Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Towards an automatic analysis of CHO-K1 suspension growth in microfluidic single cell cultivation

Jan Philip Goepfert, Barbara Hammer

2021

Preprint:

This is an accepted article published in Bioinformatics. The final authenticated version is available online at: https://doi.org/10.1093/bioinformatics/btab386

arXiv:2010.10124v2 [cs.LG] 17 May 2021

Towards an Automatic Analysis of CHO-K1 Suspension Growth in Microfluidic Single-cell Cultivation

Preprint, compiled May 18, 2021

Dominik Stallmann Jan P. Göpfert Julian Schmitz Alexander Grünberger Barbara Hammer

Bielefeld University, Germany

Abstract

Motivation: Innovative microfluidic systems carry the promise to greatly facilitate spatio-temporal analysis of single cells under well-defined environmental conditions, allowing novel insights into population heterogeneity and opening new opportunities for fundamental and applied biotechnology. Microfluidics experiments, however, are accompanied by vast amounts of data, such as time series of microscopic images, for which manual evaluation is infeasible due to the sheer number of samples. While classical image processing technologies do not lead to satisfactory results in this domain, modern deep learning technologies such as convolutional networks can be sufficiently versatile for diverse tasks, including automatic cell counting as well as the extraction of critical parameters, such as growth rate. However, for successful training, current supervised deep learning requires label information, such as the number or positions of cells for each image in a series; obtaining these annotations is very costly in this setting.

Results: We propose a novel machine learning architecture together with a specialized training procedure, which allows us to infuse a deep neural network with human-powered abstraction on the level of data, leading to a high-performing regression model that requires only a very small amount of labeled data. Specifically, we train a generative model simultaneously on natural and synthetic data, so that it learns a shared representation, from which a target variable, such as the cell count, can be reliably estimated.

Availability: The project is cross-platform, open-source and free (MIT licensed) software. We make the source code available at https://github.com/dstallmann/cell_cultivation_analysis; the data set is available at https://pub.uni-bielefeld.de/record/2945513

Contact: dstallmann@techfak.uni-bielefeld.de

1 Introduction

New and improved single-cell analysis technologies employing live cell imaging allow researchers to acquire cellular information with ever-increasing levels of detail [11]. Especially for the study of cellular heterogeneity in clonal populations, which has long been ignored, single-cell analysis is mandatory. Here, microfluidic single-cell cultivation (MSCC) in particular is a promising tool to cultivate cells and analyze single-cell behavior [27]. Due to its microfluidic setup, MSCC allows for maintaining clearly defined cultivation environments or even dynamic changes of cultivation conditions [18]. Furthermore, MSCC encompasses live cell imaging and thereby allows systematic single-cell studies with high spatial and temporal resolution of cellular behavior. Despite its beneficial features for single-cell analysis, MSCC comes with a major bottleneck: enormous amounts of detailed data are generated that need processing.

So far, image analysis has mostly been performed manually or semi-manually. Therefore, MSCC requires adaptations with regards to the processing of the data produced. This either means the training of human experts and resulting extensive labor, or the development of new algorithms, where current approaches typically require either careful tuning or manually labeled data for training. In the long run, this is not feasible, and different, more affordable computer vision solutions are required [31].

In recent years, deep neural networks, in particular convolutional neural networks, which naturally mirror spatial priors, have been established as the standard technology for automated image processing and computer vision [15], and a considerable number of applications can be found in the biomedical domain [23]. While trained deep network models such as YOLO, VGG, or ResNet are readily available for natural scenes, these models do not easily transfer to other domains or imaging technologies, due to the different statistical properties observed in these fields.

There exist a number of approaches that tackle the challenge to track cells in images [21]: Fundamental progress has been made by proposing a benchmark suite based on different imaging technologies and comparing the strengths and limitations of diverse methods for cell tracking [33]. Still, one conclusion is the observation that automatic tracking remains prone to errors, and the tuning of model parameters for novel domains can be very cumbersome. The approach [3] enables an interactive bioimage analysis framework for intuitive user interaction.

The work [14] focuses on the challenge of gathering labeled data by providing a crowd based annotation tool, which allows a distribution of the work of manual labeling, based on which deep learning becomes possible. Interestingly, the work [5] displays the possibility to transfer results in between different imaging technologies to some extent. The approach [8] provides one of the few toolboxes for cell tracking, which allows transfer learning based on given models and novel data, whereby data set enrichment technologies limit the number of required samples. Yet, the data used consists of adherent cells rather than suspension cells.

In contrast to already reported single-cell cultivation studies [7] and [18], where adherent growing cell lines were the focus of

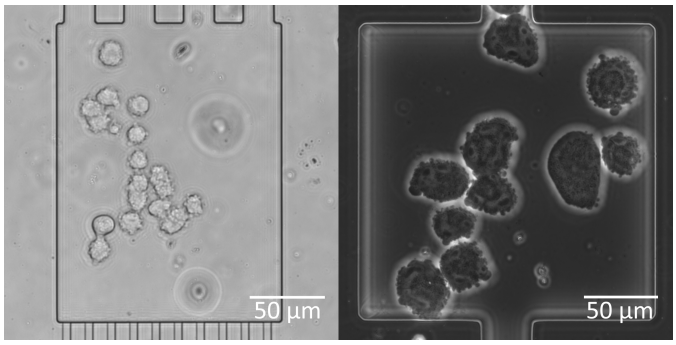


Figure 1: Samples from the data sets. Bright-field microscopy image on the left, phase-contrast microscopy image on the right. Faint and very small circles are smudges on the chip. The architecture has to differentiate between cells, smudges and background. Translational and rotational camera drift has been removed by semi-automatic image processing beforehand. The crop intentionally allows for data augmentation techniques to crop the images further. The scale bar has been added manually and is not part of the data set.

investigation, in this study we will concentrate on suspension cells, for which analysis tools of adherent cells are insufficient: suspension cells have a circular basic shape but ever-changing contour due to vesicle secretion and the addition of cell movement and floating within the chamber complicates tracking and analysis processes. Automatic analysis of these cells growing in suspension comes with different and challenging obstacles, which will be described in Section 2.

Due to this fact, we cannot transfer existing models, such as the ones mentioned above, to the given setting. We are interested in the question how to automatically provide sufficiently accurate cell counting, based on which the process dynamics can be characterized. We will use deep convolutional networks for this task. Unlike existing technologies, which are typically fully supervised, we put a particular focus on the challenge to mitigate the cumbersome task of manual labeling.

We propose a new deep twin auto-encoder architecture called `Twin-VAE`, which enriches the training data by artificial geometric data, for which ground truth labeled data can easily be generated. The proposed twin architecture greatly reduces the cost of synthetic, auxiliary training data, because that data does not need to appear realistic in all regards, such as morphological details and fidelity.

Training is based on a novel cost function that enriches the final task of cell counting by the challenge of correct representations. Optimization of this architecture is partially based on auto-ML technologies [12].

2 MATERIAL AND METHODS

2.1 MSCC and live cell imaging data

Image data applied in this study was obtained by single-cell cultivation of mammalian suspension cells as shown before [28]. CHO-K1 cells were cultivated in polydimethylsiloxane (PDMS)-glass-chips and constantly provided with nutrients by perfusion of the microfluidic device. Live cell imaging was accomplished with an automated inverted microscope (Nikon Eclipse Ti2, Nikon Instruments, Germany), every 20 minutes time lapse images of relevant positions on-chip were taken (NIS Elements AR 5.20.01 Software, Nikon Instruments, Germany). The data used in this work is split into two major parts according to the two microscopy technologies, namely bright-field microscopy and phase-contrast microscopy, that were used for the analysis of the cells (see Figure 1).

Table 1 shows an overview of all data sets. The sets declared as `Nat` contain the natural images mentioned above. Of these, `BF` declares bright-field microscopy images which consist of 12 scenarios accumulating to 956 used images, of which 7.5 % of the training images are labeled, while `PC` denotes phase-contrast microscopy images consisting of 37 scenarios and accumulates to 3976 used images of which 6.2 % of the training images are labeled. The labels are created in a regular interval over the data sets and experiment scenarios. Images with more than 30 cells were removed beforehand, since at this cell count, the following trend of the cultivation experiment is determined.

We put our focus on the larger data set `Nat-PC`. Not only does it contain more samples, but the captured biological processes are more varied, and phase-contrast microscopy is arguably more popular than bright-field microscopy. We show the distribution of images against the cell counts in them in Figure 2 for `Nat-PC`. For both `Nat-BF` and `Nat-PC`, we have labeled a number of images by hand, but we use most labels to test (`L-Te`) the reconstruction, rather than during training (`L-Tr`), because our goal is a method that does not rely on large amounts of labeled data. We use further unlabeled test data (`U-te`) to evaluate the reconstruction. We crop and rotate all images to center the cultivation chamber. Further data augmentation beyond this preprocessing is described in Section 2.3.4.

Among the more challenging aspects in the cultivation of suspension cells, in comparison to adherent cells, are the large number of visual characteristics, which prevent traditional cell counting techniques and tracking approaches [24]. As can be seen in Figure 1, there are a multitude of visual effects that complicate the automated counting process. Often, certain of these characteristics are being tackled, such as varying contrast and light conditions [6], or cells sticking together and overlapping each other [35], but additional ongoing cell divisions and changes in shape due to secretion of vesicles, which makes the cells' silhouettes highly irregular, impede the process and diminish the applicability of state-of-the-art solutions. Furthermore, appearing and vanishing of cells through the entrance of the chamber, a potential overpopulation of the culture and differing focus plains based on the cells' dimensions and inner organelles, make intracellular compartment non-uniform, further increasing the challenge to automate counting.

2.2 Approach

Our goal is reliable cell counting for suspension cell microscopic images. Since given data is limited and with only few manual annotations, we construct a novel deep neural network architecture (described in Section 2.3) to overcome these limitations. The challenge to learn from mostly unlabeled data, which are enriched by few manual labels, is usually referred to as semi-

supervised learning [9]. We will propose a novel technology to mitigate this complexity, by introducing synthetic data with known ground truth and a twin architecture, which can abstract from the fact that these enrichments are synthetic. We will now describe how we generate the data set enrichment, first, and introduce the specific neural architecture thereafter.

Since semi-supervised learning is a balancing act between reducing the share of labels required for acceptable results and increasing the share of labels to achieve desirable results [25], an additional dimension can be opened to tackle the problem from another side. We propose to generate an auxiliary training set `Syn` with simplified synthetic data, for which the ground truth is known because it is based on explicit geometric modeling, thereby simplifying the geometric heterogeneity of this data compared to real suspension cells drastically. Synthetic data has already successfully been used to improve the performance of artificial neural networks, e. g., for text processing [16]. Unlike data set enrichment, which directly enriches given data, synthetic data enables us to create a large variety of samples, which is not limited by the real data.

One might try to synthetically generate images that are indistinguishable from real ones, which would entail a considerate amount of engineering [10]. Instead, we rely only on basic geometric shapes to represent cells, neglecting concrete texture and intricate morphology. In the following, we show that this is adequate for successful training using our `Twin-VAE` architecture (Section 2.3) and it actually allows us to introduce our abstraction into the data: cells (in these kinds of images) can be represented by ellipses, especially when the objective is to count them.

Examples of the synthetic images can be seen in Figure 3. Table 1 lists this data as `Syn`, followed by the microscopy technology categorization `BF` or `PC` respectively, omitting the `U` or `L` declaration since all synthetically generated data is labeled and differentiates between independently generated training (`Tr`) and test (`Te`) images. The cell distribution in these images is close, but not identical, to that of the `Nat` data sets. These images are generated algorithmically with seed consistency, are thus reproducible and can be generated in arbitrary amount, however larger amounts of synthetic data will increase training time nearly linearly, while being assumed to improve performance with diminishing returns. For that and the sake of balance, the

Table 1: Overview of our data sets. `Nat` and `Syn` indicate natural and synthetic data. `BF` and `PC` indicate bright-field and phase-contrast. `L` and `U` indicate labeled and unlabeled data. `Tr` and `Te` indicate training and test data.

Abbreviation	Type	Technique	Labeled	Use	Size
`Nat-BF-L-Tr`	natural	bright-field	yes	Training	59
`Nat-BF-U-Tr`	natural	bright-field	no	Training	725
`Nat-BF-L-Te`	natural	bright-field	yes	Testing	62
`Nat-BF-U-Te`	natural	bright-field	no	Testing	110
`Nat-PC-L-Tr`	natural	phase-contrast	yes	Training	199
`Nat-PC-U-Tr`	natural	phase-contrast	no	Training	2983
`Nat-PC-L-Te`	natural	phase-contrast	yes	Testing	397
`Nat-PC-U-Te`	natural	phase-contrast	no	Testing	397
`Syn-BF-L-Tr`	synthetic	bright-field	yes	Training	1000
`Syn-BF-L-Te`	synthetic	bright-field	yes	Testing	1000
`Syn-PC-L-Tr`	synthetic	phase-contrast	yes	Training	3182
`Syn-PC-L-Te`	synthetic	phase-contrast	yes	Testing	794

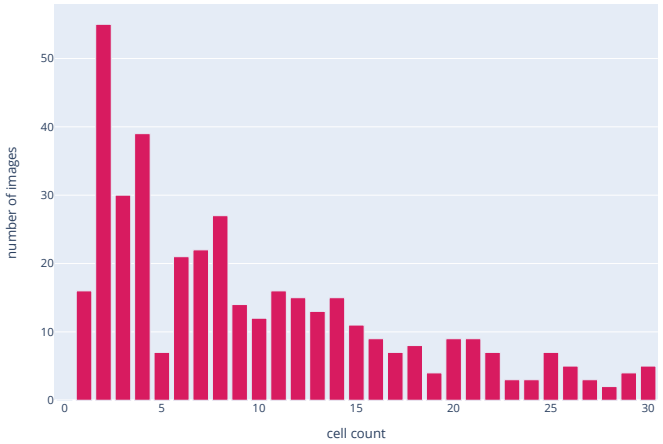


Figure 2: Distribution of images by cell count for `Nat-PC-L-Te` and `Nat-PC-U-Te`. We omit images that contain more than 30 cells because those are irrelevant for our cultivation experiments.

`Syn` set sizes are determined to roughly match the number of natural images. The background is created by taking the mean of all natural training images with few cells (to ensure high visibility of the background). The synthetic images are generated in the 128 by 128 pixels working resolution of the architecture described in Section 2.3 and their size is about 8 MB per 1000 images.

The generator is fully adjustable, creating images with a given distribution of cell counts, controls overlapping of cells, varies the brightness of the cell's inner organelles their membrane silhouette and the background, etc., such as in the natural data. Combined, these controls can also visualize the more complex visual aspects of natural data, such as ongoing cell divisions, by creating a small overlap together with more noisy cell borders. Smudges, as in Figure 1, have not been inserted, since they are an interference factor and are assumed to only hinder the training process. The cells have been given a circular shape to roughly match the shape of the natural cells. Some stretching or deforming to ellipses, noise, brightness fluctuation and Gaussian filters of varying strengths have been added to increase the variety of cells in the data. This geometric form can easily be adjusted if natural cells in other data sets have different shape characteristics.

2.3 Network Architecture & Training

2.3.1 Architecture

We propose a novel architecture that bypasses the problem of difference in appearance of synthetic and natural images, by separating the input of data for training depending on their origin, but requiring the architecture to create a tightly coupled shared inner representation to prevent high training losses. We do so by creating two identical variational autoencoders (VAE) for the two data sets, sharing the weights of their last layer of the encoder, the first layer of the decoder and the bottleneck in-between (see Figure 4). VAEs are the state-of-the-art solution for generalized few-shot learning [29] and weight-sharing has

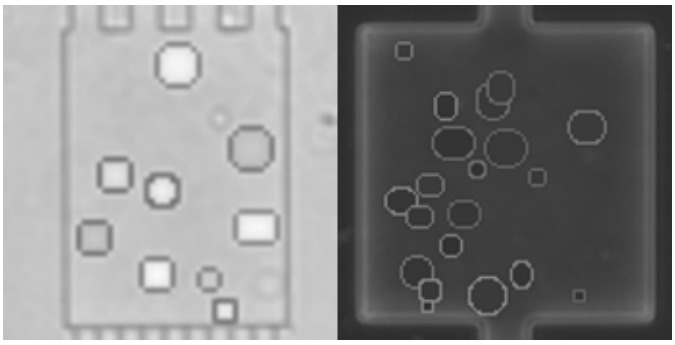


Figure 3: Samples of synthetic data from the proxy data sets `Syn-BF-L-Tr` and `Syn-PC-L-Tr`. Bright-field microscopy image on the left, phase-contrast microscopy image on the right. The images do not show scale bars, because they are dimensionless. A theoretical size could be calculated in relation to the chamber size, but it is of no importance for the work.

been used to reduce neural network sizes and to improve test performance before [32].

One of the VAEs uses synthetic data (*VAE-syn*), while the other one handles natural data only (*VAE-nat*). The different visual characteristics of synthetic and natural data are accounted for in the non-shared layers, while the shared layers rely on and enforce a common representation of relevant image characteristics. Besides auto-encoding, the architecture acts in a supervised way for data with known labels, by adding a two-layer-deep fully connected neural network regression model for the actual cell counting, based on the shared representation of the VAEs. Cell detection by regression has been shown to work well for other (less demanding) tasks [36, 35].

Our architecture addresses two objectives simultaneously:

- Encoding as well as decoding natural and synthetic input images using a shared representation.
- Predicting the present cell count for both types of images.

2.3.2 Loss

Given an input image x of pixels, a label (i. e., cell count) l between 1 and 30 and a type $t \in \{\mathfrak{n}, \mathfrak{s}\}$, representing the fact whether the image is natural or synthetic, we obtain a reconstruction loss $\mathrm{Rec}(x)$ of the VAE, a regression loss $\mathrm{Reg}(x, l)$ of the task at hand such as cell counting, and a distributional regularization loss $\mathcal{D}_{\mathrm{KL}}$, which aims for a homogeneous representation of synthetic and real data in the embedding space of the VAE. We combine these losses to form our twin loss $\mathrm{Twin}_{\mathrm{loss}}(x, l, t)$ with weighting factors C^t_{Rec}, $C^{t,l}_{\mathrm{Reg}}$, and $C^t_{\mathcal{D}_{\mathrm{KL}}}$, respectively, which allows us to balance the impacts between natural and synthetic images and to gracefully handle input images without known cell counts by setting $C^{t,l}_{\mathrm{Reg}}$ to zero:

$$\mathrm{Twin}_{\mathrm{loss}}(x, l, t) = C^t_{\mathrm{Rec}} \cdot \mathrm{Rec}(x) + C^{t,l}_{\mathrm{Reg}} \cdot \mathrm{Reg}(x, l) + C^t_{\mathcal{D}_{\mathrm{KL}}} \cdot \mathcal{D}_{\mathrm{KL}}(x) \quad (1)$$

During our experiments, the mean-squared error (MSE) $\|x - d(x)\|^2$, where $d(x)$ is the reconstruction of the input image x and $\|l - r(x)\|^2$, where $r(x)$ is the estimated cell count, yielded the best results respectively, when used as $\mathrm{Rec}(x)$ and $\mathrm{Reg}(x, l)$ for training on phase-contrast data, and as $\mathrm{Reg}(x, l)$ for bright-field data. However, for bright-field data the binary cross entropy (BCE) $-l \cdot log(r(x)) + (1 - l) \cdot log(1 - r(x))$ turned out to be the superior choice for $\mathrm{Rec}(x)$ and was often resulting in just slightly worse results than the MSE for phase-contrast data. The $\mathcal{D}_{\mathrm{KL}}$ is applied in the form of the Kullback-Leibler divergence (KLD) of the standard VAE ([17]) and is needed to generate latent vectors that are sufficiently close to a normal distribution. The weighting factors C^t_{Rec}, $C^{t,l}_{\mathrm{Reg}}$, and $C^t_{\mathcal{D}_{\mathrm{KL}}}$ have to be chosen carefully for training to succeed, punishing incorrect cell count predictions especially on natural cells, while relaxing the importance of visual reconstruction. We provide details in the following section.

2.3.3 Neural Network

The non-shared part of the encoder consists of four two-dimensional convolutional layers with kernel size 5 and stride 2, and are initialized with an orthogonal basis [26]. Between the layers are leaky rectified linear units (ReLU) with a leakiness of 0.2 as well as dropout of 0.1. The amount of channels used for the convolutions are 32, 64, 128 and 256, in this order, for the encoders. The shared part of the encoder has an additional two-dimensional convolutional layer with the same properties and 512 channels. The layer is followed by the bottleneck, which consists of three layers of fully connected neurons of sizes 512, 256 and 512 again, each with a dropout of 0.1. The shared part of the decoder has 256 channels, uses a two-dimensional transposed convolutional operator layer with the same kernel size and stride as above, and is followed by a batch normalization over a four-dimensional input and another leaky ReLU with a leakiness of 0.2. The non-shared part of the decoder consists of five layers of kernel sizes 5, 5, 5, 2, 6, a continuous stride of 2 except for the fourth layer with a stride of 1, the same leaky ReLUs and a sigmoidal activation function at the end. The output of the shared part of the decoder is also fed to a two-layer fully connected branch of neurons for the regressor of sizes 256 and 128. The regressor uses linear layers and a dropout of 0.1.

The architecture is using the Adam optimizer for phase-contrast microscopy data, and the rectified Adam (RAdam) [19] optimizer for bright-field data. The combination of the decoder loss factor $C_{\mathrm{Rec}} = 100$, the regressor loss factor $C_{\mathrm{Reg}} = 3$ and the KLD factor $C_{\mathcal{D}_{\mathrm{KL}}} = 2$ yield the best results for phase-contrast data. For the BCE, the decoder loss factor is not constant, but decays over time with a rate of 3×10^{-5} per epoch, since the BCE will not decrease significantly within the training process, but needs to decrease over time to amplify the importance of low regression losses $\mathrm{Reg}(x, l)$.

While it seems counter-intuitive that C_{Rec} is bigger than C_{Reg} and $C_{\mathcal{D}_{\mathrm{KL}}}$, it is caused by the MSE for pixel data getting very small on normalized images and it's actually being a desirable factor for the KLD to stay relatively small, since it is required to enhance the quality of the distributions, but should not impact the training of cell predictions too much by unfortunate sampling from the latent vector. Additionally a soft weight decay of 1×10^{-5} per epoch, a fixed learning rate of 1.3×10^{-4} and a delayed start for the regressor of 100 epochs were used to achieve the results presented in Section 3. A batch size of 128 for the phase-contrast

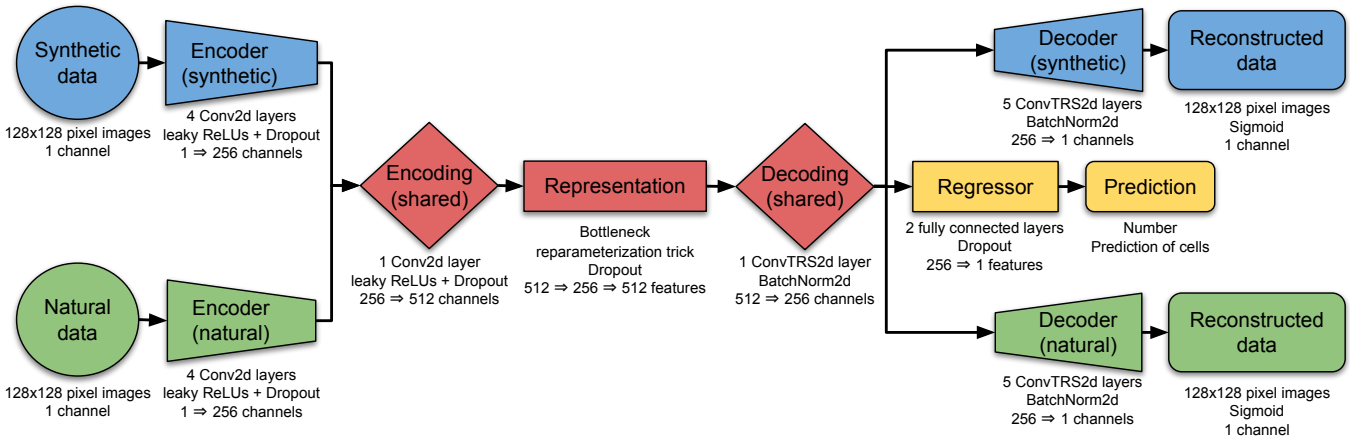


Figure 4: Visualization of the `Twin-VAE` architecture. The blue elements handle synthetic data, while the green elements handle natural data. The red elements are shared between the two VAEs and contain the inner representation of the cell imagery, while the yellow elements result in an estimation for the cell count.

images and 64 for the bright-field images worked best and the training runs for up to 50.000 epochs, unless early stopping conditions abort it.

To minimize the number of hyperparameters that had to be optimized by hand, a Bayesian optimization [4] in form of a Gaussian Process regressor [34] has been used to find good values for the learning rate, the number of channels of the bottleneck and the convolutional layers around it, as well as all three loss weight factors.

2.3.4 *Data Augmentation*

To maximize the use of the limited amounts of natural data, multiple data augmentation techniques were combined and applied to the data. Random horizontal and vertical flips are combined with a random resized crop of scale 0.9, meaning the images get randomly cropped to 115 by 115 pixels and then scaled back to 128 by 128 pixels. While the crop does add difficulty to the cell detection process by allowing cells to be at positions where, without the crop, only the chamber border and outside of the chamber would be, it proved necessary to establish the possibility of cells appearing anywhere on the image, assuring equalized detection success, barely impacted by the cell position within the image. Then, a 90 degree rotation is applied at random and a zero-centered noise map is generated and added to the image with a small amplitude factor.

2.4 *Baselines*

We implement two different methods to serve as baselines for our evaluation.

The first is a state of the art classical computer vision pipeline. First, the input images is cropped to only contain the cell chamber and blurred with a averaging kernel-based filter, then a thresholding filter is applied, followed by a watershed segmentation [22]. The regions of the segmented image are counted and used as cell estimation. To find good parameters for this pipeline, we performed an exhaustive grid search for each of the two data sets `BF` and `PC`. Our code repository contains the optimal parameters found. Because watershedding is the core component of this pipeline, we henceforth refer to it as `Watershed`.

As our second baseline, we fine-tune a pretrained state-of-the-art deep convolution neural network, specifically a variant of the *EfficientNet* [30]. We replace the last layer of the pretrained network by a fully-connected layer that outputs a single value and train this to predict the cell count for a given input image. We apply the same hyperparameter optimization as for our own method as well as the same data augmentation. Again, our code repository contains the exact implementation. *EfficientNet* is a variable architecture that comes in different sizes, referred to as *EfficientNet-B0*, *EfficientNet-B1*, and so forth. We tested *EfficientNet-B0* through *EfficientNet-B3* and found the smallest variant *EfficientNet-B0* to perform best, with larger variants performing progressively worse. We henceforth refer to this fine-tuned convolutional neural network as `EfficientNet` with a suffix to indicate the respective variant.

3 RESULTS

3.1 *Cell Counting*

We present the results of all methods on the data sets `Syn-PC-L-Te`, `Nat-PC-L-Te`, `Syn-BF-L-Te`, and `Nat-BF-L-Te` in Table 2. Our `Twin-VAE` consistently outperforms the other methods `Watershed` and `EfficientNet` by a wide margin. Even without any synthetic data, our architecture leads to respectable results (`Twin-VAE` (`Nat` only)); however, only through the simultaneous training on synthetic data do we realize its full potential.

We found different variants of our `Twin-VAE` to out-perform each other depending on whether we optimize for low deviation in cell counts or high accuracy (meaning correct cell counts) on natural data (`Nat`) Therefore, we include two variants in Table 2, namely $\texttt{Twin-VAE}_{\texttt{max-acc}}$ optimized for high accuracy, and $\texttt{Twin-VAE}_{\texttt{min-dev}}$ simultaneously optimized for low mean absolute error (MAE) and low mean relative error (MRE). The MRE normalizes the error by the true cell count, so that for high

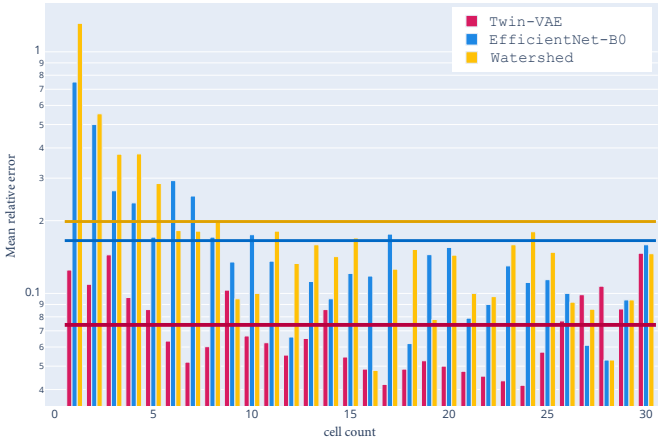


Figure 5: The mean relative error (MRE) for `Watershed`, `EfficientNet-B0`, and $\texttt{Twin-VAE}_{\texttt{min-dev}}$ on `Nat-BF-L-Te` on a logarithmic scale. Horizontal bars indicate the average MRE of their respective color and method.

cell counts, small deviations are not punished as much as they are for low cell counts.

`Twin-VAE` estimates cell counts correctly for around 58 % of all images in `Nat-PC-L-Te` and it's count is on average only 0.6 cells off the actual cell count in the images (MAE). and it achieves approximately 0.06 % mean relative error (MRE). As such, `Twin-VAE` proves capable of reliably counting cells in different microscopy technologies.

We visualize the mean relative error for `Watershed`, `EfficientNet-B0`, and $\texttt{Twin-VAE}_{\texttt{min-dev}}$ on `Nat-PC-L-Te` in Figure 5. The plot confirms the result from Table 2, showing the superiority our `Twin-VAE` over the other methods. Moreover, we see that `Twin-VAE` performs well across the entire range of cell counts in `Nat-PC-L-Te`. By contrast, `Watershed` and `EfficientNet-B0` struggle with images that contain few cells, which is the most important range of cell counts for the biological tasks, like estimating the growth rate.

3.2 Image Reconstruction

While our goal is automatic counting of cells, our loss Equation (1) includes a term for image reconstruction. An analysis of the reconstruction abilities of `Twin-VAE` is useful to ensure the learned shared representation is meaningful.

During training of `Twin-VAE`, natural inputs are first processed by a specialized encoder, then by a shared encoder and decoder, and finally reconstructed by a specialized decoder (see Figure 4). The equivalent is true for synthetic images. For the regression to work as intended and for the cell counting in natural images to benefit from synthetic data, the learned representation must be shared between the two types of data. We can verify this by encoding a natural image with the appropriate encoder but performing the reconstruction with the decoder intended and trained for synthetic images – or vice versa. In the following, we demonstrate exactly this.

In Figure 6 and Figure 7 we show examples of perfect translations, where a natural image is encoded and subsequently decoded as a synthetic image. The cell count is unchanged and the position and size of cells are also retained. The overall appearance is simplified, though: `Twin-VAE` has learned to remove noise and condense the information required to count cells.

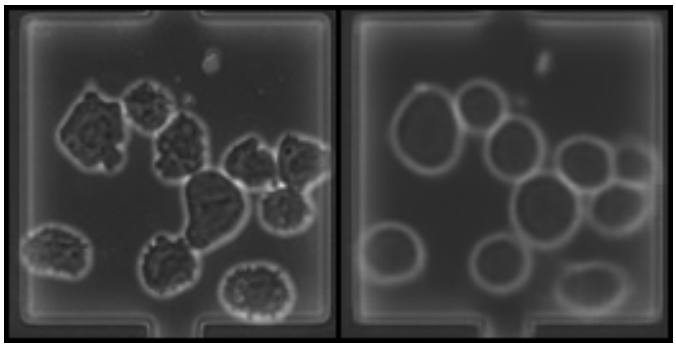


Figure 6: Example of a perfect synthetic-looking reconstruction (right) of a natural image (left) from `Nat-PC-L-Te`. The cell counts match exactly and the position as well as size of cells are preserved. While the top-right positioned smudge is recreated visually, it does not lead the regressional part of the `Twin-VAE` to a wrong cell count.

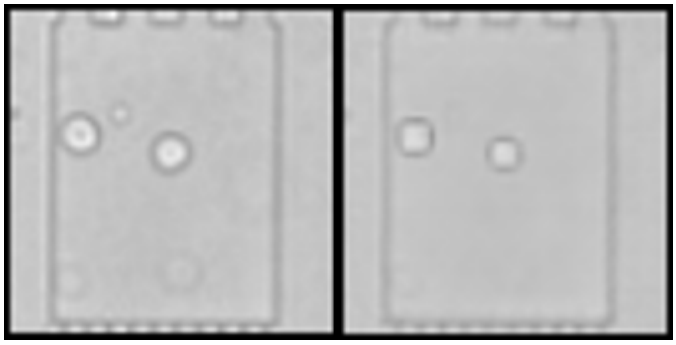


Figure 7: Example of a perfect synthetic-looking reconstruction (right) of a natural image (left) from `Nat-BF-L-Te`. The cell counts match exactly and the position as well as size of cells are preserved. For this data set, where smudges are more faint, they don't get reconstructed usually.

Even when `Twin-VAE` does not translate an image perfectly, the reconstruction can be useful to understand where an error occurs. In Figure 8 we show an example where two cells that are very close together are interpreted and reconstructed as a single cell.

As well as translating images from natural to synthetic-looking, `Twin-VAE` can perform the inverse translation from synthetic to natural-looking as well. We provide an example in Figure 9.

3.3 Visualizing the Shared Representation

The ability of `Twin-VAE` to translate between natural and synthetic images already demonstrates that the representation learned by the autoencoder is indeed semantically shared between the two types of images (natural and synthetic). We can go further and visualize the shared representation. Because each image is encoded as a 256-dimensional vector, we need to drastically reduce the dimensionality to do so.

Table 2: Evaluation of all methods on the data sets `Syn-PC-L-Te`, `Nat-PC-L-Te`, `Syn-BF-L-Te`, and `Nat-BF-L-Te`. For each method and data set, we report the mean absolute error (MAE), the mean relative error (MRE), and the accuracy. Ultimately, only the performance on natural data (`Nat`) is important, but we also report the performance on synthetic data (`Syn`) to provide further context. We use an upward arrow ↑ to indicate that higher is better and a downward arrow ↓ to indicate that lower is better.

Method	MAE (Syn) ↓	MRE / % (Syn) ↓	Acc. / % (Syn) ↑	MAE (Nat) ↓	MRE / % (Nat) ↓	Acc. / % (Nat) ↑
			PC (phase-contrast microscopy)			
Watershed	0.94	18.00	24.00	1.66	29.00	23.10
EfficientNet-B0	4.99	79.40	5.00	1.67	25.10	23.40
EfficientNet-B1	3.53	54.50	8.70	1.78	31.90	19.30
Twin-VAE (Nat only)	n/a	n/a	n/a	1.07	20.10	39.80
Twin-VAE$_{\text{max-acc}}$	**0.09**	**0.68**	**68.20**	0.60	5.92	**57.80**
Twin-VAE$_{\text{min-dev}}$	0.14	0.73	62.10	**0.59**	**5.66**	57.00
			BF (bright-field microscopy)			
Watershed	1.92	39.00	2.00	2.39	32.00	32.00
EfficientNet-B0	6.50	67.10	4.50	1.13	17.20	33.90
EfficientNet-B1	5.25	67.50	4.40	1.21	18.50	29.00
Twin-VAE (Nat only)	n/a	n/a	n/a	0.91	13.30	23.40
Twin-VAE$_{\text{max-acc}}$	**0.48**	**4.27**	**60.10**	0.68	7.60	**53.20**
Twin-VAE$_{\text{min-dev}}$	0.52	4.63	58.20	**0.63**	**7.31**	51.90

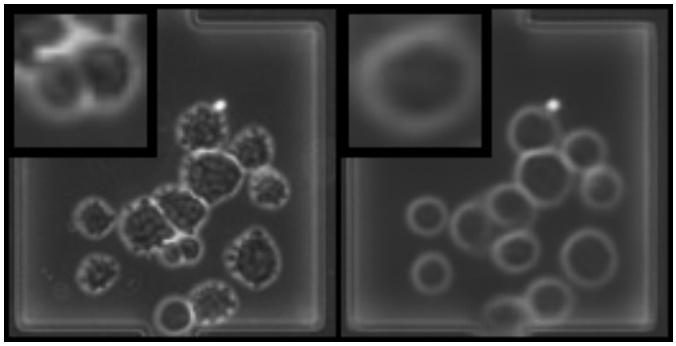


Figure 8: Example of a faulty synthetic-looking reconstruction (right) of a natural image (left) from `Nat-PC-L-Te`. The cell count is off by one, and we can visually spot the reason why: two cells that are very close in the natural original are merged into a single cell.

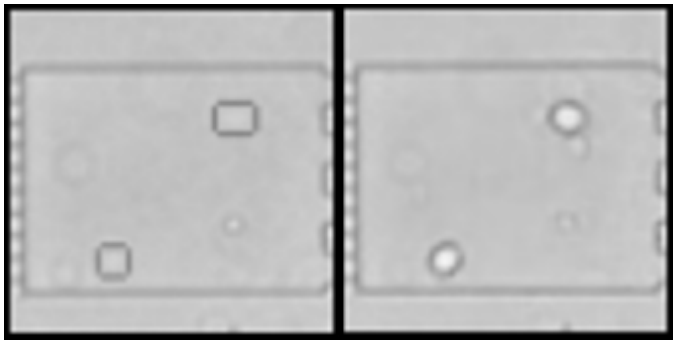


Figure 9: Example of a perfect natural-looking reconstruction (right) of a synthetic image (left) from `Syn-BF-L-Tr`. These conversions ensure representational consistency on a visually comprehensible level. More details on this in Figure 10. The cell counts match exactly and the position and size of cells are preserved.

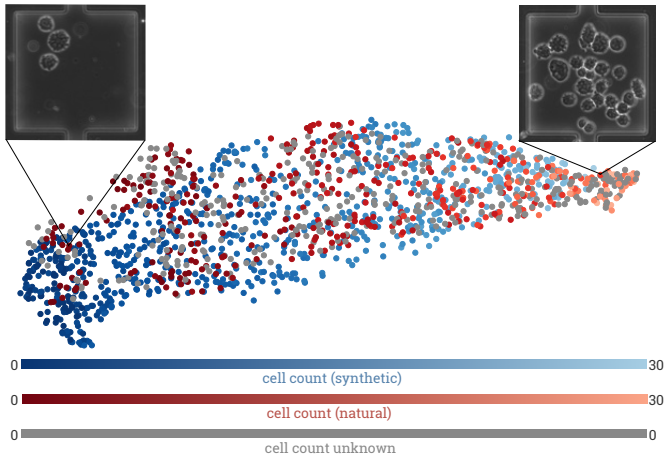


Figure 10: Embedding of learned representations, computed via UMAP. Visible are 392 natural samples (red circles) and 794 synthetic samples (blue circles) with the respective cell count indicated by brightness, where darker colors indicate lower cell counts. Additionally, we include 394 natural samples without known cell counts (gray circles). Note that the circles get darker from left to right, so this direction (determined to be important by UMAP) directly corresponds to the cell count. At the same time, natural and synthetic samples are not separated, which would contradict a truly shared representation between the two types of images.

Uniform Manifold Approximation and Projection (UMAP) [20] computes a topology-preserving embedding and has recently established itself as the state of the art for dimensionality reduction. In the resulting embedding (see Figure 10) we see that synthetic and natural data occupy the same space. Moreover, by coloring according to cell counts, we see that an encoding's position is directly related to the number of cells present in the respective image. Again, this property is shared between natural and synthetic images.

4 DISCUSSION

With this architecture we do not only provide a workflow for offline image recognition of suspension cells, it also lays the foundation for automated cell counting of microfluidic cell cultivations with single-cell resolution obtained by live cell imaging and can be extended to continuously estimate cell counts in real-time for experimentation monitoring and prediction of additional relevant information (e. g., survivability of a culture, growth rate or doubling time etc.). Since the technique works independently of the actual cell size of the analyzed organism, it is possible to generate proxy data for completely different kinds of cells (e. g., bacterial) with slight adjustments to the synthetic data generator. To reduce the amount of hand crafting meta-parameters further, the Gaussian process regressor can be extended, to allow provision of an easy to use tool, to be operated by biologists as an end-to-end solution for cell counting in live cell imaging procedures.

The novel `Twin-VAE` presented in this paper can be further expanded, e. g., by implementing an interactive manual counting to improve performance similar to [1], which could decrease the training time or even improve the overall results. As the presumably biggest challenge to overcome, overlapping of cells can be tackled with the methods presented in [2]. Another approach is to extend on data augmentation instead of explicit regularization [13].

4.1 Limitations & Future Work

For cells that appear very small in the images, the architecture currently can not deliver low error estimations, especially if the cells are only few pixels big in the downscaled version of the images. For bright-field data, cells sticking to the walls are a bigger hindrance to precise estimations, since, unlike for phase-contrast cells, there often is little to no visual hint for the boundary between the chamber border and the cell membrane. These problems can presumably be overcome with relative ease by more image preprocessing, like gray-value equalization.

Due to the high number of hyperparameter choices combined with the on average 5.6 seconds per epoch training time, a full training cycle of 50 000 epochs takes around three days per parameter set, the Gaussian Process regression is currently infeasible to be used to optimize all hyperparameters. The architecture was trained on a shared cluster that consists of a total of 28 compute nodes, each with two Intel Xeon Silver 4114, 404 GB of RAM and eight NVIDIA Tesla V100 SXM2 32GB. It has proven useful to parallelize multiple runs and create a new range of hyperparameter options based on the best results of the previous iteration. The technique shows promise to reduce the number of training cycle iterations needed but currently requires alternating between automation and handcrafting.

REFERENCES

[1] Carlos Arteta, Victor Lempitsky, J. Alison Noble, and Andrew Zisserman. “Interactive Object Counting”. In: *Computer Vision – ECCV 2014*. Ed. by David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars. Vol. 8691. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 504–518. DOI: 10.1007/978-3-319-10578-9_33. URL: http://link.springer.com/10.1007/978-3-319-10578-9_33.

[2] Carlos Arteta, Victor Lempitsky, J. Alison Noble, and Andrew Zisserman. “Detecting overlapping instances in microscopy images using extremal region trees”. In: *Medical Image Analysis* 27 (2016), pp. 3–16. ISSN: 1361-8423. DOI: 10.1016/j.media.2015.03.002.

[3] Stuart Berg et al. “ilastik: interactive machine learning for (bio)image analysis”. In: *Nature Methods* 16.12 (2019), pp. 1226–1232. DOI: 10.1038/s41592-019-0582-9.

[4] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. “Algorithms for Hyper-Parameter Optimization”. In: *Advances in Neural Information Processing Systems*. Ed. by J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger. Vol. 24. Curran Associates, Inc., 2011. URL: https://proceedings.neurips.cc/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf.

[5] Roger Brent and Laura Boucheron. “Deep learning to predict microscope images”. In: *Nature Methods* 15.11 (2018), pp. 868–870. DOI: 10.1038/s41592-018-0194-9.

[6] Steven W. Chen, Shreyas S. Shivakumar, Sandeep Dcunha, Jnaneshwar Das, Edidiong Okon, Chao Qu, Camillo J. Taylor, and Vijay Kumar. “Counting Apples and Oranges With Deep Learning: A Data-Driven Approach”. In: *IEEE Robotics and Automation Letters* 2.2 (Apr. 2017), pp. 781–788. ISSN: 2377-3774. DOI: 10.1109/LRA.2017.2651944.

[7] Dino Di Carlo, Liz Y Wu, and Luke P Lee. “Dynamic single cell culture array”. In: *Lab on a Chip* 6.11 (2006), pp. 1445–1449. DOI: 10.1039/b605937f.

[8] Thorsten Falk et al. “U-Net: deep learning for cell counting, detection, and morphometry”. In: *Nature Methods* 16.1 (2019), pp. 67–70. DOI: 10.1038/s41592-018-0261-2.

[9] Christina Göpfert, Shai Ben-David, Olivier Bousquet, Sylvain Gelly, Ilya O. Tolstikhin, and Ruth Urner. “When can unlabeled data improve the learning rate?” In: *Conference on Learning Theory, COLT 2019, 25-28 June 2019, Phoenix, AZ, USA*. Ed. by Alina Beygelzimer and Daniel Hsu. Vol. 99. Proceedings of Machine Learning Research. PMLR, 2019, pp. 1500–1518. URL: http://proceedings.mlr.press/v99/gopfert19a.html.

[10] Jan Philip Göpfert, Christina Göpfert, Mario Botsch, and Barbara Hammer. “Effects of variability in synthetic training data on convolutional neural networks for 3D head reconstruction”. In: *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE. 2017, pp. 1–7. DOI: 10.1109/SSCI.2017.8285305.

[11] Alexander Grünberger, Wolfgang Wiechert, and Dietrich Kohlheyer. “Single-cell microfluidics: opportunity for bioprocess development”. In: *Current Opinion in Biotechnology* 29 (2014). Cell and Pathway Engineering, pp. 15–23. ISSN: 0958-1669. DOI: 10.1016/j.copbio.2014.02.008.

[12] Xin He, Kaiyong Zhao, and Xiaowen Chu. *AutoML: A Survey of the State-of-the-Art*. 2020. arXiv: 1908.00709.

[13] Alex Hernández-García and Peter König. *Data augmentation instead of explicit regularization*. 2018. arXiv: 1806.03852.

[14] Alex J. Hughes, Joseph D. Mornin, Sujoy K. Biswas, Lauren E. Beck, David P. Bauer, Arjun Raj, Simone Bianco, and Zev J. Gartner. “Quanti.us: a tool for rapid, flexible, crowd-based annotation of images”. In: *Nature Methods* 15.8 (2018), pp. 587–590. DOI: 10.1038/s41592-018-0069-0.

[15] Anastasia Ioannidou, Elisavet Chatzilari, Spiros Nikolopoulos, and Ioannis Kompatsiaris. "Deep Learning Advances in Computer Vision with 3D Data: A Survey". In: *ACM Comput. Surv.* 50.2 (Apr. 2017). ISSN: 0360-0300. DOI: 10.1145/3042064.

[16] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. *Synthetic data and artificial neural networks for natural scene text recognition*. Workshop on Deep Learning, Advances in Neural Information Processing Systems (NIPS); Palais des Congrès de Montréal, Montréal Canada. 2014. arXiv: 1406.2227.

[17] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2013. arXiv: 1312.6114.

[18] Martin Kolnik, Lev S Tsimring, and Jeff Hasty. "Vacuum-assisted cell loading enables shear-free mammalian microfluidic culture". In: *Lab on a chip* 12.22 (2012), pp. 4732–4737. DOI: 10.1039/C2LC40569E.

[19] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. *On the Variance of the Adaptive Learning Rate and Beyond*. 2020. arXiv: 1908.03265.

[20] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. "UMAP: Uniform Manifold Approximation and Projection". In: *The Journal of Open Source Software* 3.29 (2018), p. 861.

[21] Erick Moen, Dylan Bannon, Takamasa Kudo, William Graf, Markus Covert, and David Van Valen. "Deep learning for cellular image analysis". In: *Nature Methods* 16.12 (2019), pp. 1233–1246. DOI: 10.1038/s41592-019-0403-1.

[22] M. S. Rahman and M. R. Islam. "Counting objects in an image by marker controlled watershed segmentation and thresholding". In: *2013 3rd IEEE International Advance Computing Conference (IACC)*. 2013 3rd IEEE International Advance Computing Conference (IACC). 2013, pp. 1251–1256. DOI: 10.1109/IAdCC.2013.6514407.

[23] Muhammad Imran Razzak, Saeeda Naz, and Ahmad Zaib. "Deep Learning for Medical Image Processing: Overview, Challenges and the Future". In: *Classification in BioApps: Automation of Decision Making*. Cham: Springer International Publishing, 2018, pp. 323–350. ISBN: 978-3-319-65981-7. DOI: 10.1007/978-3-319-65981-7_12.

[24] Jason Riordon, Dušan Sovilj, Scott Sanner, David Sinton, and Edmond W. K. Young. "Deep Learning with Microfluidics for Biotechnology". In: *Trends in Biotechnology* 37.3 (2019), pp. 310–324. ISSN: 0167-7799. DOI: 10.1016/j.tibtech.2018.08.005.

[25] Deepak Babu Sam, Neeraj N. Sajjan, Himanshu Maurya, and R. Venkatesh Babu. "Almost Unsupervised Learning for Dense Crowd Counting". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.1 (July 17, 2019), pp. 8868–8875. ISSN: 2374-3468. DOI: 10.1609/aaai.v33i01.33018868.

[26] Andrew M Saxe, James L McClelland, and Surya Ganguli. "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks". In: *International Conference on Learning Representations*. 2013. arXiv: 1312.6120.

[27] Julian Schmitz, Thomas Noll, and Alexander Grünberger. "Heterogeneity Studies of Mammalian Cells for Bioproduction: From Tools to Application". In: *Trends in Biotechnology* 37.6 (2019), pp. 645–660. DOI: 10.1016/j.tibtech.2018.11.007. URL: https://www.cell.com/trends/biotechnology/abstract/S0167-7799(18)30317-2.

[28] Julian Schmitz, Sarah Täuber, Christoph Westerwalbesloh, Eric von Lieres, Thomas Noll, and Alexander Grünberger. "Development and application of a cultivation platform for mammalian suspension cell lines with single-cell resolution". In: *Biotechnology and Bioengineering* 118.2 (2021), pp. 992–1005. DOI: 10.1002/bit.27627. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/bit.27627. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/bit.27627.

[29] Edgar Schönfeld, Sayna Ebrahimi, Samarth Sinha, Trevor Darrell, and Zeynep Akata. *Generalized Zero- and Few-Shot Learning via Aligned Variational Autoencoders*. Apr. 5, 2019. arXiv: 1812.01784.

[30] Mingxing Tan and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: (2019), pp. 6105–6114. URL: http://proceedings.mlr.press/v97/tan19a.html.

[31] Axel Theorell, Johannes Seiffarth, Alexander Grünberger, and Katharina Nöh. "When a single lineage is not enough: Uncertainty-Aware Tracking for spatio-temporal live-cell image analysis". In: *Bioinformatics (Oxford, England)* 35.7 (2019), pp. 1221–1228. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/bty776.

[32] Karen Ullrich, Edward Meeds, and Max Welling. *Soft Weight-Sharing for Neural Network Compression*. 2017. arXiv: 1702.04008.

[33] Vladimír Ulman et al. "An objective comparison of cell-tracking algorithms". In: *Nature Methods* 14.12 (2017), pp. 1141–1152. DOI: 10.1038/nmeth.4473.

[34] Christopher K. I. Williams and Carl Edward Rasmussen. "Gaussian Processes for Regression". In: *Advances in Neural Information Processing Systems 8*. Ed. by D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo. MIT Press, 1996, pp. 514–520. URL: http://papers.nips.cc/paper/1048-gaussian-processes-for-regression.pdf.

[35] Weidi Xie, J. Alison Noble, and Andrew Zisserman. "Microscopy cell counting and detection with fully convolutional regression networks". In: *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization* 6.3 (May 4, 2018), pp. 283–292. DOI: 10.1080/21681163.2016.1149104.

[36] Yuanpu Xie, Fuyong Xing, Xiangfei Kong, Hai Su, and Lin Yang. "Beyond classification: Structured regression for robust cell detection using convolutional neural network". In: *Medical Image Computing and Computer-Assisted Intervention* (International conference on medical image computing and computer-assisted intervention). Springer. 2015, pp. 358–365. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5226438/.