SHAPR - An AI approach to predict 3D cell shapes from 2D microscopic images

Dominik Jens Elias Waibel[1,2,3], Niklas Kiermeyer[1,2], Scott Atwell[5], Ario Sadafi[1,2,4], Matthias Meier[5,*], Carsten Marr[1,2,*]

[1]Institute of AI for Health, Helmholtz Zentrum München - German Research Center for Environmental Health, Neuherberg, Germany
[2]Institute of Computational Biology, Helmholtz Zentrum München - German Research Center for Environmental Health, Neuherberg, Germany
[3]Technical University of Munich, School of Life Sciences, Weihenstephan, Germany
[4]Computer Aided Medical Procedures, Technical University of Munich, Munich, Germany
[5]Helmholtz Pioneer Campus, Helmholtz Zentrum München - German Research Center for Environmental Health, Neuherberg, Germany
*email: matthias.meier@helmholtz-muenchen.de, carsten.marr@helmholtz-muenchen.de

Keywords: 3D shape prediction; single-cell morphology; AI; autoencoder; adversarial learning

Reconstruction of shapes, forms, and sizes of three-dimensional (3D) objects from two-dimensional (2D) information is one of the most complex functions of the human brain. It also poses an algorithmic challenge and at present is a widely studied subject in computer vision. We here focus on the single cell level and present a neural network-based SHApe PRediction autoencoder SHAPR that accurately reconstructs 3D cellular and nuclear shapes from 2D microscopic images and may have great potential for application in the biomedical sciences.

Recording single cells in three dimensions (3D) is prohibitively time-consuming as it requires stacking scores of two-dimensional (2D) images. This raises the question of how to optimally trade-off between throughput and resolution in space and time. Deep learning-based solutions for predicting 3D object shapes from photographs exist, creating meshes[1,2], voxels[3], or point clouds[4] for airplanes, cars, and furniture, but they cannot be directly translated to fluorescence microscopy for several reasons. Fluorescence microscopy imaging is fundamentally different from real-world photographs in terms of color, contrast, and object orientation. Unlike the shapes of cars or furniture that might vary due to differing photographic viewpoints, the shapes of single cells are similar but never the same, and it is often not feasible to image the same cell from different angles. Existing computer vision algorithms have been trained on tens of thousands of photographs where synthetic 3D models are available[5–7]. However, to the best of our knowledge no established synthetic models exist in the biomedical domain, and the number of potential training images is considerably reduced.

We addressed these problems with SHAPR, a deep learning network algorithm that combines a 2D encoder for feature extraction from 2D images with a 3D decoder to predict 3D shapes from a latent space representation (Fig. 1a and Supplementary Fig. 1). For proof of concept, we predicted cell shapes using a recently published library detailing 3D red blood cell shapes (n = 825 cells)[8]. Each cell shape was reconstructed from 68 confocal images with a z-resolution of 0.3µm. Using a 2D image intersecting the red blood cell at the center slice and the corresponding segmentation as input, SHAPR was trained by minimizing binary cross-entropy and dice loss between the true and the predicted 3D red blood cell shape (Fig. 1b and Methods). To increase SHAPR's prediction accuracy, a discriminator model (Fig. 1b) was trained to differentiate between true and predicted 3D cell shapes. SHAPR and the discriminator were trained until the predicted cell shape converged to an optimum. In each of the five cross-validation runs, 495 (60%) red blood cells from the library were used for training and 165 (20%) for intermediate validation during training. During testing, we predicted the 3D shapes of

165 (20%) previously unseen red blood cells. The results demonstrate that SHAPR is able to predict single red blood cell 3D shapes: while non-complex morphologies from red blood cells with a biconcave discoid shape (stomatocyte-discocyte-echinocyte (SDE) shape class) were predicted with low volume error, more complex shapes with irregular protrusions or cavities, normally seen in knizocytes and acanthocytes, were predicted with larger errors (Fig. 1c). We compared this cell shape prediction to two naïve 3D models, i.e. a cylindrical and an ellipsoid fit (see Methods). Compared with these models, the intersection over union in 3D is significantly higher for SHAPR (Fig. 1d). Furthermore, SHAPR predictions exhibited a lower volume error, 2D surface area error, and surface roughness error compared with the two naïve models (Fig. 1d).

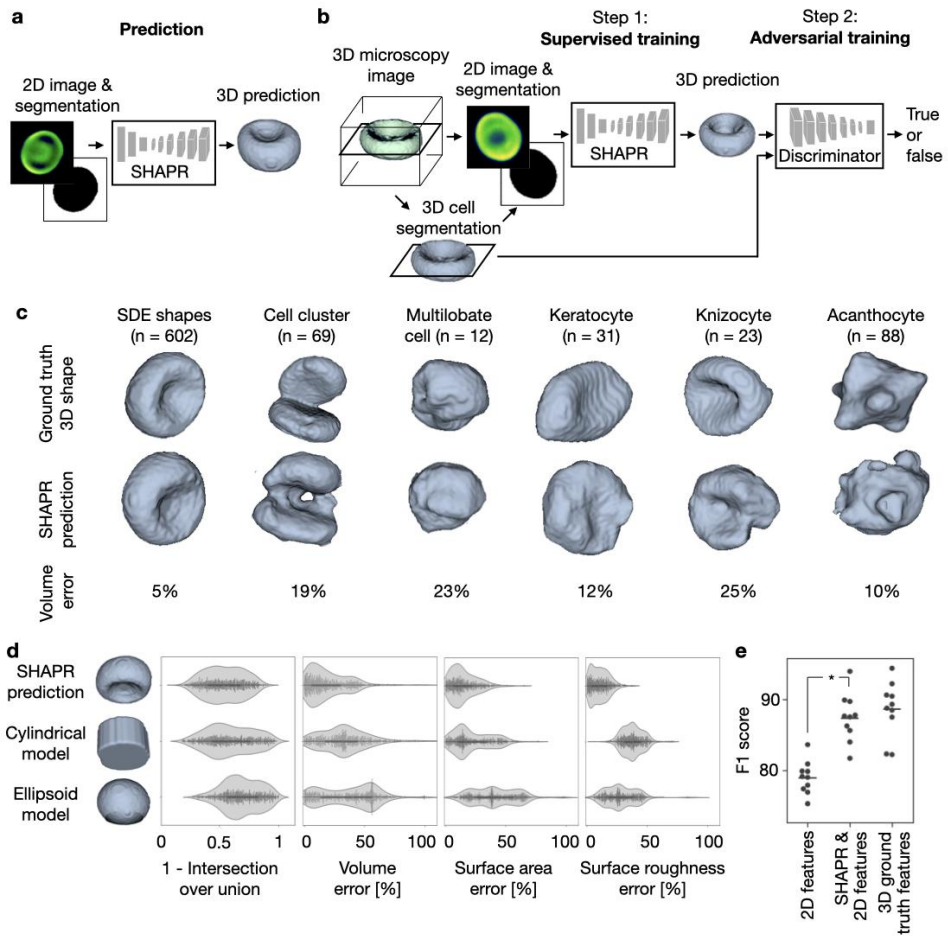


Fig. 1 | SHAPR predicts 3D cell shapes from 2D microscopic images more accurately than naïve models and improves shape classification significantly. a, SHAPR consists of an encoder for embedding 2D images into a 128-dimensional latent space and a decoder for reconstructing 3D cell shapes from the latent space representations. **b**, Two-step training approach: during step 1, SHAPR was trained in a supervised fashion with 2D fluorescent confocal cell microscopy images and their corresponding binary segmentations from a red blood cell library. During step 2, SHAPR was fine-tuned with a discriminator challenging its cell shape predictions. **c**, Example predictions for a set of red blood cells representing six different classes. The SDE shape class combines spherocytes, stomatocytes, discocytes, and echinocytes. **d**, The intersection over union is significantly higher for SHAPR (63%±12%) as compared to two naïve models ($IoU_{cylindrical}$ = 30±28, $p_{cylindrical} = 5.9 \times 10^{-74}$; and $IoU_{ellipsoid}$ =54±13; $p_{ellipsoid} = 9.7\times10^{-122}$, n = 825, paired Wilcoxon signed-rank test). Volume, surface area, and roughness error are significantly reduced. **e**, Random forest-based red blood cell classification is significantly improved when morphological features extracted from SHAPR predicted cell shapes are added to features derived from 2D images (p=0.005, paired Wilcoxon signed-rank test, n = 825).

Simionato et al.[8] classified red blood cells into six morphological categories (Fig. 1c) based on their 3D morphology. Can SHAPR predictions improve such downstream classification tasks? To investigate this, we extracted 126 morphological features, 11 features from an additionally predicted object mesh, object moments up to third order, correlation and dissimilarity of gray level co-occurrence matrices, and 64 Gabor features (see Methods for more details) from each predicted 3D cell shape. Using random forest[9], we classified each blood cell into one of the six classes and compared SHAPR's performance with the 3D ground truth features and a 2D baseline, where only features derived from the 2D image and segmentation were used (see Methods). As expected, classification based on ground truth features led to the highest F1 score (Fig. 1e; 88.6%±3.7%). Strikingly, enriching 2D features with SHAPR derived features performed significantly better (F1 = 87.4%±3.1%, mean±std.dev., n=10 cross-validation runs) than using 2D features only (F1 = 79.0%±2.2%) in a tenfold cross-validation (Fig. 1e, p=0.005, paired Wilcoxon signed-rank test).

Predicting shapes from 2D planes close to the cell's center of mass does not accurately reflect the complex nature of 3D cell shape. Therefore, we challenged SHAPR with the task of predicting cell nuclei shapes from confocal z-stacks containing fluorescence counterstained nuclei from human-induced pluripotent stem cells (iPSCs) cultured in a spheroidal aggregate. In order to generate the ground truth data, all cell nuclei were manually segmented in 3D (Fig. 2a). SHAPR was provided with one input 2D image slice taken at an aggregate depth of 22 μm (Fig. 2b) and its the corresponding segmentation. Nuclei were thus cut at random heights, leading to a variety of segmented areas, markedly complicating the prediction of 3D shapes (Fig. 2c). Following this, we trained five SHAPR models with a total of 887 nuclei from 2D images of six iPSC-derived aggregates during cross-validation. Predictions were compared to cylindrical and ellipsoid fits, as described above. Again, the intersection over union was significantly higher for SHAPR (Fig. 2d; IoU_{SHAPR} = 46±16 versus $IoU_{Cylindrical}$ = 41±19, $p = 1.3\times10^{-51}$, and $IoU_{Ellipsoid}$ = 31±16, $p = 2.8\times10^{-135}$, n = 887, paired Wilcoxon signed-rank test) compared to the naïve models. More importantly, SHAPR predictions were also closer to the true nuclei shapes in terms of volume and surface area compared with cylindrical and ellipsoid model predictions (Fig. 2d). To determine how much information our model had learned about nuclear shape, we compared the 2D segmentation area with the volume of the ground truth, the SHAPR predictions, and the cylindrical and ellipsoid fits (Fig. 2e). As expected, the cylindrical and ellipsoid models were simply extrapolating area to volume monotonically, while the ground truth suggested a more complex relationship. SHAPR was able to infer the complexity; i.e., small segmentation areas (<200 pixels) do not emerge from minuscule nuclei but from slices at the nuclear edge. Notably, our model could only obtain a high intersection over union scores for slices close to the center of mass, in contrast to volume and surface error (Fig. 2f). This suggests that it has learned to predict volume and surface correctly, but not if a nucleus is cut in its upper or the lower half.

In conclusion, while SHAPR did not reach the ground truth shape classification accuracy for red blood cells, it was able to retrieve real-world 3D information and outperformed naïve 3D shape fitting models. Predicting 3D shapes from 2D images thus offers a simple way to reduce imaging time and data storage while retaining morphological details. Trained SHAPR models can enable efficient predictions of single-cell volume distributions and density, can screen organoids, or even identify outlier events. In future applications, SHAPR could be used to enhance label-free transcription factor prediction[10,11] or perform single-cell classification for diagnostic purposes in computational pathology. Going beyond single-cell shape prediction, our neural network approach using SHAPR may be extended to other biological structures, including organelles and proteins, and may increase the efficiency of biomedical imaging in multiple domains.

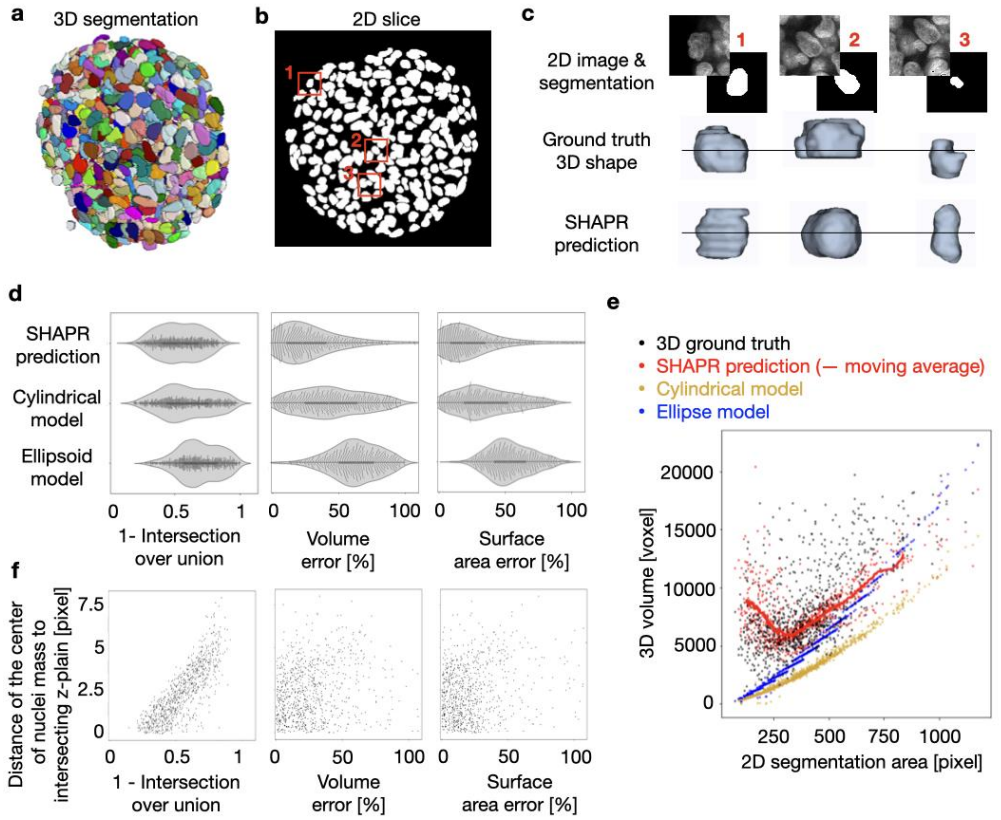


Fig. 2 | SHAPR learns fundamental 3D shape properties of human-induced pluripotent stem cell (iPSC) nuclei from a single 2D slice. a, Representative image of a segmented human-iPSC derived 3D cell culture with fluorescently stained nuclei. In order to generate ground truth data, six 3D cell cultures were manually segmented. **b,** 2D nuclei segmentation from a single slice at 22µm depth. **c**, 2D segmentation areas and fluorescent image intensities varied considerably with the position of the intersecting slice. Arrows indicate the distance to the nucleus center of mass. **d,** SHAPR predictions outperform cylindrical and ellipsoid models in terms of intersection over union, volume, and surface area error. **e,** While the cylindrical and ellipsoid models are only able to extrapolate volumes in a naïve manner, SHAPR learned complex, non-linear relationships between the 2D segmentation area and the 3D volume of a nucleus. **f,** While the intersection over union decreased with distance to the nucleus center of mass, volume, and surface predictions were unaffected.

Methods

SHAPR

Our SHApe PRediction algorithm SHAPR S consists of an encoder and a decoder with parameters θ (see Supplementary Figure 1a) that transforms an 2D input $i \in I$, which is a 2D fluorescent image and a corresponding binary mask (see Fig. 1a), to a binary 3D output p :

$$p = S(i; \theta).$$

A discriminator D with parameters τ tries to distinguish if a 3D shape x comes from SHAPR or from real data:

$$l = D(x; \tau).$$

Parameters θ and τ are learned during training when the objective function $\mathcal{L}$ is minimized:

$$\mathcal{L} = \mathcal{L}_{rec} + \alpha (\mathcal{L}_{adv} + \mathcal{L}_{dis}).$$

Here, $\mathcal{L}_{rec}$ is the reconstruction loss, $\mathcal{L}_{adv}$ is the adversarial loss, $\mathcal{L}_{dis}$ is the discriminator loss and α regulates the impact of adversarial and discriminator loss during training. The reconstruction loss tries to match the generated 3D output p with 3D ground truth y and is defined as:

$$\mathcal{L}_{rec}(\theta) = d(p, y) + b(p, y)$$

where $d(.,.)$and $b(.,.)$ are Dice loss and binary cross entropy loss, respectively (see Supplementary Figure 1a). The adversarial loss tries to match the distribution of generated shapes with the dataset ground truth:

$$\mathcal{L}_{adv}(\theta) = \mathbb{E}_{i \in I} \log(1 - D(S(i; \theta), \tau).$$

The discriminator loss is defined by:

$$\mathcal{L}_{dis}(\tau) = \mathbb{E}_{y \in Y} \log D(y; \tau) + \mathbb{E}_{i \in I} \log(1 - D(S(i; \theta), \tau).$$

In the following implementation details are explained in further details: The encoder is built of three blocks (see Supplementary Figure 1a). Each block contains two 3D convolutional layers with a kernel size of (1,3,3), followed by a batch normalization, a dropout, and a max pooling layer to downsample convolutions. The last activation function of the encoder is a sigmoid. The decoder consists of seven convolutional blocks and each of them contains two 3D convolutional layers, followed by batch normalization, a dropout, and a 3D transpose convolutional layer for upsampling. We upsample the z-dimension seven times and the x-y dimensions 3 times in an alternating fashion. The discriminator D consists of five convolutional layers with a kernel size of (3,3,3) followed by an average-pooling in each dimension and two dense layers, one with 128 and one with 1 unit, followed by a sigmoid activation function, which outputs a binary label for each 3D input shape. The regularization parameter α is a step function starting with 0 so the model is trained using the reconstruction loss alone. After 30 epochs or if the validation loss has not improved for 10 epochs, α switches to 1. From then on, SHAPR is training in an adversarial fashion. Model was implemented using Tensorflow and Keras[13,14].

Training parameters

Five independent models were trained on both datasets in a round-robin fashion, so each input image was contained in the test set exactly once. Adam optimizer [15] with an initial learning rate of $1*10^{-3}$, beta1 of 0.9, and beta2 of 0.999 was used.

For data augmentation, training data was randomly flipped horizontally and vertically and rotated with a chance of 33% for each augmentation to be applied on each data point.

To obtain a binary image all SHAPR predictions are thresholded at 126, as their pixel values range from 0 to 255.

Evaluation metrics

For comparison with different models, five metrics are used: Intersection over union (IoU), Relative voxel error, relative volume error, relative surface error and relative surface roughness error. With Y being the ground truth and P the prediction, these are defined as:

$$IoU(Y,R) = \frac{Y \cap P}{Y \cup P}$$

$$Relative\ voxel\ error\ (Y,P) = \frac{1}{NMK} \sum_{x=1}^{N} \sum_{y=1}^{M} \sum_{z=1}^{K} \left| \frac{Y_{xyz} - P_{xyz}}{Y_{xyz}} \right|$$

$$Relative\ volume\ error\ (Y,P) = \frac{volume(Y) - volume(P)}{volume(Y)}$$

$$Relative\ surface\ error\ (Y,P) = \frac{surface(Y) - surface(P)}{surface(Y)}$$

$$Relative\ surface\ roughness\ error\ (Y,P) = \frac{surface\ roughness\ (Y) - surface\ roughness(P)}{surface\ roughness\ (Y)}$$

where N, M, and K are the bounding box sizes and $volume(.)$ and $surface(.)$ yield the volume by counting non-zero voxels:

$$Volume(P) = \sum_{x=1}^{N} \sum_{y=1}^{M} \sum_{z=1}^{K} \mathbb{1}(P_{xyz} > 0),$$ where $\mathbb{1}$ is the indicator function.

and surface area by counting all voxels on the surface of a given 3D binary shape:

$$Surface(P) = \sum_{x=1}^{N} \sum_{y=1}^{M} \sum_{z=1}^{K} \mathbb{1}(P_{xyz} = \partial P)$$, where ∂P is defined as the surface of P.

The function $surface\ roughness(.)$ is defined as:

$$Surface\ roughness\ (P) = \sum_{x=1}^{N} \sum_{y=1}^{M} \sum_{z=1}^{K} \left| P_{xyz} - P_{xyz}^{gaussian} \right|$$

with:

$$P_{xyz}^{gaussian} = \frac{1}{(\sqrt{2\pi}\ \sigma)^3} \exp\left(-\frac{x^2 + y^2 + z^2}{2\sigma^2}\right), \quad \forall x, y, z \in P$$

Feature extraction

We extract 126 features from each 3D shape, comprising volume, surface, shape index, roughness, convexity, and gabor features with NumPy and the Skimage toolbox (Supplementary Table 3)[16]. Eleven features are derived by describing the object with a mesh consisting of faces and vertices. The mesh is calculated using the marching cubes algorithm in python's Skimage toolbox[16,17], resulting in faces and vertices. The surface, 19 mesh principals, the first nine mesh inertia eigenvalues were calculated using trimesh[18]. The moments of inertia represent the spatial distribution of mass in a rigid 3D shape. This depends on the 3D shapes mass, size, and shape. The moment of inertia is calculated as the angular momentum divided by the angular velocity around a principal axis. We also calculate the objects' moments up to the third order, correlation and dissimilarity of the gray level co-occurrence matrices, and one gabor feature for each z-slice, resulting in 64 gabor features all using the Skimage toolbox[16].

From 2D segmentations, we extract 9 features. These are the mean pixel value, area, boundary length, boundary roughness, convexity, two moments, and three gabor features and 5 features from the 2D microscopy images (see Supplementary Table 3).

Feature-based classification

To establish the baseline for the feature-based classification, we extract 9 morphological features (mean pixel value, area, boundary length, boundary roughness, convexity, two moments, and three gabor features) and 5 features from the 2D microscopy images that have been multiplied with the respective segmentation mask to reduce noise (mean and standard deviation of the pixel intensity, one gabor feature, and the correlation and dissimilarity of the gray level co-occurrence matrices). In total, we extract 15 features from each of the paired 2D images and 2D segmentations. We compared random forest, decision tree, K-nearest-neighbors, linear discriminant analysis, naïve Bayes, and support vector machine classifiers using the Sklearn toolbox [19]. The random forest classifier with 1000 estimators performed best.

Prior to the classification we oversample the training data to compensate for class imbalance, and normalize all features by subtracting their mean and dividing by their standard deviation. Prior to the tenfold cross-validation classification, we trained one random forest model to investigate feature importance. We found that accuracy increases if we remove features with an importance lower than 0.005 (see Supplementary Fig. 1c). We do not only achieve an overall higher F1 score using ShapeAEs predictions, but increase the number of true positives for four of the six classes, while for two classes we obtain the same scores (Supplementary Fig. 1a).

Datasets

Red blood cells

We use 825 publicly available 3D images of red blood cells[8] of size (64,64,64) voxels, each assigned to one of the following six classes: SDE shapes, cell cluster, multilobate, keratocyte, knizocyte, and acanthocyte. Spherocytes, stomatocytes, discocytes, and echinocytes are combined into the SDE shapes class, which is characterized by the stomatocyte–discocyte–echinocyte transformation[20]. The other classes' shapes occur in samples from patients with blood disorders or other pathologies.

The number of cells in each class were: 602 for SDE shapes (93 spherocytes, 41 stomatocytes, 176 discocytes, and 292 echinocytes), 69 for cell clusters, 12 multilobates, 31 keratocytes, 23 knizocytes, and 88 acanthocytes. Red blood cells were drawn from 10 healthy donors and 10 patients with hereditary spherocytosis via finger-prick blood sampling and then fixed. Thereafter the cells were imaged with a confocal microscope and then manually classified. We extracted the 2D image from the central slide of each 3D image and segmented it by thresholding. Also, the 3D ground truth was obtained by thresholding.

Human pluripotent stem cells derived 3D cultures

Six human induced pluripotent stem cell-derived 3D cultures were imaged with a Zeiss LSM 880 Airyscan inverted confocal microscope with DAPI as a nuclear counterstain. Full 3D stacks were acquired using a 20X objective with a resolution of 0.25µm/pixel and a distance between slices of 1µm. We rescaled the images to 0.5µm/pixel in x-y dimension. From the center z-slice of each 3D cell culture, we manually segment and isolate all nuclei in 2D and the corresponding 3D nuclei, resulting in a dataset of 887 paired 2D/3D single-cell shapes of shapes (64,64) pixel and (64,64,64) pixel. We interpolated the single 3D nuclei to an isotropic spacing of 0.5µm/pixel. While for the red blood cell dataset we could expect to cut each single cell roughly in its middle, in this dataset the nuclei are cut at any possible z-position.

Data availability

The datasets are available at:
https://hmgubox2.helmholtz-muenchen.de/index.php/s/YAds7dA2TcxSDtr

Code availability

SHAPR is available as a well documented, pip installable package together with commented analysis scripts at https://github.com/marrlab/SHAPR

Acknowledgments

We thank Mohammad Mirkazemi and Ron Fechnter for reviewing our code. We thank Ario Sadafi, Sophia Wagner, Tingying Peng, Sayedali Shetab Boushehri, and Matthias Hehr for discussions and for contributing their ideas. We thank Marius Bäuerle, Ario Sadafi and Valerio Lupperger for their feedback on the figures and manuscript.

Author contributions

DW and NK implemented code and conducted experiments. DW, NK, and CM wrote the manuscript with AS, SA and MM. DW created figures and the main storyline with CM. SA and MM provided the 3D cell culture dataset and ideas. CM supervised the study. All authors have read and approved the manuscript.

Additional Information

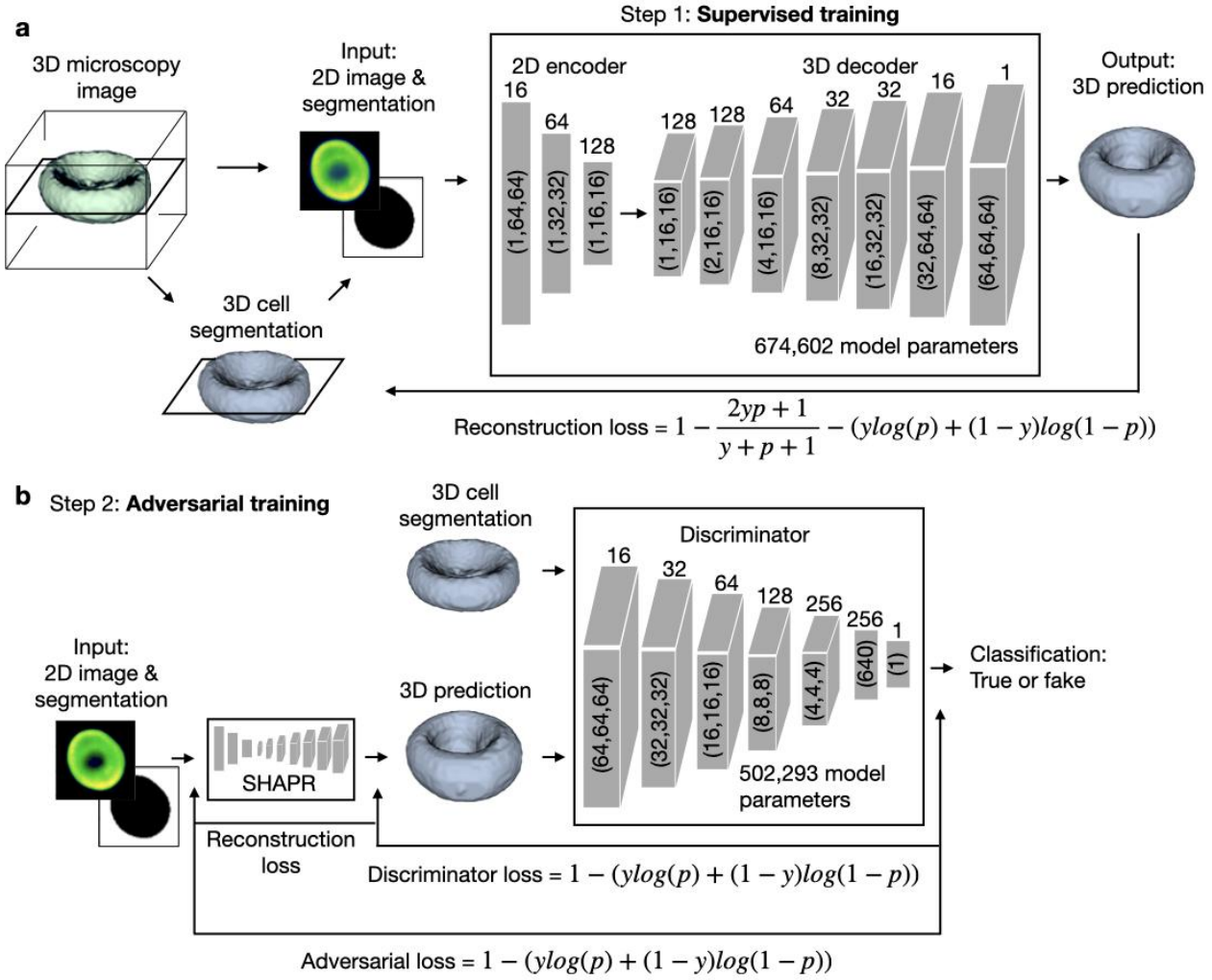
Competing interests

The author(s) declare no competing interests.

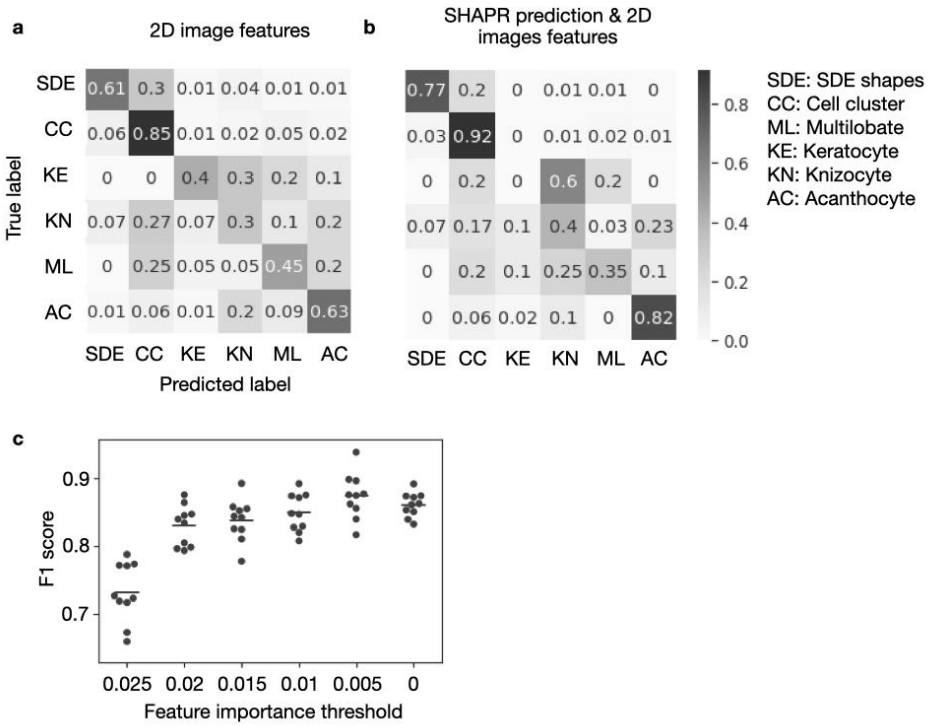
Funding

CM has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 866411).

Supplementary material



Supplementary Figure 1 | **a**, SHAPR consists of a 2D encoder, which embeds 2D images into a 128-dimensional latent space, and a 3D decoder, which reconstructs 3D shapes from the latent space representation. To train SHAPR we segment 3D microscopy images (we show an exemplary single red blood cell). We pair a 2D segmentation with the microscopy image of the same slice to enter the encoder as input. During supervised training (Fig. 1, step 1), we minimize the reconstruction loss (see Methods), which is the sum of the Dice loss and the binary cross entropy loss between the 3D segmentations y and SHAPR predictions p. For an input image of 64 x 64 pixels, we provide the pixel sizes for each layer in the gray boxes and the filter sizes on top of each box. **b**, In the second step, we fine-tune SHAPR by adding a discriminator. The discriminator is trained to differentiate between SHAPR output p ground truth segmentation r and minimize the adversarial loss. It thereby challenges SHAPR to output realistic 3D objects.



Supplementary Figure 2 | **a,b,** Confusion matrix for red blood cell classification using 2D images features (a) and 2D images features combined with features from SHAPR 3D prediction (b). SHAPR increases the mean true positive rate in a 5-fold cross-validation for 3 out of 5 classes. For keratocytes and knizocytes the mean true positive rate is the same. **c,** Our random forest trained on the combination of 3D SHAPR features, 2D image features, and 2D segmentation features yields the highest F1 score if we remove features that are less important than a threshold of 0.005 before retraining the random forest.

Supplementary Table 1 | Using SHAPR, we obtain a higher intersection over union, a lower volume, surface, and roughness estimation predicting the 3D shape of red blood cells compared to predicting the quantities using a cylindrical or ellipsoid fit (Fig. 2b). Best results are highlighted in boldface.

Mean±std.dev. [%]	3D red blood cell fit (n = 825)		
	SHAPR	Cylindrical model	Ellipsoid model
Intersection over union	**63±12**	30±28	54±13
Relative Volume error	**20±18**	33±22	37±23
Relative Surface area error	**15±11**	22±15	40±18
Relative surface roughness error	**11±7**	38±9	28±13

Supplementary Table 2 | Using SHAPR, we obtain a higher intersection over union, a lower volume, surface, and roughness estimation predicting the 3D shape of nuclei as compared to predicting the quantities using a cylindrical or ellipsoid fit (Fig. 3f). Best results are highlighted in boldface.

Mean±std.dev. [%]	3D nuclei cell fit (n = 887)		
	SHAPR	Cylindrical model	Ellipsoid model
Intersection over union	**46±16**	41±19	31±16
Relative Volume error	**33±41**	44±25	62±19
Relative Surface area error	**32±36**	36±22	54±16

Supplementary Table 3 | List of features extracted for different image modalities: For 3D images, a total of 126 features were acquired by using built-in functions of the python Skimage package. Since there are fewer features in 2D than in 3D, only 7 features were extracted from the binary mask and 5 features from the image.

3D features (126)	2D segmentation features (9)	2D image features (5)
Volume	Mean pixel value	Mean pixel value
Surface	Surface	Standard deviation of pixel value
Mean		
Shape index	Boundary of outline	Gabor feature
Roughness	Roughness	Gray-level co-occurrence dissimilarity and correlation (2 features)
Gaussian roughness	Convexity	
Convexity	Gray-level co-occurrence dissimilarity and correlation (2 features)	
3D boundary	Moment center row and center column (2 features)	
Gabor feature of z-projection		
Gray-level co-occurrence dissimilarity and correlation (2 Features)		
Mean, standard deviation, and maximum of z-projection (3 features)		
Convexity of central slices (3 features)		
Boundary of central slices (3 features)		
Inertia eigenvalues (3 features)		
Mean, median, and Standard deviation of triangular faces (calculated with Lewiner marching cubes algorithm) (3 features)		
Mean, median, standard deviation of vertices (calculated with Lewiner marching cubes algorithm) (3 features)		
Mesh volume, mesh convex hull volume (2 features)		
Number of vertices and faces (2 features)		
Moments (8 features)		
Mesh inertia eigenvalues (9 features)		
Mesh principals (12 features)		
Gabor filter of each z-slice (64 features)		

References

1. Wang, N. *et al.* Pixel2mesh: Generating 3d mesh models from single rgb images. in *Proceedings of the European Conference on Computer Vision (ECCV)* 52–67 (2018).
2. Gkioxari, G., Malik, J. & Johnson, J. Mesh r-cnn. in *Proceedings of the IEEE/CVF International Conference on Computer Vision* 9785–9795 (2019).
3. Choy, C. B., Xu, D., Gwak, J., Chen, K. & Savarese, S. 3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction. in *Computer Vision – ECCV 2016* 628–644 (Springer International Publishing, 2016).
4. Fan, H., Su, H. & Guibas, L. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017) doi:10.1109/cvpr.2017.264.
5. Chang, A. X. *et al.* ShapeNet: An Information-Rich 3D Model Repository. *arXiv [cs.GR]* (2015).
6. Sun, X. *et al.* Pix3d: Dataset and methods for single-image 3d shape modeling. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 2974–2983 (2018).
7. Xiang, Y., Mottaghi, R. & Savarese, S. Beyond PASCAL: A benchmark for 3D object detection in the wild. in *IEEE Winter Conference on Applications of Computer Vision* 75–82 (2014).
8. Simionato, G. *et al.* Red blood cell phenotyping from 3D confocal images using artificial neural networks. *PLoS Comput. Biol.* **17**, e1008934 (2021).
9. Breiman, L. Random Forests. *Mach. Learn.* **45**, 5–32 (2001).
10. Ounkomol, C., Seshamani, S., Maleckar, M. M., Collman, F. & Johnson, G. R. Label-free prediction of three-dimensional fluorescence images from transmitted-light microscopy. *Nat. Methods* (2018) doi:10.1038/s41592-018-0111-2.
11. Christiansen, E. M. *et al.* In Silico Labeling: Predicting Fluorescent Labels in Unlabeled Images. *Cell* **173**, 792–803.e19 (2018).
12. Mirza, M. & Osindero, S. Conditional Generative Adversarial Nets. *arXiv [cs.LG]* (2014).
13. Abadi, M. *et al.* TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv [cs.DC]* (2016).
14. Chollet, F. & Others. Keras. *https://keras.io* (2015).
15. Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. *arXiv:1412. 6980 [cs]* (2014).
16. Boulogne, F., Warner, J. D. & Neil Yager, E. Scikit-image: Image processing in Python. *J. PeerJ* **2**, 453 (2014).

17. Lewiner, T., Lopes, H., Vieira, A. W. & Tavares, G. Efficient Implementation of Marching Cubes' Cases with Topological Guarantees. *Journal of Graphics Tools* **8**, 1–15 (2003).
18. Basic Installation — trimesh 3.9.24 documentation. https://trimsh.org/.
19. Varoquaux, G. *et al.* Scikit-learn. *GetMob. Mob. Comput. Commun.* **19**, 29–33 (2015).
20. Chen, M. & Boyle, F. J. An Enhanced Spring-Particle Model for Red Blood Cell Structural Mechanics: Application to the Stomatocyte-Discocyte-Echinocyte Transformation. *J. Biomech. Eng.* **139**, (2017).