International Journal of Computer Assisted Radiology and Surgery
https://doi.org/10.1007/s11548-019-02010-3

ORIGINAL ARTICLE

Hyperparameter optimization for image analysis: application to prostate tissue images and live cell data of virus-infected cells

Christian Ritter[1] · Thomas Wollmann[1] · Patrick Bernhard[1] · Manuel Gunkel[2] · Delia M. Braun[3] · Ji-Young Lee[5,6] · Jan Meiners[4] · Ronald Simon[4] · Guido Sauter[4] · Holger Erfle[2] · Karsten Rippe[3] · Ralf Bartenschlager[5,6] · Karl Rohr[1]

Received: 16 February 2019 / Accepted: 30 May 2019

Abstract

Purpose Automated analysis of microscopy image data typically requires complex pipelines that involve multiple methods for different image analysis tasks. To achieve best results of the analysis pipelines, method-dependent hyperparameters need to be optimized. However, complex pipelines often suffer from the fact that calculation of the gradient of the loss function is analytically or computationally infeasible. Therefore, first- or higher-order optimization methods cannot be applied.

Methods We developed a new framework for zero-order black-box hyperparameter optimization called HyperHyper, which has a modular architecture that separates hyperparameter sampling and optimization. We also developed a visualization of the loss function based on infimum projection to obtain further insights into the optimization problem.

Results We applied HyperHyper in three different experiments with different imaging modalities, and evaluated in total more than 400.000 hyperparameter combinations. HyperHyper was used for optimizing two pipelines for cell nuclei segmentation in prostate tissue microscopy images and two pipelines for detection of hepatitis C virus proteins in live cell microscopy data. We evaluated the impact of separating the sampling and optimization strategy using different optimizers and employed an infimum projection for visualizing the hyperparameter space.

Conclusions The separation of sampling and optimization strategy of the proposed HyperHyper optimization framework improves the result of the investigated image analysis pipelines. Visualization of the loss function based on infimum projection enables gaining further insights on the optimization process.

Keywords Microscopy image analysis · Hyperparameter optimization · Optimization framework · Visualization

Introduction

Automatic analysis of microscopy data typically requires complex pipelines comprising multiple methods to solve different image analysis tasks (e.g., cell segmentation [21], cell tracking [27], particle tracking [3], track analysis [7,19,24], and image registration [25]). However, most methods suffer from determining application-dependent hyperparameters to obtain the best performance. More generally, medical decision support systems evaluating the patient status in the clinic can highly depend on specific hyperparameters [5]. Also the quality and performance of image-guided interven-

✉ Christian Ritter
christian.ritter@bioquant.uni-heidelberg.de

Thomas Wollmann
thomas.wollmann@bioquant.uni-heidelberg.de

Karl Rohr
k.rohr@dkfz-heidelberg.de

1 Biomedical Computer Vision Group, BioQuant, IPMB, University of Heidelberg and DKFZ, Im Neuenheimer Feld 267, Heidelberg, Germany

2 High-Content Analysis of the Cell (HiCell) and Advanced Biological Screening Facility, BioQuant, University of Heidelberg, Im Neuenheimer Feld 267, Heidelberg, Germany

3 Division of Chromatin Networks, DKFZ and BioQuant, Im Neuenheimer Feld 267, Heidelberg, Germany

4 Department of Pathology, University Medical Center Hamburg-Eppendorf, Martinistraße 52, Hamburg, Germany

5 Department of Infectious Diseases, Molecular Virology, University Hospital Heidelberg, Im Neuenheimer Feld 344, Heidelberg, Germany

6 German Center of Infection Research, Heidelberg Partner Site, Heidelberg, Germany

Published online: 08 June 2019

tion [4] generally highly depends on hyperparameters. For complex analysis pipelines, manual optimization of hyperparameters is generally very time-consuming and difficult for a high-dimensional hyperparameter space. Thus, automated optimization is required. However, computation of the gradient of the loss function is analytically or computationally infeasible, which prevents the use of first- or higher-order optimization methods. This limitation can be overcome by using zero-order optimization also known as black-box optimization [28], which does not require gradient information of the loss function. Black-box optimization uses only a limited number of evaluations (hyperparameter configurations) to determine a (local) optimum of the generally non-convex optimization problem.

In this work, we propose a framework for black-box hyperparameter optimization for biomedical image analysis pipelines called HyperHyper. This framework has several advantages compared to existing hyperparameter optimization frameworks such as Google Vizier [10], Sherpa [11], Auto-WEKA [26], Spearmint [23], and Hyperopt [14]. In Table 1, an overview of key features of most popular existing optimization frameworks and HyperHyper is provided. The table extends the comparison in [11] and also includes updated information about the frameworks. Existing frameworks lack certain features (e.g., modular optimizer, job wrapper, and integrated scheduler), which are essential to optimize complex image analysis pipelines using different computing paradigms and environments. The pipelines for biomedical image analysis typically include a large variety of hyperparameters, which increase the complexity of the hyperparameter space and make optimization challenging. To determine optimal solutions, our HyperHyper framework employs more than 40 different optimization methods, while existing frameworks include significantly less methods (e.g., up to five methods as in Table 1). The high number of optimizers in HyperHyper was realized by separation of hyperparameter sampling and optimization strategy. Except Auto-Weka, all frameworks in Table 1 can operate in a distributed computing environment. However, the frameworks (except HyperOpt) do not include an integrated scheduler. To optimize hyperparameters on different cluster computing infrastructures, we implemented an integrated scheduler which is advantageous when deploying image analysis methods on heterogeneous computing infrastructures.

To demonstrate the suitability of our optimization framework HyperHyper, we conducted three different experiments. In the first experiment, we evaluate HyperHyper using different optimizers for two different complex pipelines to segment cell nuclei in prostate tissue microscopy images. This experiment shows the impact of our proposed separation of sampling and optimization strategy to obtain a better performance of the image analysis pipelines. With the second experiment, we evaluate HyperHyper to detect the hepatitis C virus (HCV) nonstructural protein 5A (NS5A) in live cell microscopy images. Again, we demonstrate the effect of separating the sampling and optimization strategy. Furthermore, we show how our proposed visualizations of the loss function can help to assess the used optimizers. In the third experiment, we consider an extension of the pipeline for detecting HCV proteins by image pre-processing. We show how a visualization based on infimum projection of a high-dimensional loss function can gain insights into the optimization problem.

Material and methods

Overview of HyperHyper

The computing environments in the scientific community are very heterogeneous due to different computing paradigms (e.g., HPC, Cloud, Mainframe) and multiple programming languages. Moreover, the use cases for optimization of hyperparameters vary a lot. Incorporation of prior knowledge about the hyperparameters from domain knowledge or previous optimizations can significantly reduce the search space. Therefore, a hyperparameter optimization framework should be designed to be environment agnostic (e.g., programming language, compute infrastructure), extendable through modularity, and should allow incorporating prior knowledge. Moreover, flexible distribution of the computation should be supported since evaluation of hyperparameter configurations is often computational expensive. In addition, visualization of the optimization process and hyperparameter space is important to reveal insights about the optimization problem.

We propose the black-box optimization framework HyperHyper for distributed computing. This framework subdivides hyperparameter optimization in a hyperparameter space definition, a general optimizer containing a hyperparameter candidate sampler and optimization strategy, and an evaluation loop (Fig. 1). The candidate sampler and optimization strategy can be selected from a model zoo to design an optimizer for a specific application. In addition, the hyperparameter space definition incorporates prior distributions, bounds, and the sampling resolution. The candidate sampler and optimization strategy can exploit the structure of the hyperparameter space to improve convergence of the optimization. To find the global optimum, Grid Search can be used. Moreover, by design the execution of the evaluation loop can be performed highly distributed and is programming language agnostic. We integrated modules for monitoring and visualization to analyze the optimization problem. These visualizations including an infimum projection can reveal insights into, for example, the performance of the optimization process and the dependencies of the hyperparameters.

Table 1 Comparison of different hyperparameter optimization frameworks

Feature	Google Vizier	Sherpa	Auto-WEKA	Spearmint	HyperOpt	HyperHyper
Number of optimization methods	3	5	1	5	5	**> 40**
Modular optimizer	No	No	No	No	No	**Yes**
Job wrapper	No	No	No	No	No	**Yes**
Distributed	**Yes**	**Yes**	No	**Yes**	**Yes**	**Yes**
Integrated scheduler	No	No	No	No	**Yes**	**Yes**
Early stopping	**Yes**	**Yes**	No	No	No	**Yes**
Visualization	**Yes**	**Yes**	No	No	No	**Yes**

The best results are highlighted in bold

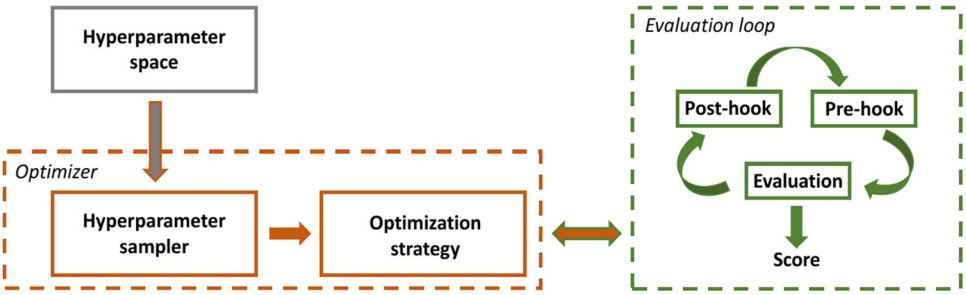

Fig. 1 Schematic representation of HyperHyper software architecture with modular structure

Optimizer

To perform optimization, constraints on the hyperparameter space have to be specified. This includes the bounds and hierarchy of each parameter, the sampling resolution, and additional prior distributions (e.g., discrete or continuous uniform, Gaussian, log Gaussian, exponential distributions). In our experiments, we used pipelines that involve non-ordinal parameters. Therefore, we decided to choose optimizers which can handle variables without a natural order (Table 2). To create optimizers in HyperHyper as listed in Table 2, the sampling and optimization strategy can be selected from the model zoo.

The most naive optimization strategy is to perform Random Search (Random) by random sampling from the prior distributions. In Sequential Model-based Optimization (SMBO) like SMAC [12], a surrogate model is fitted to the best performing hyperparameters. We investigated SMAC with the original random forest (SMAC-RF), and with XGBoost [2] as surrogate model. We decided to use XGBoost, since it is currently one of the most popular decision tree-based models. Moreover, we investigated the Tree of Parzen Estimator (TPE), which performs a nonparametric density approximation of the best performing hyperparameter configurations [18]. Finally, we use Covariance Matrix Adaptation Evolution Strategy (CMA-ES), which is a generic population-based meta-heuristic based optimizer [9]. In CMA-ES feature sets are assumed as "genomes", which undergo evolutionary processes like selection, recombination, or mutation to increase the probability for sampling promising hyperparameter configurations.

HyperHyper evaluation loop

In the HyperHyper software architecture, the hyperparameter evaluation is decoupled from the optimization strategy. The hyperparameter evaluation is split into the pre-hook, the evaluation, and a post-hook (see Fig. 1). The pre-hook is used for preparation of the experiment by performing a single experiment with a specific set of hyperparameters based on the hyperparameter sampling and optimization strategy. The evaluation step calculates the performance of the current hyperparameter configuration using the desired objective function. The post-hook performs clean up operations. Due to the generic formulation of the evaluation loop, any concrete implementations can be used in this plug and play system. We investigated direct hooks for Python, job wrapper, and remote execution of workflows (e.g., Galaxy Imaging [1,30]). For scripts directly written in Python, entry points can be called directly by HyperHyper. A job wrapper wraps pipelines that can be called via command line. This approach is the most generic, since it can handle arbitrary programming languages. Finally, remote execution of workflows in workflow engines like Galaxy is useful to leverage already optimized third-party high-performance computing (HPC) or cloud infrastructure. The execution of the loop can be distributed using a central database for coordination and workers for actual execution. Moreover, the distributed optimization process can be monitored by retrieving status information

Table 2 Investigated optimizers and corresponding sampling and optimization strategies

Optimizer	Sampling strategy	Optimization strategy
Random Search	Random	–
TPE	Parzen estimator	–
CMA-ES	Multivariate normal	Evolutionary
SMAC-RF	Random	Random forest
SMAC-XGBoost	Random	XGBoost

from the central database. The workers can be scheduled to available HPC or cloud infrastructure using, for example, Nextflow [6]. This approach has the benefit, that it can leverage and even combine a vast variety of schedulers or cloud systems, even at multiple sites.

Experimental results

To showcase the HyperHyper optimization framework, we conducted three different experiments. First, we consider the segmentation of cell nuclei in prostate tissue slides using a clustering and a deep learning pipeline. Second, we study the detection of HCV proteins in live cell fluorescence microscopy images. In these two experiments, we show that the separation of sampling and optimization strategy improves the optimal solution. In the third experiment, we consider an extension of the pipeline for detection of HCV proteins by image pre-processing. In this experiment, we show how an infimum projection of the loss surface can provide insights into the optimization problem. In the following, the used hyperparameters that need to be optimized are highlighted in italics.

Experiment 1: segmentation of cell nuclei

In the field of histopathology, segmentation of cell nuclei in tissue microscopy images is a pivotal and essential task. We acquired 2D DAPI stained prostate tissue image slices using an Opera spinning disk confocal microscope at 60x magnification with a resolution of 107.7 nm × 107.7 nm. Cell nuclei segmentation is needed for telomere quantification on a single-cell basis. Due to image noise, variation of cell shape and image intensity, and low contrast, cell nuclei segmentation is challenging (Fig. 2) [29]. For analyzing high-content histological screening data, hyperparameters of image analysis methods need to be optimized, which is often very difficult or not feasible to perform manually due to the high dimensionality of the hyperparameter space. We applied HyperHyper in conjunction with two different segmentation pipelines to analyze tissue images of the prostate and investigate the suitability of our black-box hyperparameter optimization framework.

The first pipeline consists of K-means clustering after image smoothing by a Gaussian filter with σ_{Gauss}. The type of *cluster initialization* (random, k-means++) is optimized and the seed value is set to a fixed value to generate a deterministic segmentation pipeline. Subsequently, we applied median filtering and morphological closing of small holes. Then, we computed a *geometric feature* (e.g., major axis length, eccentricity) determined by optimization and for each cluster compared it to the mean of all clusters to assign the cluster to foreground. The segmented objects are thresholded regarding *area* (upper and lower threshold) and *solidity*. In summary, for this pipeline the hyperparameters σ_{Gauss}, *cluster initialization*, *geometric feature*, *area*, and *solidity* need to be optimized.

The second pipeline consists of a U-Net [21] with Adam optimizer [13] and early stopping. For the network, we used the same seed value for sampling the initial weights as for the K-means clustering pipeline for a fair comparison. Data augmentation was performed with image rotation, flipping, and elastic deformation. To discard small objects, we thresholded the geometric feature "area" with *area threshold*. In summary, for this pipeline the hyperparameters *learning rate*, *batch size*, and *area threshold* need to be optimized.

We applied HyperHyper for both pipelines using multiple optimizers namely Random, TPE, CMA-ES, SMAC-RF, and SMAC-XGBoost and performed a distributed optimization on 20 computer nodes. We evaluated the K-means clustering pipeline with 10 runs and 200 evaluations for each optimizer (total: 10.000 evaluations) and the U-Net pipeline with 1 run and 200 evaluations for each optimizer (total: 1.000 evaluations). To determine the global optimum, we applied Grid Search using 17.388 evaluations. The tissue images have different sizes and were divided into 256 × 256 pixel image patches before randomly splitting them into 75% for training and 25% for testing. For optimization, we used 60 ground truth images annotated by an expert. The K-means clustering and U-Net segmentation performance was optimized with the soft Dice loss [16].

The results for both pipelines are reported in Table 3 as mean and standard deviation of Dice (balancing precision and sensitivity) and as mean and standard deviation of the difference (ΔDice) to the Dice value after the warm-up phase. The optimizers perform for each run a warm-up phase and explore the hyperparameter space by evaluating 20 random samples before applying the optimization strategy. The overall improvement by the optimization for each optimizer is

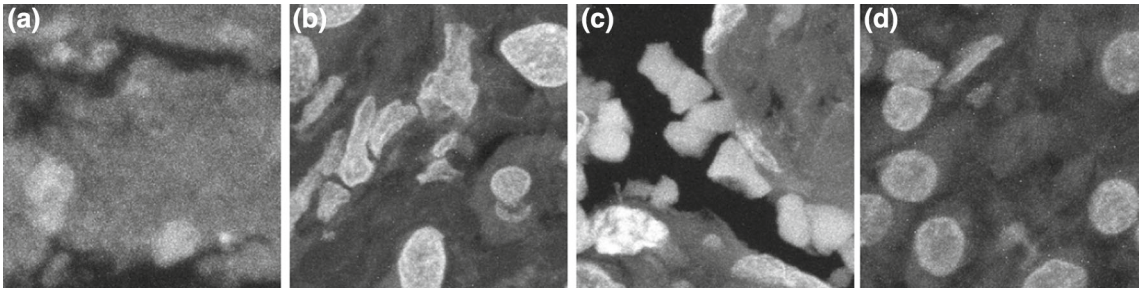

Fig. 2 Examples of prostate tissue images showing various challenges for image analysis. **a** Strong background noise. **b** Strong shape variation. **c** Strong intensity variation. **d** Low contrast

Table 3 Results for the *K*-means clustering and U-Net pipeline with different optimizers

Pipeline	Optimizer	ΔDice (improvement)	Dice
K-means clustering	Random	$0.030 \pm \mathbf{0.028}$	0.606 ± 0.025
	TPE	$0.045 \pm \mathbf{0.028}$	$0.609 \pm \mathbf{0.020}$
	CMA-ES	0.077 ± 0.034	$\mathbf{0.642} \pm 0.021$
	SMAC-RF	$\mathbf{0.094} \pm 0.043$	$\mathbf{0.642} \pm 0.026$
	SMAC-XGBoost	0.064 ± 0.038	0.634 ± 0.021
	Grid Search (coarse)	–	0.614
	Grid Search	–	*0.654*
U-Net	Random	0.019	0.847
	TPE	0.038	0.850
	CMA-ES	0.033	**0.852**
	SMAC-RF	0.017	0.846
	SMAC-XGBoost	**0.039**	0.847
	Grid Search	–	*0.864*

The table shows the improvement ΔDice (mean ± std.) after the warm-up phase and the absolute Dice value (mean ± std.). The best results are highlighted in bold. The global optimum is determined by Grid Search and highlighted in italic

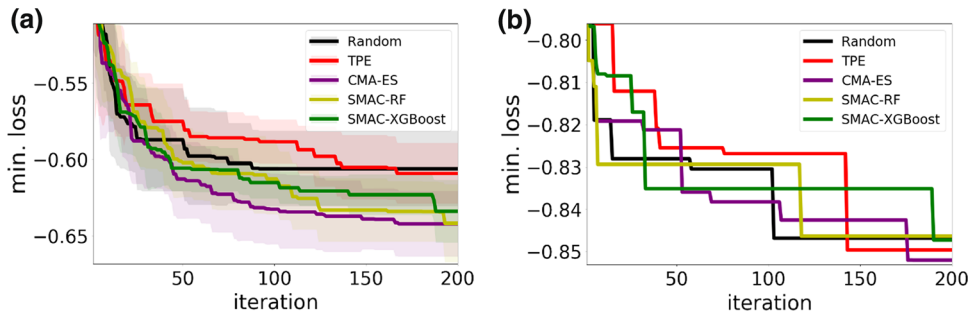

Fig. 3 Convergence of different optimizers as a function of the number of iterations. **a** *K*-means clustering pipeline. **b** U-Net pipeline

reflected by the mean ΔDice. The Dice values as a function of the number of iterations for all optimizers are shown in Fig. 3. For the *K*-means clustering pipeline, CMA-ES and SMAC-RF yield the best segmentation performance with a Dice value of 0.642. However, CMA-ES has a lower standard deviation with a Dice value of 0.021 compared to SMAC-RF. TPE yields the lowest standard deviation regarding Dice among all optimizers. The global optimum of 0.654 was determined by Grid Search. Instead of using (dense) Grid Search with a high number of evaluations (17.388 evaluations), we also used a coarse grid with a similar number of evaluations as for the optimizers (198 evaluations). For the *K*-means clustering pipeline, we obtained a Dice value of 0.614, which is lower compared to SMAC-XGBoost, SMAC-RF, and CMA-ES, but higher than Random and TPE. Regarding ΔDice, SMAC-RF obtains the largest improvement $\Delta\text{Dice} = 0.094$ compared to all other optimizers. For the U-Net pipeline, CMA-ES achieves the largest Dice value of 0.852, SMAC-XGBoost obtains the highest improvement $\Delta\text{Dice} = 0.039$. In Fig. 4, the segmentation results for the best optimizer for each pipeline for Dice and the best optimizer for ΔDice are shown. Comparing the results with the

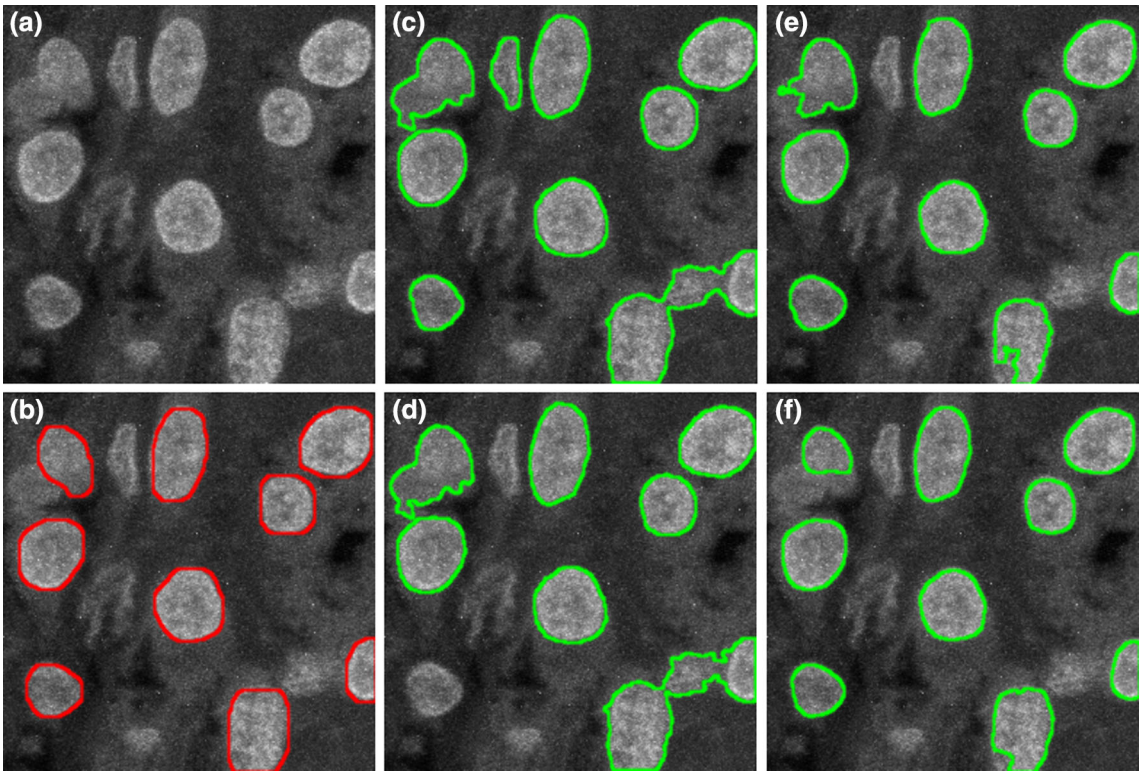

Fig. 4 Comparison of segmentation results for both pipelines with different optimizers. **a** Original image. **b** Original image with ground truth annotations by an expert. **c** *K*-means clustering pipeline with CMA-ES. **d** *K*-means clustering pipeline with SMAC-RF. **e** U-Net pipeline with CMA-ES. **f** U-Net pipeline with SMAC-XGBoost

ground truth, it can be seen that the *K*-means clustering pipeline has problems with splitting cell nuclei for both best optimizers for Dice and ΔDice. Overall, the best segmentation performance determined by Grid Search is achieved by the U-Net pipeline, which yields a significantly higher Dice value of 0.864 compared to the *K*-means clustering pipeline with 0.654. For *K*-means clustering, comparing SMAC-RF and SMAC-XGBoost in Table 3, SMAC-RF yields a better Dice value than SMAC-XGBoost. In contrast, for the U-Net pipeline, SMAC-XGBoost yields a better Dice value than SMAC-RF. Since SMAC-RF and SMAC-XGBoost use the same sampling strategy but different optimization strategies, this demonstrates that the proposed separation of sampling and optimization strategy in HyperHyper yields better solutions.

Experiment 2: detection of HCV proteins

In the second experiment, we evaluated HyperHyper for live cell fluorescence microscopy data displaying fluorescently labeled HCV NS5A as small round particles. Detection of subcellular structures such as proteins is a prerequisite for tracking [3] to obtain quantitative information on cellular processes. The live cell data were acquired by an Ultra-View ERS spinning disk confocal microscope with an image size of 355 × 447 pixels. To detect HCV proteins, we used the spot-enhancing filter (SEF) [22] which consists of applying a Laplacian-of-Gaussian filter (LoG) with standard deviation σ_{LoG}, followed by thresholding the filtered image. The threshold is based on the mean intensity of the filtered image plus a factor c times the standard deviation of the filtered image intensities [8,20]. To detect HCV proteins, the hyperparameters σ_{LoG} and c have to be optimized.

As in experiment 1, we used Random, TPE, CMA-ES, SMAC-RF, and SMAC-XGBoost for hyperparameter optimization. We performed 10 runs per optimizer with 3.500 evaluations distributed on 20 compute nodes (total: 175.000 evaluations). To determine the global optimum, we used dense Grid Search with 35.000 evaluations distributed on 20 compute nodes. The performance of SEF detection was optimized and evaluated using the F1 score (balancing precision and sensitivity) and 128 ground truth annotations. The assignment between the ground truth annotations and SEF detections was determined using the Munkres algorithm [15] and a gating distance of 5 pixels. Similar to experiment 1, we computed the mean and standard deviation of ΔF1 showing the difference to the F1 score after the warm-up phase.

Table 4 Results for the HCV protein detection pipeline with different optimizers

Pipeline	Optimizer	ΔF1 (improvement)	F1
SEF	Random	0.043 ± 0.033	0.871 ± 0.002
	TPE	0.041 ± 0.023	0.867 ± 0.000
	CMA-ES	0.022 ± **0.008**	0.871 ± 0.001
	SMAC-RF	**0.050** ± 0.037	**0.872 ± 0.000**
	SMAC-XGBoost	0.041 ± 0.037	**0.872 ± 0.000**
	Grid Search	–	*0.872*

The table shows the improvement ΔF1 (mean ± std.) after the warm-up phase and the absolute F1 score (mean ± std.). The best results are highlighted in bold. The global optimum is determined by Grid Search and highlighted in italic

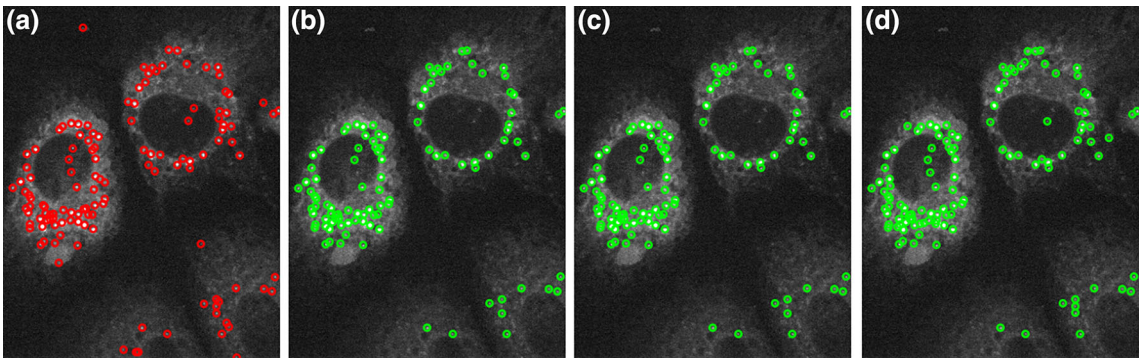

Fig. 5 Detection results for HCV live cell microscopy data with different hyperparameter optimizations. **a** Ground truth annotated by an expert. **b** Experiment 2 using Grid Search. **c** Experiment 2 using SMAC-RF. **d** Experiment 3 using Grid Search

The results for the different optimizers are shown in Table 4. The best performance is obtained by SMAC-RF and SMAC-XGBoost with an F1 score of 0.872, which are the only optimizers reaching the global optimum. The largest improvement ΔF1 is obtained by SMAC-RF with 0.050. In Fig. 5c, the result for SMAC-RF (green circles) is shown together with the ground truth (red circles) in Fig. 5a and the global optimum (Grid Search) in Fig. 5b. The F1 score as a function of the number of iterations for all optimizers is depicted in Fig. 6. To better assess the convergence of the optimizers, only the first 2000 iterations are shown. The fastest convergence is obtained by SMAC-RF and SMAC-XGBoost, whereas TPE is slowest. To obtain more insights into the optimization process and to visualize the dependency between the hyperparameters, we computed the loss surface with Grid Search. The loss surface is shown in Fig. 7a and the global optimum is marked by a blue star. A clear dependence between c and σ_{LoG} is visible by the valley shape of the loss surface. In addition, a visualization of the trail of an optimizer on the loss surface is provided for spatial assessment of the convergence process. In Fig. 7b, the best trails of Random and SMAC-RF are depicted. A trail represents the connection of the best evaluations per optimization step. It can be seen that Random finds the optimum more directly compared to SMAC-RF. However, according to Fig. 6, SMAC-RF has a much faster convergence than Random.

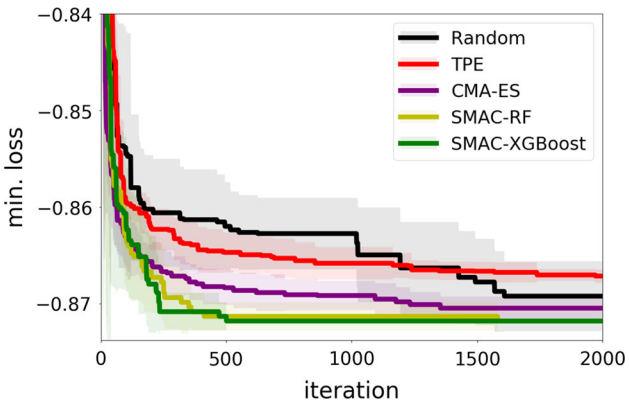

Fig. 6 Convergence of different optimizers as a function of the number of iterations

Experiment 3: image pre-processing for detection of HCV proteins

With this experiment, we show the importance of an infimum projection as visualization of the loss function to gain further insight on the dependency of the hyperparameters. We study an additional image pre-processing step for the pipeline for HCV protein detection from experiment 2. As pre-processing, we smoothed the image with a Gaussian filter with standard deviation σ_{Gauss} and subtracted the filtered

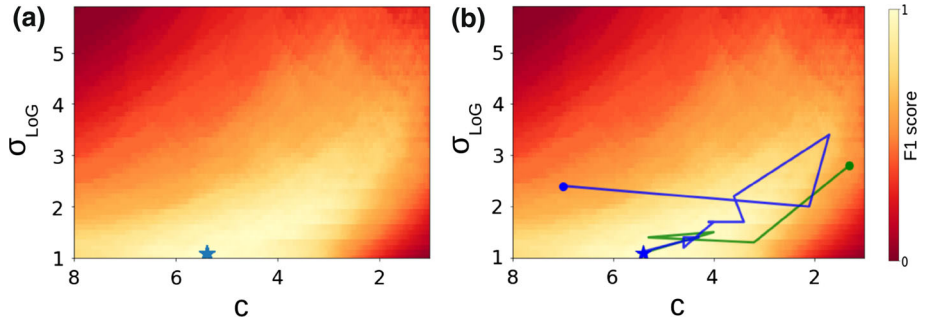

Fig. 7 Loss surface of experiment 2 for 2D hyperparameter space (c and σ_{LoG}). **a** The hyperparameter space was sampled with Grid Search and the global optimum is marked with a blue star. **b** Same as in **a** but with optimization trails of Random (green) and SMAC-RF (blue). For both trails, the dot is the starting point and the star shows the found optimal solution. Both trails represent the best evaluations per optimization step over time

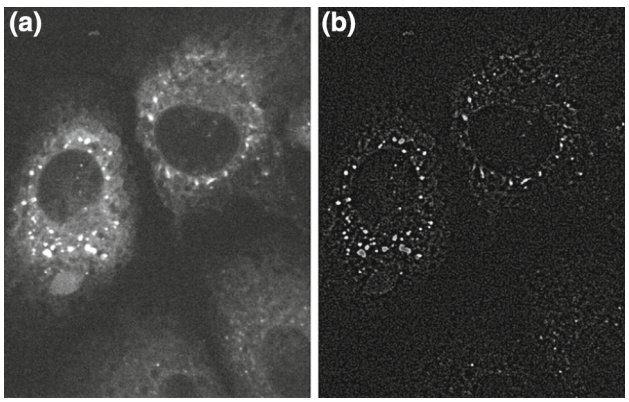

Fig. 8 HCV fluorescence microscopy data. **a** Original image. **b** Pre-processed image with optimal σ_{Gauss} obtained by Grid Search

Table 5 Results for the two HCV protein detection pipelines (experiment 2 and 3) with Grid Search

Pipeline	F1
SEF	0.872
SEF + pre-processing	**0.888**

The table shows the absolute F1 score. The best result is highlighted in bold

image from the original image to enhance the particles and suppress background noise. We now have a 3D hyperparameter space containing σ_{Gauss}, σ_{LoG}, and c. We computed the global optimum with Grid Search (total: 175.000 evaluations).

Figure 8a, b displays the original and filtered live cell data, respectively. Table 5 shows the improvement using the pre-processing step. The F1 score using pre-processing is 1.6% higher compared to the result from experiment 2 without pre-processing. The detection result using pre-processing is displayed in Fig. 5d. All HCV proteins within the upper two cells are detected, whereas without pre-processing as in experiment 2 only a few of them were detected.

To obtain insights into the optimization process and to quantify the dependency of the hyperparameters, we conducted a principal component analysis (PCA) [17] of the loss function. The results are shown in Table 6. The values of the loss function were normalized (zero mean and variance of one), and the eigenvectors with corresponding eigenvalues were computed (principal components, PCs). It can be seen that in order to represent 90% of the variance, the first three PCs need to be taken into account. For the first PC the hyperparameter σ_{Gauss} has a more than ten times smaller influence than c and σ_{LoG}. In addition, the other two PCs have a minor influence on the loss. Therefore, the influence of σ_{Gauss} on the loss is relatively small. However, Table 5 shows that pre-processing by a Gaussian filter improves the detection pipeline performance.

To further investigate the dependencies of the hyperparameters, we propose to generate infimum projection visualizations. The infimum projection of a countable finite n-dimensional loss $\mathrm{L} : Q_1 \times \cdots \times Q_n \to \mathbb{R}$ into a lower dimensional projection P onto the index set $I \subseteq [\#Q]$ of features $Q = \{Q_1, \ldots, Q_n\}$ with m elements can be performed by:

$$\mathrm{P}(I; q_1, \ldots, q_n) = \min_{\substack{q_k \in Q_k \\ k \notin I}} \{\mathrm{L}(q_1, \ldots, q_n)\} \tag{1}$$

In Fig. 9a–c, the infimum projections between the three hyperparameters are shown. Figure 9a can be compared with the loss surface for c and σ_{LoG} in experiment 2 in Fig. 7a where both hyperparameters (c and σ_{LoG}) are plotted. Both loss surfaces show the same structure and therefore a priori knowledge from experiment 2 could be transferred to the optimization problem in experiment 3. In addition, from the loss surfaces in Fig. 9b, c one can see that the former hyperparameters (c and σ_{LoG}) and the additional parameter σ_{Gauss} seem to be independent due to the homogeneous structures of the loss surfaces. Thus, the infimum projection yields addi-

Table 6 Results of PCA for the whole loss surface data

PCA	Variable	PC 1	PC 2	PC 3	PC 4
Eigenvectors	c	0.224	−0.948	0.001×10^{-13}	0.224
	σ_{LoG}	− 0.668	− 0.316	− 0.081	−0.668
	σ_{Gauss}	− 0.054	− 0.026	0.997	− 0.054
	$Loss$	− 0.707	-0.002×10^{-12}	0.003×10^{-14}	0.707
Eigenvalues		1.692	1.000	1.000	0.308
Cumulative variance ratio		42.3 %	67.3 %	92.3 %	100.0 %

The table provides the eigenvectors and eigenvalues of the four principal components (PC) together with the ratio between the cumulative variance and the total variance in [%]

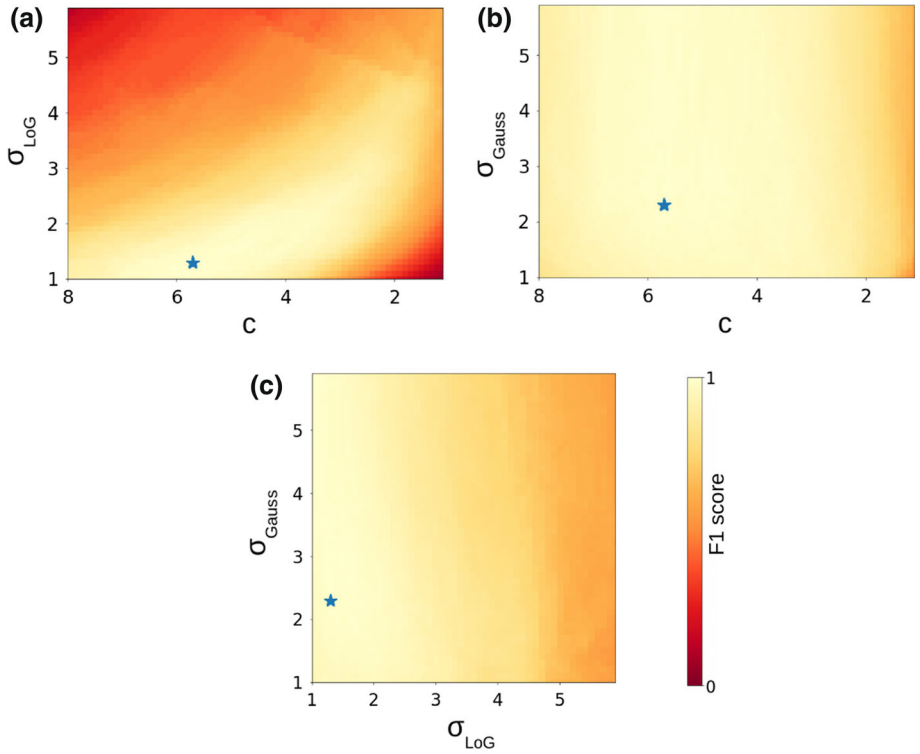

Fig. 9 Infimum projections of the loss surface from experiment 3 for the 3D hyperparameter space (c, σ_{LoG}, and σ_{Gauss}) sampled with Grid Search. The global optimum is marked with a blue star

tional information to the PCA analysis. Figure 9c, d together with Table 5 indicates that the optimization problem can be restructured by optimizing separately σ_{Gauss} and c along with σ_{LoG}. This sequential optimization procedure reduces the 3D hyperparameter optimization to a 1D optimization along with a 2D optimization.

Discussion

The experimental results using the HyperHyper framework with different optimizers for different image analysis tasks show that the separation of sampling and optimization strategy improves the performance. In experiment 1 on the segmentation of cell nuclei we found that CMA-ES and SMAC-RF achieved the best Dice value for K-means clustering, while CMA-ES obtained the best Dice value for the U-Net pipeline. In this application, the task is to segment cell nuclei for telomere quantification on a single-cell basis. False-negative segmentations are not critical. The Dice score was used, which favors high true-positive and low false-negative segmentations. In experiment 2 the task was to detect HCV particles in live cell microscopy images. SMAC-XGBoost, SMAC-RF, and CMA-ES yielded the highest F1 score. The F1 score favors high true-positive and low false-negative detections. In this experiment, false-negative detections are critical, since detection is a prerequisite for single-particle tracking. From the experiment, it turned out that the number of false-negative detections is relatively low.

The results using our hyperparameter optimization framework are relatively close to the maximum possible accuracy, i.e., the global optimum by (dense) Grid Search (Tables 3, 4). The global optimum in all experiments was determined by dense Grid Search with a high number of evaluations, much

higher than the number of samples for the optimizers. We also performed coarse Grid Search with a similar number of evaluations as for the optimizers. For coarse Grid Search, the performance was worse compared to SMAC-XGBoost, SMAC-RF, and CMA-ES. This shows that an appropriate dense grid is necessary to compute the global optimum and to achieve better results than the optimizers.

In order to show the improvement in an optimization strategy, we calculated the improvement ΔDice and ΔF1 with respect to Random Search (warm-up phase with 20 samples). Random Search is a basic method for optimization and thus well suited as baseline for a comparison. To reduce the influence of the randomly selected samples within the warm-up phase, we performed several runs (for K-means clustering in experiment 1 and for SEF in experiment 2) and computed the mean and standard deviation of ΔDice and ΔF1.

The development of efficient optimization strategies is important for image analysis pipelines. For fast implementation of new or extensions of existing optimization and sampling strategies, we used a modular architecture for HyperHyper. Therefore, sampling and optimization strategies for specific use cases can easily be integrated. To find the most appropriate sampling and optimization strategy, we proposed using the infimum projection visualization of the hyperparameter space. By combining spatial and temporal visualizations of the convergence of optimizers, it is possible to obtain insights into the optimization process which helps to select a task-specific sampling and optimization strategy.

Conclusion

We proposed a new hyperparameter optimization framework named HyperHyper, which has several advantages compared to existing optimization frameworks. While existing frameworks include only a limited number of optimization methods, HyperHyper comprises more than 40 different optimizers which was achieved by a modular architecture which separates the sampling and optimization strategy. Using two complex pipelines for segmentation of cell nuclei in tissue microscopy images, we demonstrated the impact of separating sampling and optimization. Furthermore, HyperHyper includes an integrated scheduler and job wrapper to deal with different cluster computing infrastructures and pipelines written in various programming languages. We also demonstrated for a pipeline to detect HCV proteins in live cell microscopy data that visualizations of the loss function help to assess the convergence properties of different optimizers. In addition, we showed that an infimum projection of the loss function can yield insights into the structure of the optimization problem. Furthermore, dependencies between hyperparameters can be revealed which can help to reduce the dimensionality of the hyperparameter space by sequential optimization. This might also help in selecting an optimal sampling and optimization strategy for similar optimization problems.

Acknowledgements This work is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)—Projektnumber 240245660—SFB 1129 (projects P11, Z4) and the BMBF within the projects CancerTelSys (e:Med, #01ZX1602) and de.NBI (HD-HuB, #031A537C).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval All procedures performed in studies involving human participants were in accordance with the ethical standards of the institutional and/or national research committee and with the 1964 Helsinki Declaration and its later amendments or comparable ethical standards.

Informed consent Informed consent was obtained from all individual participants included in the study.

References

1. Afgan E, Baker D, Batut B, van den Beek M, Bouvier D, Čech M, Chilton J, Clements D, Coraor N, Grüning BA, Guerler A, Hillman-Jackson J, Hiltemann S, Jalili V, Rasche H, Soranzo N, Goecks J, Taylor J, Nekrutenko A, Blankenberg D (2018) The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update. Nucleic Acids Res 46(1):537–544
2. Chen T, Guestrin C (2016) Xgboost: a scalable tree boosting system. In: Proceedings of SIGKDD. ACM, pp 785–794
3. Chenouard N, Smal I, De Chaumont F, Maška M, Sbalzarini IF, Gong Y, Cardinale J, Carthel C, Coraluppi S, Winter M, Cohen AR, Godinez WJ, Rohr K, Kalaidzidis Y, Liang L, Duncan J, Shen H, Xu Y, Magnusson KE, Jaldén J, Blau HM, Paul-Gilloteaux P, Roudot P, Kervrann C, Waharte F, Tinevez JY, Shorte SL, Willemse J, Celler K, van Wezel GP, Dan HW, Tsai YS, Ortiz de Solórzano C, Olivo-Marin JC, Meijering E (2014) Objective comparison of particle tracking methods. Nat Methods 11(3):281–290
4. Cleary K, Peters TM (2010) Image-guided interventions: technology review and clinical applications. Annu Rev Biomed Eng 12:119–142
5. Cypko MA, Stoehr M, Kozniewski M, Druzdzel MJ, Dietz A, Berliner L, Lemke HU (2017) Validation workflow for a clinical Bayesian network model in multidisciplinary decision making in head and neck oncology treatment. Int J Comput Assist Radiol Surg 12(11):1959–1970
6. Di Tommaso P, Chatzou M, Floden EW, Barja PP, Palumbo E, Notredame C (2017) Nextflow enables reproducible computational workflows. Nat Biotechnol 35(4):316–319
7. Godinez WJ, Lampe M, Koch P, Eils R, Muller B, Rohr K (2012) Identifying virus-cell fusion in two-channel fluorescence microscopy image sequences based on a layered probabilistic approach. IEEE Trans Med Imaging 31(9):1786–1808
8. Godinez WJ, Rohr K (2015) Tracking multiple particles in fluorescence time-lapse microscopy images via probabilistic data association. IEEE Trans Med Imaging 34(2):415–432
9. Goldberg DE (1989) Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, Boston

10. Golovin D, Solnik B, Moitra S, Kochanski G, Karro J, Sculley D (2017) Google vizier: a service for black-box optimization. In: Proceedings of SIGKDD. ACM, pp 1487–1495
11. Hertel L, Collado J, Sadowski P, Baldi P (2018) Sherpa: hyperparameter optimization for machine learning models. In: Proceedings of NIPS (submitted)
12. Hutter F, Hoos HH, Leyton-Brown K (2011) Sequential model-based optimization for general algorithm configuration. In: Proceedings of LION. Springer, pp 507–523
13. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv:1412.6980
14. Komer B, Bergstra J, Eliasmith C (2014) Hyperopt-sklearn: automatic hyperparameter configuration for scikit-learn. In: Proceedings of ICML workshop on AutoML, pp 2825–2830
15. Kuhn HW (2005) The Hungarian method for the assignment problem. NRL 2:7–21
16. Milletari F, Navab N, Ahmadi SA (2016) V-net: fully convolutional neural networks for volumetric medical image segmentation. In: Proceedings of 3DV. IEEE, pp 565–571
17. Mitchell HB (2012) Data fusion: concepts and ideas. Springer, Berlin
18. Parzen E (1962) On estimation of a probability density function and mode. Ann Math Stat 33(3):1065–1076
19. Rahman SA, Koch P, Weichsel J, Godinez WJ, Schwarz U, Rohr K, Lamb DC, Kräusslich HG, Müller B (2014) Investigating the role of f-actin in human immunodeficiency virus assembly by live-cell microscopy. J Virol 88(14):7904–7914
20. Ritter C, Imle A, Lee JY, Müller B, Fackler OT, Bartenschlager R, Rohr K (2018) Two-filter probabilistic data association for tracking of virus particles in fluorescence microscopy images. In: Proceedings of ISBI. IEEE, pp 957–960
21. Ronneberger O, Fischer P, Brox T (2015) U-Net: convolutional networks for biomedical image segmentation. In: Proceedings of MICCAI. Springer, pp 234–241
22. Sage D, Neumann FR, Hediger F, Gasser SM, Unser M (2005) Automatic tracking of individual fluorescence particles: application to the study of chromosome dynamics. IEEE Trans Image Process 14(9):1372–1383
23. Snoek J, Larochelle H, Adams RP (2012) Practical Bayesian optimization of machine learning algorithms. In: Proceedings of the advances in neural information processing systems, pp 2951–2959
24. Svensson CM, Medyukhina A, Belyaev I, Al-Zaben N, Figge MT (2018) Untangling cell tracks: quantifying cell migration by time lapse image data analysis. Cytom Part A 93(3):357–370
25. Tektonidis M, Rohr K (2017) Diffeomorphic multi-frame non-rigid registration of cell nuclei in 2D and 3D live cell images. IEEE Trans Image Process 26(3):1405–1417
26. Thornton C, Hutter F, Hoos HH, Leyton-Brown K (2013) Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In: Proceedigns of SIGKDD. ACM, pp 847–855
27. Ulman V, Maška M, Magnusson KE, Ronneberger O, Haubold C, Harder N, Matula P, Matula P, Svoboda D, Radojevic M, Smal I, Rohr K, Jaldén J, Blau HM, Dzyubachyk O, Lelieveldt B, Xiao P, Li Y, Cho SY, Dufour AC, Olivo-Marin JC, Reyes-Aldasoro CC, Solis-Lemus JA, Bensch R, Brox T, Stegmaier J, Mikut R, Wolf S, Hamprecht FA, Esteves T, Quelhas P, Demirel Ö, Malmström L, Jug F, Tomancak P, Meijering E, Muñoz-Barrutia A, Kozubek M, Ortiz-de Solorzano C (2017) An objective comparison of cell-tracking algorithms. Nat Methods 14(12):1141–1552
28. Wang Y, Du S, Balakrishnan S, Singh A (2017) Stochastic zeroth-order optimization in high dimensions. arXiv:1710.10551
29. Wollmann T, Bernhard P, Gunkel M, Braun DM, Meiners J, Simon R, Sauter G, Erfle H, Rippe K, Rohr K (2019) Black-box hyperparameter optimization for nuclei segmentation in prostate tissue images. In: Proceedings of Bildverarbeitung für die Medizin. Springer, pp 345–350
30. Wollmann T, Erfle H, Eils R, Rohr K, Gunkel M (2017) Workflows for microscopy image analysis and cellular phenotyping. J Biotechnol 261:70–75