higher-resolution* subset containing 7360 images.

Methods

In this section we present a detailed description of a method for generation of synthetic images with microbial colonies. Then we provide an overview of the neural network models used to: make the generated images more realistic, and to implement the colony detector. Our source code with the python implementation of the generation framework is publicly available at[44]. Let us start with a general scheme of the introduced method. In Fig. 1, our strategy to generate a synthetic dataset is presented. The method consists of three subsequent steps. We start with labeled real images of Petri dishes and perform colony segmentation using traditional computer vision algorithms including proper filtering, thresholding, and energy-based segmentation. To get a balanced working dataset, we randomly select 20 images for each of the 5 microbial species (giving 100 images in total) from the *higher-resolution* subset of the AGAR dataset[12].

In the second step, segmented colonies and clusters of colonies are randomly arranged on fragments of an empty Petri dish (we call them patches). We select a random fragment of one of the 10 images of a real empty dish. We repeat this step many times, placing subsequent clusters in random places making sure they do not overlap. Simultaneously, we store the position of each colony from the cluster placed on the patch and its segmentation mask, creating a dictionary of annotations for that patch.

Finally, in the third step, we apply the augmentation method via deep learning style transfer. We transfer the style of a given raw patch to one of the selected real images that serve as style carriers. We select 20 real fragments with very different lighting conditions serving as style carriers to increase the diversity of the generated patches.

Colony segmentation. At first we read the annotations—each stored as 4 numbers (x, y, *width*, *height*) defining a *bounding box* that represents a single labelled colony on a given image. Using this information, we check which colonies overlap by more than 0.01 part of its area and, this way, we build an *adjacency matrix* between colonies. This step is important because colonies naturally overlap in highly populated dishes. Conse-

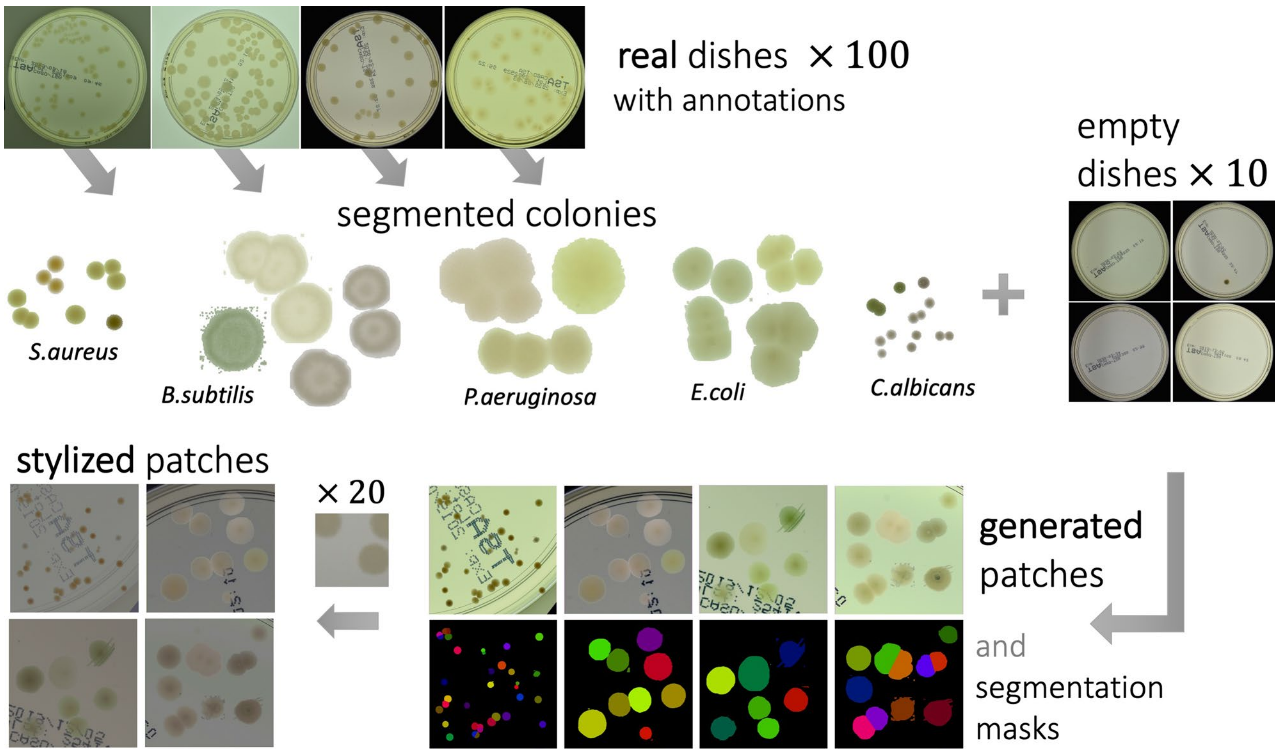

Figure 1. Scheme for synthetic microbial dataset generation. Microbial colonies are segmented from real images using traditional computer vision algorithms, and then randomly arranged on fragments of an empty dish giving synthetic patches with precise annotations. To improve the realism of generated data, patches are then stylized using a neural style transfer method.

quently, to capture the geometrical details of their mutual shapes, we want to segment whole clusters of overlapping colonies—not just individual ones. Using the adjacency matrix and breadth-first search algorithm for the graph (representing connections between colonies), we obtain connected clusters of bacterial colonies (clearly, some clusters contain a single colony) that we segment in the next step.

In this step, we cut off the rectangular fragment of the image which contains a single cluster (and repeat this step for every cluster), and add a binary *alpha channel* m_{bx} with zero values on areas not covered by any of the bounding boxes from the cluster. We call it boxes' mask, and obviously m_{bx} has the same size as the rectangular fragment. We are now ready to perform a segmentation procedure to remove the colony background and—consequently—refine the alpha channel for that fragment. Examples of colonies with removed background ready to be arranged on an empty dish are presented in Fig. 1.

The segmentation procedure consists of multiple substeps using computer vision methods. Such methods have already been used to detect and segment microbial objects[10,11,45–47]. The first step consists of filtering out unwanted artifacts. At first, we apply the unsharp mask filter and convert the given fragment to the CIELab colorspace. Secondly, we implement simple removal of dark objects (i.e., contamination and characteristic text labels across the dish substrate): they are detected via the *luminance* and *b*-value thresholding[10] in the CIELab colorspace. Then, we dilate the mask m_d for better coverage of artifacts (dark regions). Finally, each pixel in the detected dark region is replaced with the nearest valid (i.e., not belonging to the dark region mask m_d) pixel found by a random walk algorithm. An example of the method operation is shown in Fig. 2. Such an approach successfully removes the above-mentioned artifacts, but unfortunately random walk introduces a lot of speckle noise to a given fragment. Therefore, in the last step we apply denoising using non-local means[48] to remove such speckle noise.

Now we are ready to perform colony cluster segmentation. We use the powerful Chan–Vese algorithm[49,50] based on a signal energy minimization provided by the scikit-image library[51]. Then add some margins to the resulting segmentation mask m_s to be sure that we segment whole colonies together with their edges—see examples in Fig. 3. To get better future blending with the empty dish background, we create an additional mask m_b with values for each pixel (in a range of $[0 : 255]$) proportional to its distance from average background color (i.e. outside the segmentation mask m_s) in the CIELab colorspace. Such a mask has lower values in areas where the pixel color is similar to the patch background average color. The idea in this step is to increase opacity (lower alpha channel) in areas that are similar to the background and contain a lower amount of interesting information, i.e., we assume that colonies are different from their background. This step can significantly improve the realism of the visual look of the generated patches. Next, this blending mask multiplies the segmentation and box masks, and using Hadamard product of $m_{bx} \circ m_s \circ m_b$, we obtain the final alpha channel. Finally, we use it to remove the colony background.

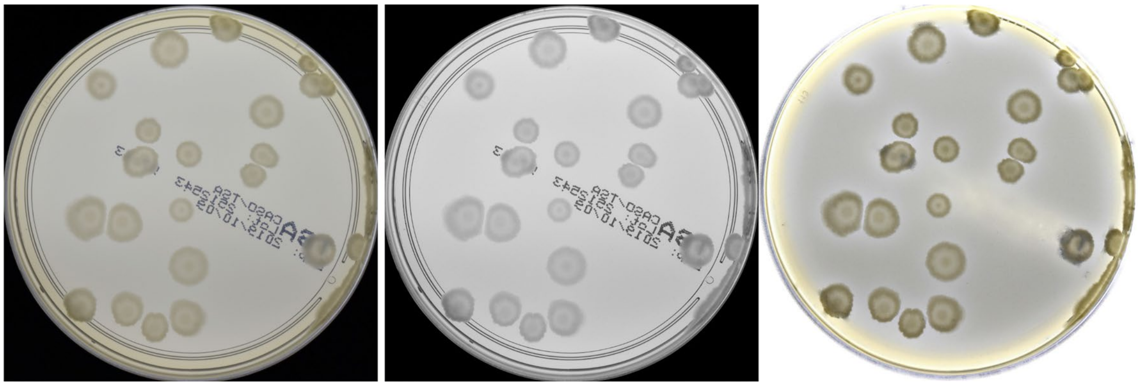

Figure 2. Removal of artifacts detected by thresholding in the CIELab colorspace (*luminance* channel presented in the middle), and replaced by a nearest valid pixel found by a random walk.

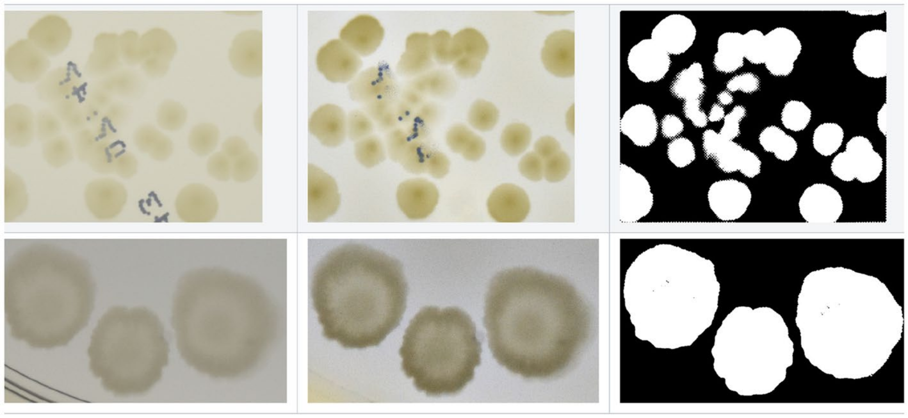

Figure 3. Example patch segmentation (binary masks m_s on the right) using the Chan–Vese algorithm.

Patch generation. After extracting numerous clusters of colonies for every 5 types of microbes, we are ready to generate a synthetic dataset. To generate a single patch, we select one of the 10 empty dishes, randomly rotate it, and take its 512×512 sized patch. Moreover, one of the microbe types is selected. Then, we take a random colony cluster of that type, rotate and flip it randomly, and put it in a random place on the patch. The position of each placed colony (i.e., bounding box) and its segmentation mask is stored. Next, we repeat this step many times placing subsequent clusters in random places but, at the same time, making sure they do not overlap. We also select the number of colonies placed on the patch using the exponential probability distribution $\lambda e^{-\lambda x}$, with $1/\lambda = 10$ mean. This corresponds to the distribution of the number of colonies on the patch in AGAR dataset. Note, however, that in case of larger colonies typically fewer colonies will fall into a single patch. We did not observe such phenomena in our experiments, but in some cases, to avoid biasing different classes by the size of colonies, it may be also helpful to provide an additional augmentation technique by varying (scaling) the size of placed colonies during the generation. We repeat the whole patch-generating process and get a dataset of 50k patches. All subsequent steps of the patch generation method together with the number of parameters that need to be tuned are listed in Table 1.

Neural style transfer. Having described how to generate raw patches, we move on to neural style transfer. We utilize the stylization method described in[38] with architecture similar to this used in seminal work[40] but with HRNet[52] high-resolution representation network instead of a standard convolutional encoder-decoder architecture. We chose this approach after several trials because it gives the most realistic stylization of our raw Petri dish images without introducing unwanted artifacts, which was the case for other tested methods, e.g, the original one introduced in[39]. We base our style transfer implementation on the code repository provided in[38]. The architecture consists of two deep networks: an image generation network—HRNet and pretrained VGG19[53] used to calculate content and style reconstruction losses. Deep convolutional neural network VGG19 is used because we are not interested in exactly matching the pixels of output y and style y_s or content y_c images; instead we force them to have similar feature representations and compare the VGG-activations between output and style or content. The resulting feature maps are combined via Gram matrix[39] G, and the weighted differences

$$\mathcal{L}(y, y_c, y_s) = (1-\lambda)\|G(y) - G(y_c)\| + \lambda\|G(y) - G(y_s)\| \tag{1}$$

Subsequent steps	No of parameters to be tuned
Colonies clustering	1
Unsharp mask filtering	2
Removal of artifacts	
Thresholding in CIELab space	2
Mask dilation	1
Speckle noise cancellation	2
Chan–Vese segmentation	2
Mask dilation	1
Blending mask	1
Neural style transfer	1
Total	13

Table 1. Number of parameters (that take non-default values) for each step in the patch generation method.

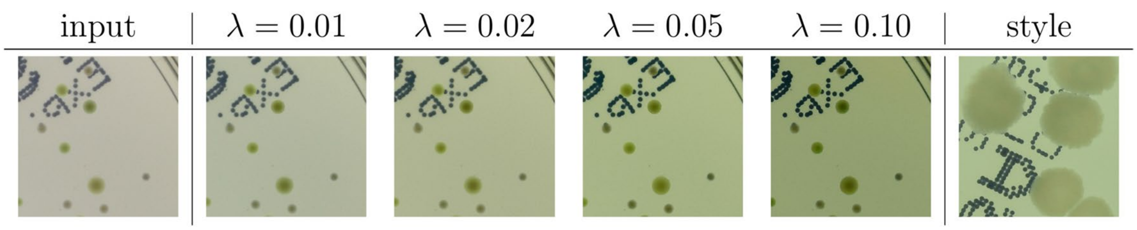

Figure 4. Image style transfer for different stylization weights. Style (right) is transferred to input content image (left) with different stylization amount (middle) tuned by λ parameter.

give the loss function $\mathscr{L}$ with λ-weight. Increasing λ introduces more style to the content image as presented in Fig. 4. The image generation network is trained using Stochastic Gradient Descent to minimize $\mathscr{L}$.

Stylization is the longest step in the generation method. To speed up the stylization procedure, a bigger image of size 1024×1024 containing 4 patches was stylized using a single style at once. Training such a one-shot stylization model lasts about 30 seconds. Moreover, in the subsequent stylization iterations, the model initial weights were taken from the previous ones, and this considerably speeds up the convergence of the stylization training.

Neural object detector. We trained two widely used deep learning object detectors from R-CNN family: Faster[18] and Cascade[54] using synthetic data, and show the usefulness of the generation method by testing the performance of microbe detection on *real* images of Petri dishes—the test part of the *higher-resolution* subset of AGAR dataset. For detecting and counting microbial colonies, we used Faster R-CNN with ResNet-50[55] as a backbone, and Cascade R-CNN with HRNet[52] backbone. The same architectures were used as the baseline (Faster R-CNN) and a more advanced model (Cascade R-CNN) during experiments with the real AGAR dataset[12], which makes them good networks to compare performance. Moreover, the Faster detector was extended to Mask R-CNN in case of instance segmentation experiments. Both used detectors are the top deep learning models with quite complex structure, e.g. Cascade R-CNN model with HRNet has got 956 different layers in total.

We trained the networks for 20 epoches, with a batch size of 8 (for Faster) or 3 (for Cascade), and an adaptive learning rate initialized to 0.0001. During training, Stochastic Gradient Descent (SGD) method was used for optimizing the network weights. Calculations were performed on NVIDIA Titan RTX GPU, and the training process lasted about 2 (Faster) to 4 (Cascade) days depending on the neural network model used. We used the detectors implementation from the MMDetection[56] toolbox. Backbones' weights were initialized using models pre-trained on ImageNet[57], available in the *torchvision* package of the PyTorch library.

Results

In this section we present results for the generation of synthetic images with microbial colonies, and the results for training deep learning detectors using the generated data.

Generation and stylization of synthetic dataset. In the first step, we take 100 annotated real images of Petri dishes (20 for each microbe type in AGAR dataset) and perform colony segmentation using traditional computer vision algorithms. In the second step, segmented colonies and clusters of colonies are randomly arranged on fragments of an empty Petri dish. Examples of generated patches together with their masks denoting different colonies are shown in Fig. 1. Further examples of generated patches with different microbe types are listed in Fig. 5. In the third step, we apply the augmentation method via deep learning style transfer. Examples of stylization of five different raw synthetic patches using five different real style images are presented in Fig. 6.

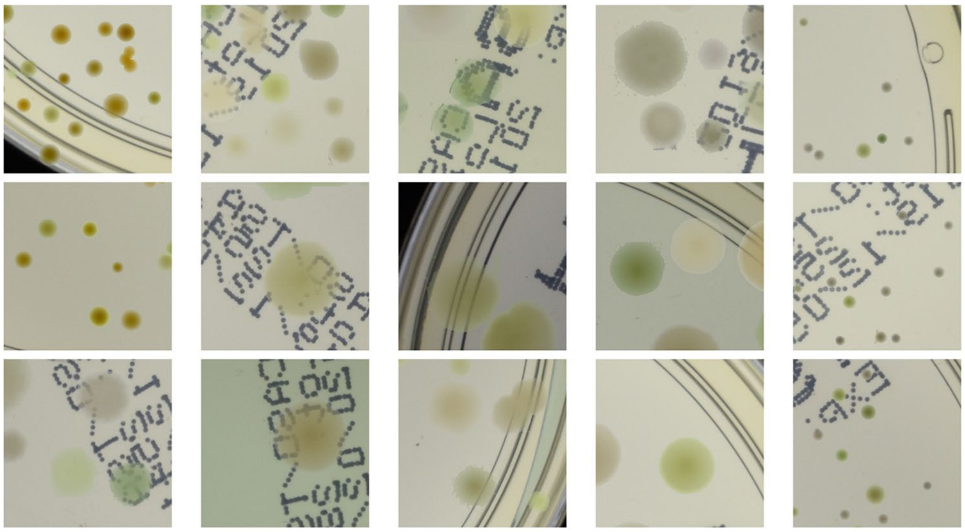

Figure 5. Examples of generated synthetic patches before the stylization step. In some situations, colonies do not blend well with the background, and their color does not match the background color.

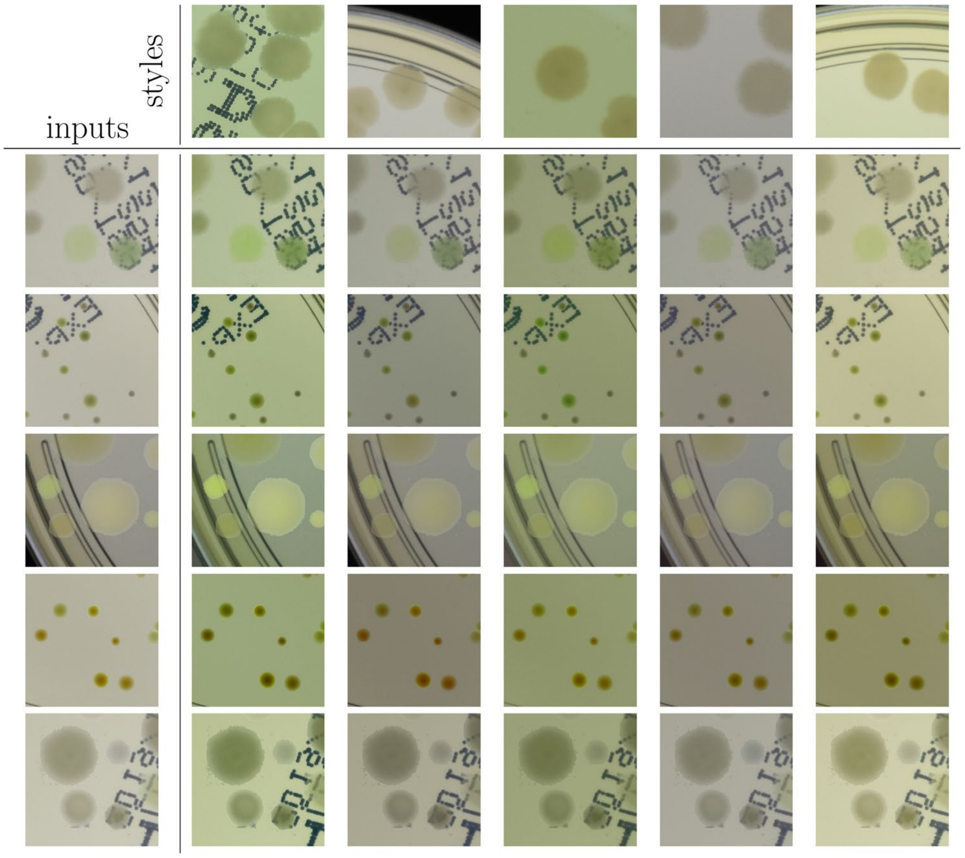

Figure 6. Stylization of generated microbial data. Five synthetic patches (left) are stylized (with $\lambda = 0.05$) using the style of five real images (top). Stylization improves realism of the generated images and significantly increases performance of a deep learning detector trained on this data.

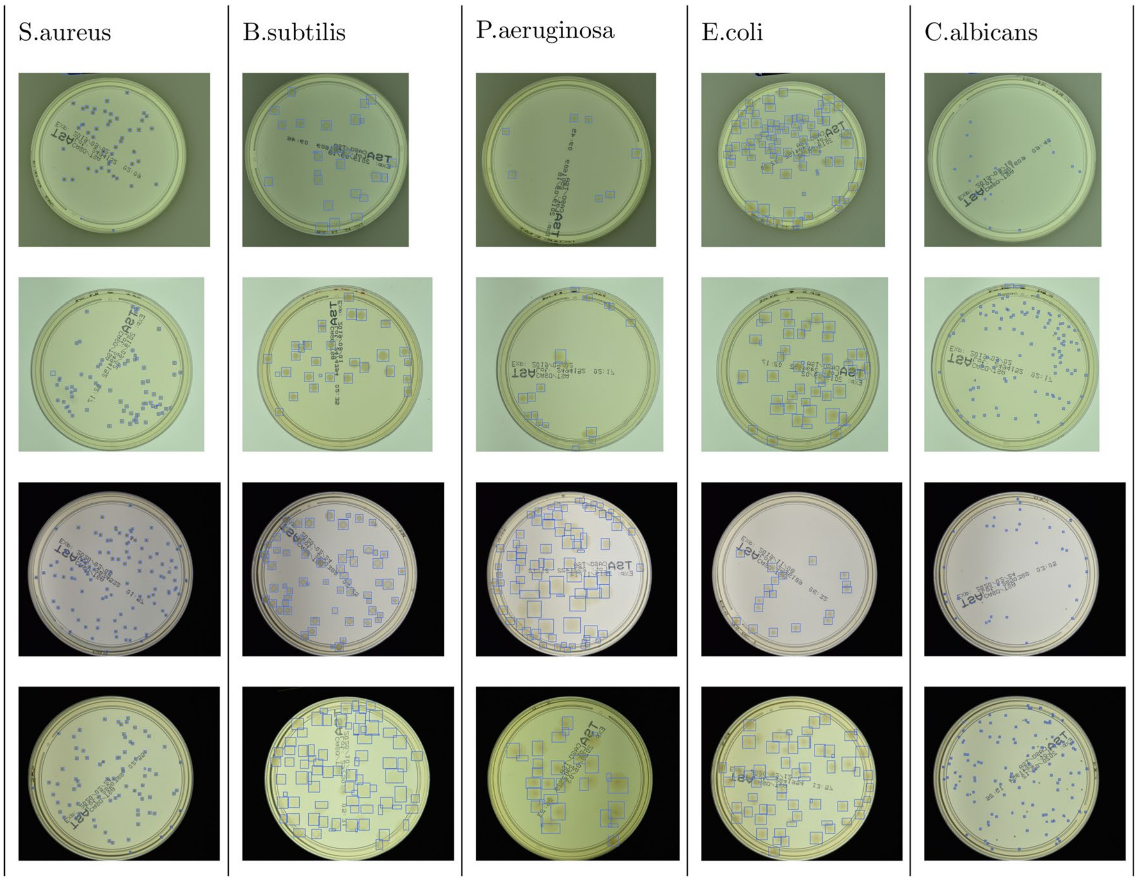

Figure 7. Examples of microbial colonies detection on real data—Petri dishes from AGAR dataset. The Faster R-CNN detector correctly detects the vast majority of colonies, however some missing bounding boxes especially for crowded samples can be observed.

We introduce stylizing for two visual reasons. First, it makes the generated images look more realistic—they become difficult to distinguish from real images of Petri dishes. Second, stylizing mitigates the mismatch at the edges of pasted colonies, which is an artifact that strongly impacts the detector performance—the detector would undesirably learn to detect nonrealistic edges of microbial colonies. However, another important factor is that the addition of stylizing, and thus augmenting using these 20 different styles, simply increased the diversity of the quite homogeneous raw dataset, and had a big impact on detector training.

Using the method, we generated 50k of *raw* patches with different microbe types and their amount. To test the stylization impact, we use two different stylization strengths controlled by λ weight—see "Methods" section for further details, and apply them separately to the *raw* patches. Weaker stylization with $\lambda = 0.02$ gives *semi-stylized* set, while stronger one with $\lambda = 0.05$ gives a *fully-stylized* set. We also store the *raw* set with no stylization. Each of the three sets contains 50k patches which corresponds to about 65 k of patches in the training part of the *higher-resolution* subset of AGAR dataset.

Deep learning detection on real data. The idea behind the conducted experiments is to train a neural network model using synthetic data to detect microbial colonies, and then test its performance on real images with bacterial colonies in Petri dish.

Examples of detection on real data. We train the Faster R-CNN and Cascade R-CNN on the three synthetic sets separately. The best results are obtained for training on the *fully-stylized* set—the examples of tests for real patches in case of Faster R-CNN are presented in Fig. 7. Detector performs quite well, but one may notice missing bounding boxes for some colonies, especially in crowded samples where colonies frequently overlap. The worst performance occurs for blurred *P.aeruginosa* microbes that form the biggest colonies; the best occurs for sharp small colonies of *S.aureus* and *C.albicans* microbes. In some cases, some excessive (false positive) bounding boxes also appear. In case of Cascade R-CNN, the results are similar with a slightly lower tendency of the detector to generate excessive bounding boxes, which results in better counting statistics—see Table 2.

Automatic instance segmentation is a problem that occurs in many biomedical applications. During the patch generation, we also store a segmentation mask at a pixel level for each colony. This additional information can be used to train a deep learning instance segmentation model. We use the Mask R-CNN[19] model which extends Faster R-CNN detector that we have already trained. In the study, we train the model using pretrained weights

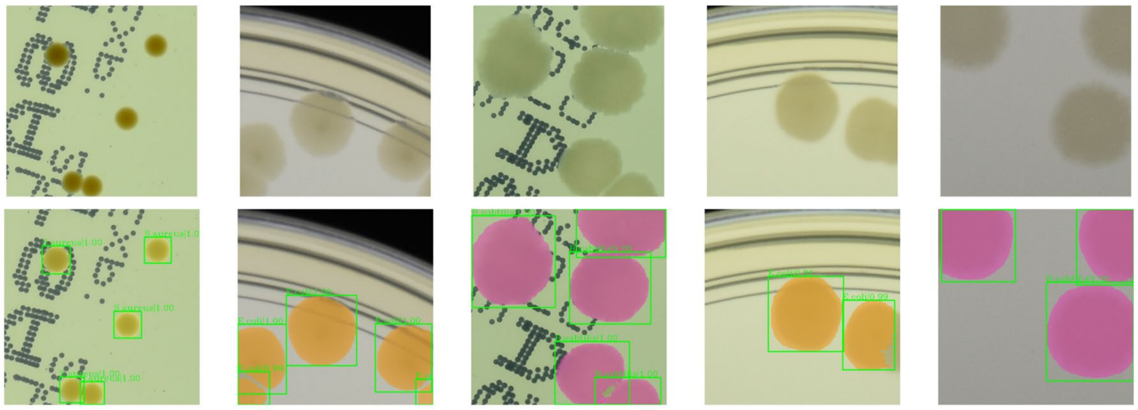

Figure 8. Example results for instance segmentation of real samples in case of different microbial species.

Metrics	Faster R-CNN			Cascade R-CNN		
	Detection	Counting		Detection	Counting	
Datasets	mAP (IoU = 0.50:0.95)	MAE	sMAPE (%)	mAP (IoU = 0.50:0.95)	MAE	sMAPE (%)
Raw	0.203	17.54	33.02	0.201	17.89	34.38
Semi-stylized	0.329	6.26	12.91	0.325	6.62	15.35
Fully-stylized	0.401	5.82	11.65	0.416	4.49	10.83
Real[12]	0.493	4.75	5.32	0.520	4.31	4.86

Table 2. Microbial colonies detection tested on real data for three different synthetic training datasets: *raw*, *semi-stylized*, and *fully-stylized*. For comparison, we also provide results for training on the real dataset[12]. We present results for Faster R-CNN and Cascade R-CNN detectors along with various metrics describing detection (mAP), and counting fidelity (MAE, sMAPE).

and refine colony bounding boxes obtained from the detector model. The segmentation results for real samples are presented in Fig. 8. The obtained instance segmentation for different microbial colony types correctly reproduces the colony shapes.

Colony counting metrics. One of the main applications of object detection in microbiology is to automate the process of counting microbial colonies grown on a Petri dish. We verify the proposed method of synthetic dataset generation by comparing it with a standard approach where we collect a big real dataset[12] and train the detector for colony detecting and counting tasks.

There are several types of metrics typically used in the context of object detection and counting. To characterize the quality of detection, we calculate the standard mean Average Precision (mAP) established by the famous COCO competition[58]. The Average Precision (AP) of detection is calculated for a given Intersection over Union (IoU) threshold. IoU describes the level of overlapping between the truth and predicted bounding boxes. The mean value of AP for different IoU thresholds and classes gives the mAP. To measure the effectiveness of colony counting, two separate metrics were used, namely, standard Mean Absolute Error (MAE) and symmetric Mean Absolute Percentage Error (sMAPE), which additionally weights the errors using information about the number of instances present on a given dish. Precise definitions of both measures can be found in Supplementary Material for[12].

Let us now discuss the results of counting microbial colonies on a Petri dish. We train the chosen detector (Faster RCNN with ResNet-50 backbone and Cascade R-CNN with HRNet) on a 50k large dataset generated using 100 images of the training part of the *higher-resolution* AGAR subset, and test microbial colonies counting in the same task as conducted in[12] (testing part of AGAR *higher-resolution*). The resulting detection metrics are presented in Table 2, and counting results in Fig. 9 (the detectors were trained on *raw*, *semi-stylized*, and *fully-stylized* sets separately). It turns out that the detection precision itself (mAP) and counting errors (MAE and sMAPE) for the *fully-stylized* set are only slightly worse (especially MAE) than for the same detector but trained on the whole big set containing over **7360** of real images giving about 65 k of patches. Cascade R-CNN detects colonies slightly better than Faster R-CNN and this is true for training on both synthetic and real data. It is also interesting to note that Cascade R-CNN is better for the most stylized set, for the others, *semi-stylized* and *raw*, Faster R-CNN performs better.

It is also clear that introducing style transfer augmentation greatly improves the detection quality, and without stylization the results are rather poor—see results for *raw* set in Fig. 9(left). The top row in Fig. 9 shows counting results for Faster, while the bottom row for Cascade detector. Moreover, in the Fig. 9(right)—results for *fully–stylized* set, we observe the typical problem where the detector underestimates the number of colonies

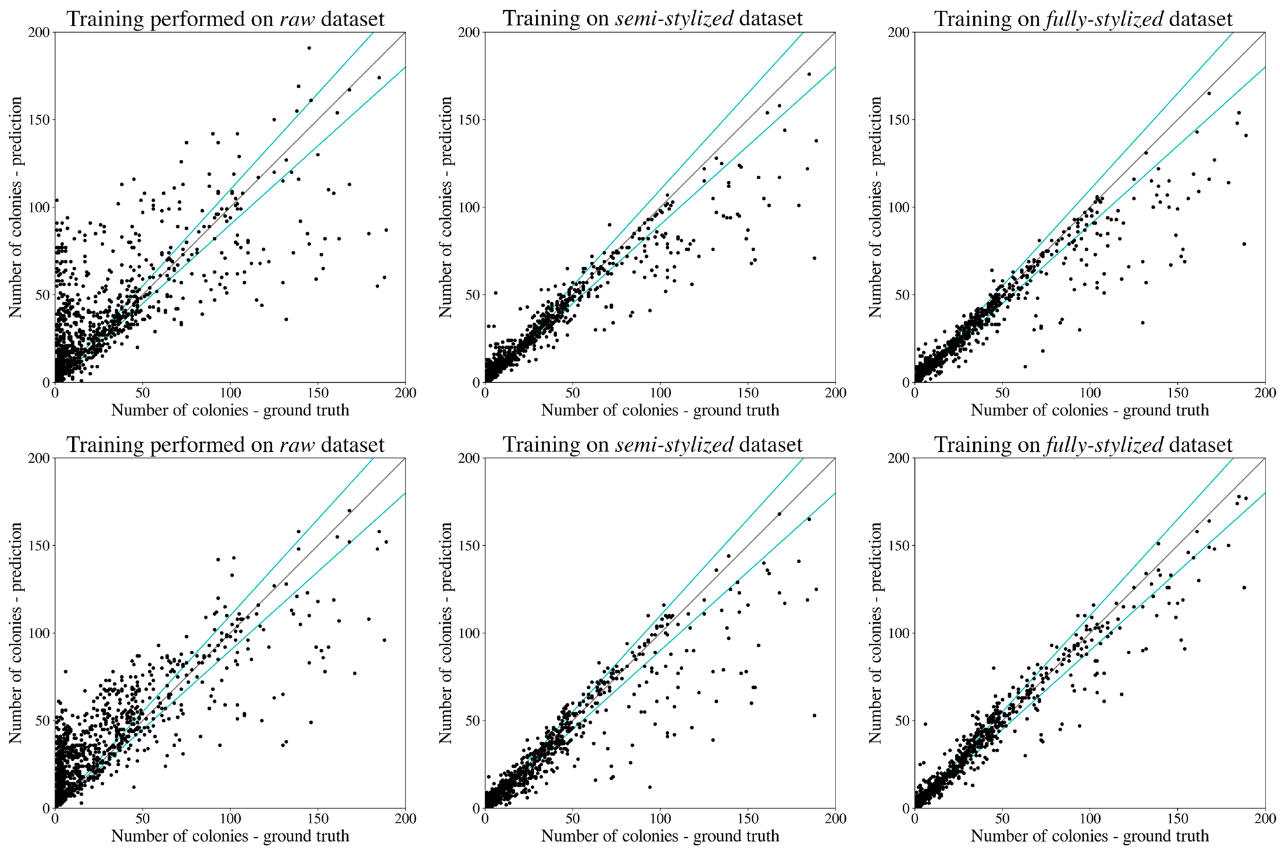

Figure 9. Microbial colonies counting tested on real data for three different synthetic training datasets: *raw* (left), *semi-stylized* (middle), and *fully-stylized* (right). In ideal detection every black dot representing a single Petri dish image should lay on the $y = x$ black line. Top row shows results for Faster, while bottom row for the Cascade detector.

for highly populated dishes. This is because in such cases colonies (especially bigger ones) frequently overlap, which makes detection (and counting) harder.

Discussion

Object counting problems, such as estimating the number of cells in a microscopic image or the number of humans or cars in a video frame, have become one of the major tasks in computer vision these days[59]. Generation of synthetic microbiological images using simple geometrical transformations that emulate microscopic views of bacterial colonies was used for validation of various algorithms for automated image analysis[60], or to train a high-resolution convolutional neural network to build density-map-based cell counters[61]. In the latter, authors use methods similar to ours, where the model was trained using synthetic data and tested on real images. However, in our approach we train a much more complicated but thus more flexible (and prone to generalize) object detector. Therefore, we need to use a much bigger and more diverse dataset containing objects of different sizes and with different lighting conditions. We also verified that further enlarging the dataset by adding another 10k patches practically did not improve the detection fidelity. Deep CNNs were also trained to count cells[62] and microbial colonies[16,46], but in these approaches the network training was done entirely in a supervised manner using real datasets.

Simple augmentation techniques were applied successfully to extend microbial datasets[46,63] in bacteria classification tasks. We also flipped or rotated the images, or added salt and pepper noise or speckle noise, but with no significant effect on the training process. According to our experiments, only advanced augmentation by using style transfer significantly affects the training of the detector. It is also worth mentioning that adding more styles did not help much; during the generation we now choose from 20 style images, but when we increased this number to 50, it did not give further improvement. On the other hand, the degree of styling was more important, but if it was too high, i.e $\lambda > 0.05$, the fidelity of microbe classification would decrease. Other groups involved in the generation of synthetic microbial images also reported the usefulness of style transfer methods[35]. They use the style transfer algorithm in the microbial colony segmentation task to improve the realism of synthetic images of Petri dishes generated using GANs. However, we additionally show that style transfer allows us not only to improve realism, but also to introduce different color or lighting domains, and thus enhance the trained detector's ability to generalize. We saw in Fig. 7 that by using different style images during training, the detector can learn to count in slightly different lighting conditions (domains) at the same time. Therefore, our approach can be used to quickly change domains, e.g., we can collect styles and empty dishes in different lighting, and then generate synthetic images in that domain.

It is also worth emphasizing that standard GANs are not applicable in our case because to train the detector in a supervised manner we need to know exactly where each colony will appear. We need to control precisely where they are located—their placement and number on a dish, which is not possible with generic, unconstrained GAN. On the other hand, there are variations of GANs such as cycle GAN or conditional GAN (e.g. pix2pix translation[24]) that would allow colonies to be generated at desired locations. However, our preliminary experiments have shown that while we are able to generate realistic looking images in a single style with pix-2pix, training of this type of networks is difficult and it is much harder to achieve sufficient diversity of domains using this approach. As a result, the detector trained on such data performs worse. On the other hand, one-shot style transfer is simply faster (especially for high resolution images), easier to train (without mode collapse or converge failure), and able to transfer with high resolution to different domains. Moreover, data leakage, which is more problematic for medical investigation due to privacy issues, can be controlled with proposed generation strategy. Therefore, to generate the data we used a hybrid of traditional computer vision and image style transfer rather than individual generative models.

Conclusions

Training modern deep learning models requires large and diverse datasets containing thousands of labeled images[58]. By using traditional computer vision techniques complemented by a deep learning style transfer algorithm, we were able to build a microbial data generator supplied with only 100 real images that demands much less effort and resources than collecting and labeling a vast dataset containing thousands of real images.

In principle, any object with some local differences from the background (in colorspace or brightness) can be segmented and then used to generate images in our scheme together with the appropriate transfer of styles. We showed that, once generated, our synthetic dataset allows us to train state-of-the-art deep learning object detectors, making it applicable to any object detection task in microbiology, but also in other scientific or industrial applications.

Received: 3 December 2021; Accepted: 21 March 2022
Published online: 25 March 2022

References

1. Levine, A. B. *et al.* Rise of the machines: Advances in deep learning for cancer diagnosis. *Trends Cancer* **5**, 157–169. https://doi.org/10.1016/j.trecan.2019.02.002 (2019).
2. Wang, G., Ye, J. C. & Man, B. D. Deep learning for tomographic image reconstruction. *Nat. Mach. Intell.* **8**, 737–748. https://doi.org/10.1038/s42256-020-00273-z (2020).
3. Varma, M. *et al.* Automated abnormality detection in lower extremity radiographs using deep learning. *Nat. Mach. Intell.* **1**, 578–583. https://doi.org/10.1038/s42256-019-0126-0 (2019).
4. Faust, K. *et al.* Intelligent feature engineering and ontological mapping of brain tumour histomorphologies by deep learning. *Nat. Mach. Intell.* **1**, 316–321 (2019).
5. Zeune, L. L. *et al.* Deep learning of circulating tumour cells. *Nat. Mach. Intell.* **2**, 124–133 (2020).
6. Salehinejad, H. *et al.* A real-world demonstration of machine learning generalizability in the detection of intracranial hemorrhage on head computerized tomography. *Sci. Rep.*https://doi.org/10.1038/s41598-021-95533-2 *(2021).*
7. Vaidyanathan, A. *et al.* Deep learning for the fully automated segmentation of the inner ear on mri. *Sci. Rep.*https://doi.org/10.1038/s41598-021-82289-y *(2021).*
8. Belikova, K., Rogov, O. Y., Rybakov, A., Maslov, M. V. & Dylov, D. V. Deep negative volume segmentation. *Sci. Rep.*https://doi.org/10.1038/s41598-021-95526-1 *(2021).*
9. Jumper, J. *et al.* Highly accurate protein structure prediction with alphafold. *Nature*https://doi.org/10.1038/s41586-021-03819-2 *(2021).*
10. Selinummi, J., Seppälä, J., Yli-Harja, O. & Puhakka, J. A. Software for quantification of labeled bacteria from digital microscope images by automated image analysis. *Biotechniques* **39**, 859–863 (2005).
11. Chen, W.-B. & Zhang, C. An automated bacterial colony counting and classification system. *Inf. Syst. Front.* **11**, 349–368 (2009).
12. Majchrowska, S. *et al.* AGAR a microbial colony dataset for deep learning detection. Preprint arXiv:2108.01234 (2021).
13. Beznik, T., Smyth, P., de Lannoy, G. & Lee, J. A. Deep learning to detect bacterial colonies for the production of vaccines. Preprint arXiv:2009.00926 (2020).
14. Jiang, N. & Yu, F. Multi-column network for cell counting. *OSA Contin.* **3**, 1834–1846. https://doi.org/10.1364/OSAC.396603 (2020).
15. Zhang, Y., Jiang, H., Ye, T. & Juhas, M. Deep learning for imaging and detection of microorganisms. *Trends Microbiol.* **29**, 569–572. https://doi.org/10.1016/j.tim.2021.01.006 (2021).
16. Wang, H. *et al.* Early detection and classification of live bacteria using time-lapse coherent imaging and deep learning. *Light Sci. Appl.* **9**, 1–17 (2020).
17. Jiao, L. *et al.* A survey of deep learning-based object detection. *IEEE Access* **7**, 128837–128868. https://doi.org/10.1109/access.2019.2939201 (2019).
18. Ren, S., He, K., Girshick, R. & Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**, 1137–1149 (2016).
19. He, K., Gkioxari, G., Dollár, P. & Girshick, R. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 2980–2988, https://doi.org/10.1109/ICCV.2017.322 (2017).
20. Zhu, L., Zhang, H., Ali, S., Yang, B. & Li, C. Crowd counting via multi-scale adversarial convolutional neural networks. *J. Intell. Syst.* **30**, 180–191. https://doi.org/10.1515/jisys-2019-0157 (2021).
21. Ronneberger, O., Fischer, P. & Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015* (eds Navab, N. *et al.*) 234–241 (Springer, 2015).
22. Goodfellow, I. *et al.* Generative adversarial networks. *Adv. Neural Inf. Process. Syst.*https://doi.org/10.1145/3422622 *(2014).*
23. Zhu, J.-Y., Park, T., Isola, P. & Efros, A. Unpaired image-to-image translation using cycle-consistent adversarial networks. 2242–2251, https://doi.org/10.1109/ICCV.2017.244 (2017).
24. Isola, P., Zhu, J.-Y., Zhou, T. & Efros, A. A. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1125–1134 (2017).

25. Yi, X., Walia, E. & Babyn, P. Generative adversarial network in medical imaging: A review. *Med. Image Anal.* **58**, 101552. https://doi.org/10.1016/j.media.2019.101552 (2019).
26. Gibson, E. *et al.* Niftynet: A deep-learning platform for medical imaging. *Comput. Methods Prog. Biomed.* **158**, 113–122. https://doi.org/10.1016/j.cmpb.2018.01.025 (2018).
27. Zakka, C., Saheb, G., Najem, E. & Berjawi, G. Mammoganesis: Controlled generation of high-resolution mammograms for radiology education. Preprint arXiv:2010.05177 (2020).
28. Segal, B., Rubin, D. M., Rubin, G. & Pantanowitz, A. Evaluating the clinical realism of synthetic chest x-rays generated using progressively growing gans. *SN Comput. Sci.* **2**, 1–17 (2021).
29. Bissoto, A., Perez, F., Valle, E. & Avila, S. Skin lesion synthesis with generative adversarial networks. In Stoyanov, D. *et al.* (eds.) *OR 2.0 Context-Aware Operating Theaters, Computer Assisted Robotic Endoscopy, Clinical Image-Based Procedures, and Skin Image Analysis*, 294–302 (Springer, 2018).
30. Ghorbani, A., Natarajan, V., Coz, D. & Liu, Y. Dermgan: Synthetic generation of clinical skin images with pathology. In *Machine Learning for Health Workshop*, 155–170 (PMLR, 2020).
31. Costa, P. *et al.* Towards adversarial retinal image synthesis. arXiv:abs/1701.08974 (2017).
32. Niu, Y., Gu, L., Zhao, Y. & Lu, F. Explainable diabetic retinopathy detection and retinal image generation. *IEEE J. Biomed. Health Inform.* (2021).
33. Zunair, H. & Hamza, A. B. Melanoma detection using adversarial training and deep transfer learning. *Phys. Med. Biol.* **65**, 135005 (2020).
34. Welander, P., Karlsson, S. & Eklund, A. Generative adversarial networks for image-to-image translation on multi-contrast mr images-a comparison of cyclegan and unit. Preprint arXiv:1806.07777 (2018).
35. Andreini, P., Bonechi, S., Bianchini, M., Mecocci, A. & Scarselli, F. Image generation by gan and style transfer for agar plate image segmentation. *Comput. Methods Prog. Biomed.*https://doi.org/10.1016/j.cmpb.2019.105268 *(2020).*
36. Savardi, M., Ferrari, A. & Signoroni, A. Automatic hemolysis identification on aligned dual-lighting images of cultured blood agar plates. *Comput. Methods Prog. Biomed.* **156**, 13–24. https://doi.org/10.1016/j.cmpb.2017.12.017 (2018).
37. Zieliński, B. *et al.* Deep learning approach to bacterial colony classification. *PloS one* **12**, e0184554 (2017).
38. Li, M., Ye, C. & Li, W. High-resolution network for photorealistic style transfer. Preprint arXiv:1904.11617 (2019).
39. Gatys, L. A., Ecker, A. S. & Bethge, M. Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2414–2423, https://doi.org/10.1109/CVPR.2016.265 (2016).
40. Johnson, J., Alahi, A. & Fei-Fei, L. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 694–711 (Springer, 2016).
41. Zhang, H. & Dana, K. Multi-style generative network for real-time transfer. Preprint arXiv:1703.06953 (2017).
42. Li, Y. *et al.* Universal style transfer via feature transforms. Preprint arXiv:1705.08086 (2017).
43. Yoo, J., Uh, Y., Chun, S., Kang, B. & Ha, J.-W. Photorealistic style transfer via wavelet transforms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 9036–9045 (2019).
44. Pawłowski, J. Microbial dataset generation software. https://github.com/jarek-pawlowski/microbial-dataset-generation (2022).
45. Brugger, S. D. *et al.* Automated counting of bacterial colony forming units on agar plates. *PloS one* **7**, e33695 (2012).
46. Ferrari, A., Lombardi, S. & Signoroni, A. Bacterial colony counting with convolutional neural networks in digital microbiology imaging. *Pattern Recogn.* **61**, 629–640. https://doi.org/10.1016/j.patcog.2016.07.016 (2017).
47. Bär, J., Boumasmoud, M., Kouyos, R. D., Zinkernagel, A. S. & Vulin, C. Efficient microbial colony growth dynamics quantification with coltapp, an automated image analysis application. *Sci. Rep.* **10**, 1–15 (2020).
48. Buades, A., Coll, B. & Morel, J.-M. Non-Local Means Denoising. *Image Process. Line* **1**, 208–212 (2011).
49. Chan, T. & Vese, L. An active contour model without edges. In *International Conference on Scale-Space Theories in Computer Vision*, 141–151 (Springer, 1999).
50. Getreuer, P. Chan-Vese segmentation. *Image Process. Line* **2**, 214–224 (2012).
51. Van der Walt, S. *et al.* scikit-image: Image processing in python. *PeerJ* **2**, e453 (2014).
52. Wang, J. *et al.* Deep high-resolution representation learning for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **43**, 3349–3364. https://doi.org/10.1109/TPAMI.2020.2983686 (2021).
53. Simonyan, K. & Zisserman, A. Very deep convolutional networks for large-scale image recognition. Preprint arXiv:1409.1556 (2014).
54. Cai, Z. & Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6154–6162, https://doi.org/10.1109/CVPR.2018.00644 (2018).
55. He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778, https://doi.org/10.1109/CVPR.2016.90 (2016).
56. Chen, K. *et al.* MMDetection: Open mmlab detection toolbox and benchmark. Preprint arXiv:1906.07155 (2019).
57. Deng, J. *et al.* Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255, https://doi.org/10.1109/CVPR.2009.5206848 (2009).
58. Lin, T.-Y. *et al.* Microsoft coco: Common objects in context. In Fleet, D., Pajdla, T., Schiele, B. & Tuytelaars, T. (eds.) *Computer Vision—ECCV 2014*, 740–755 (Springer, 2014).
59. Lempitsky, V. & Zisserman, A. Learning to count objects in images. *Adv. Neural Inf. Process. Syst.* **23**, 1324–1332 (2010).
60. Lehmussola, A., Ruusuvuori, P., Selinummi, J., Huttunen, H. & Yli-Harja, O. Computational framework for simulating fluorescence microscope images with cell populations. *IEEE Trans. Med. Imaging* **26**, 1010–1016 (2007).
61. Xie, W., Noble, J. A. & Zisserman, A. Microscopy cell counting and detection with fully convolutional regression networks. *Comput. Methods Biomech. Biomed. Eng. Imaging Vis.* **6**, 283–292 (2018).
62. Xie, Y., Xing, F., Kong, X., Su, H. & Yang, L. Beyond classification: structured regression for robust cell detection using convolutional neural network. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 358–365 (Springer, 2015).
63. Khalifa, N. E. M., Taha, M. H. N., Hassanien, A. E. & Hemedan, A. A. Deep bacteria: Robust deep learning data augmentation design for limited bacterial colony dataset. *Int. J. Reason.-Based Intell. Syst.* **11**, 256–264 (2019).

Acknowledgements

Authors would like to thank Agnieszka Pawlak for valuable discussions. Project "Development of a new method for detection and identifying bacterial colonies using artificial neural networks and machine learning algorithms" is co-financed from European Union funds under the European Regional Development Funds as part of the Smart Growth Operational Program. Project implemented as part of the National Centre for Research and Development: Fast Track (Grant No. POIR.01.01.01-00-0040/18).

Author contributions
J.P. constructed the generator algorithm and generated the synthetic dataset. J.P. and S.M. trained the deep learning detectors and analyzed the results. T.G. led the team throughout the project. All authors provided critical feedback and helped shape the research, analysis, and manuscript.

Competing interests
The authors declare no competing interests.

Additional information
Correspondence and requests for materials should be addressed to J.P.