NATIONAL RESEARCH UNIVERSITY HIGHER SCHOOL OF ECONOMICS

as a manuscript

Nikita Moshkov

APPLICATION OF DEEP LEARNING ALGORITHMS TO SINGLE-CELL SEGMENTATION AND PHENOTYPIC PROFILING

PhD Dissertation

for the purpose of obtaining academic degree
Doctor of Philosophy in Computer Science

Academic supervisors:
Attila Kertész-Farkas, Ph.D.
Peter Horvath, Ph.D.
Juan C. Caicedo, Ph.D.

Moscow
2022

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ВЫСШАЯ ШКОЛА ЭКОНОМИКИ

на правах рукописи

Мошков Никита

ПРИМЕНЕНИЕ АЛГОРИТМОВ ГЛУБОКОГО ОБУЧЕНИЯ ДЛЯ СЕГМЕНТИРОВАНИЯ ОДИНОЧНЫХ КЛЕТОК И ФЕНОТИПИЧЕСКОГО ПРОФИЛИРОВАНИЯ

ДИССЕРТАЦИЯ

на соискание учёной степени кандидата компьютерных наук

Научные руководители:
Ph.D. Кертес-Фаркаш Аттила
Ph.D. Хорват Питер
Ph.D. Кайседо Хуан

Москва
2022

APPLICATION OF DEEP LEARNING ALGORITHMS TO SINGLE-CELL SEGMENTATION AND PHENOTYPIC PROFILING

PhD Thesis

Nikita Moshkov

Doctoral School of Interdisciplinary Medicine,
Faculty of Medicine, University of Szeged

Academic supervisors:
Attila Kertész-Farkas, Ph.D.
Peter Horvath, Ph.D.
Juan C. Caicedo, Ph.D.

Szeged
2022

Acknowledgement

First of all, I would like to thank my supervisors: **Prof. Peter Horvath**, **Dr. Attila Kertesz-Farkas** and **Dr. Juan Caicedo** who did a lot of work for my scientific and personal development. I am thankful for giving me the opportunities to do a great research work. I was very lucky to get such mentors and my PhD experience was extremely positive.

I would like to thank people from BIOMAG who helped me since my first days in Hungary, especially Ferenc Kovacs, Krisztian Koos and Reka Kresak. Also, I would like to thank Reka Hollandi, we did a lot of research work together.

"1st part of room 636" folks : Istvan Grexa, Akos Diosdi and Dominik Hirling.

I would like to thank my wife **Valeria Alexandrovich** and my mother **Elena Morozova** for being supportive through all of this process.

I would like to thank eCOST and NEUBIAS (Network of European BioImage Analysts) for providing the travel grants in 2019 and 2020. I am thankful to Dr. Anne Carpenter and Imaging Platform for hosting me in 2019 and to Dr. Juan C. Caicedo for hosting me in 2020 at the Broad Institute.

I would like to acknowledge other grants:

- Kooperatív Doktori Program 2020 (granted by Hungarian Ministry for Innovation and Technology)
- LENDULET-BIOMAG Grant (2018–342)
- EU Regional Development Funds GINOP-2.3.2-15-2016-00006
- EU Regional Development Funds GINOP-2.3.2-15-2016-00026
- EU Regional Development Funds GINOP-2.3.2-15-2016-00037
- EU Regional Development Funds GINOP-2.2.1-15-2017-00072
- Horizon 2020 (ERAPERMED-COMPASS, ERAPERMED-SYMMETRY, DiscovAIR, FAIR-CHARM)
- ELKH-Excellence grant
- HSE University Basic Research Program
- NVIDIA GPU grant

Preface

This thesis was prepared under the agreement on joint research between the University of Szeged, Hungary (SZTE) and the National Research University Higher School of Economics, Russia (NRU HSE). The thesis meets the requirements of both doctoral schools: the Doctoral School of Interdisciplinary Medicine (SZTE) and the Faculty of Computer Science (NRU HSE).

Összefoglalás

Az egysejtes analízis ötlete befolyásolta a sejtmechanizmusok megértését és a gyógyszerhatások értékelését. Ez a genomszekvenálás, a proteomika, a metabolomika és a képalkotás nagy áteresztőképességű technológiáinak fejlődésével vált lehetővé. A képek az egysejtek állapotát leíró adatok egyik meghatározó forrásává, és ezzel a modern mélytanulási módszerek alapjává váltak. Ez a dolgozat az ilyen jellegű elemzések különböző lépéseire és alkalmazásaira összpontosít.

Az egyes sejtek képalapú elemzésének egyik első lépése a sejt- vagy sejtmag-szegmentálás. A dolgozat tartalmazza az egyes sejtek (vagy sejtmagjaik) mélytanuláson alapuló szegmentálására jelenleg használt módszerek áttekintését. A mélytanuláson alapuló szegmentáláshoz jelentős mennyiségű annotált adatra van szükség, ez azonban költséges folyamat, mivel a biológus szakértőktől időt és erőfeszítést igényel. A sejtmagok mélytanulással történő annotálására kifejlesztettünk egy szoftvert, amely segít a sejtek gyorsabb és pontosabb annotálásában. A szegmentálás utófeldolgozási módszerei hasznosak lehetnek, ezért test-time augmentation módszert fejleszttünk sejtmagok szegmentálására majd kiértékeltük a két legnépszerűbb képszegmentálási architektúra esetében.

A szegmentálást gyakran a biológiai fenotípusok azonosítása követi melyet a sejtmorfológia kvantifikálásával végzünk. Ezek változása megkülönböztethet kezelt és kezeletlen sejteket a gyógyszerszűrési kísérletek során. Elemeztük a képosztályozó mély neurális hálózatok és a gyengén felügyelt tanulás reprezentációs képességét a sejtmorfológiára Cell Painting adathalmazokon.

A morfológián kívül a sejtek állapota leírható a génexpresszióval vagy annak a gyógyszernek a kémiai szerkezetével, amellyel a sejtet kezelték. A dolgozat utolsó projektjében három adatforrás: vegyületek kémiai szerkezetének reprezentációi, génexpressziós reprezentációk és morfológiai reprezentációk (CellProfilerrel nyert) prediktív erejét vizsgáltuk a vegyületek aktivitásának előrejelzésére.

Contents

1 Introduction

1.1 The relevance of research

The decisive element in approaching fundamental questions in biology and designing efficient disease treatments is the understanding of cellular molecular processes [1]. The analysis of the single cell has become one of the most important challenges in natural sciences in the 21st century. The game-changing idea [2] is to treat every single cell in tissues as a separate building block with its state and therefore treat tissues as a diverse set of such building blocks, rather than as a homogeneous entity. The means of an extensive investigation of this idea were the new high-throughput technologies for genome sequencing, proteomics, metabolomics and imaging.

Such advancements made it possible to automatically and objectively analyze even on scales as large as millions and billions of cells, thus we have an opportunity to perform high-throughput experiments with single cells (live-cell imaging [3] [4], gene expression profiling [5] and proteomics [6]) and then perform analysis with computational methods, applicable for the obtained type of data and try to make biological sense out of this data.

Different types of data (or data modalities) can allow us to inspect the state of each particular cell from different perspectives. One of the practical tasks, where all the possible information can be useful to make decisions, is drug discovery, especially in personalized medicine. The biggest challenge is to accurately and cost-effectively combine and use the existing expensive treatment modalities.

Here we focus mostly on the imaging data and one of the first steps of the image-based analysis of single cells is *cell or nucleus segmentation* – classification of each pixel as a background or foreground (semantic segmentation), or determining if the pixel belongs to a specific object (instance segmentation), examples are in Figure 1. In recent years this field has been emerging by adopting and creating deep learning algorithms for this task, bringing significant improvements [7].

The segmentation might be followed by the identification of biological phenotypes through the quantification of cell morphology, variation of which might show, for instance, differences between treated and not treated cells in drug screening experiments [8]. The phenotypes can be described by feature-vectors, also called *profiles* and the process of the extraction is called profiling and morphological profiling is also might be referred to as *image-based profiling* [9] [10].

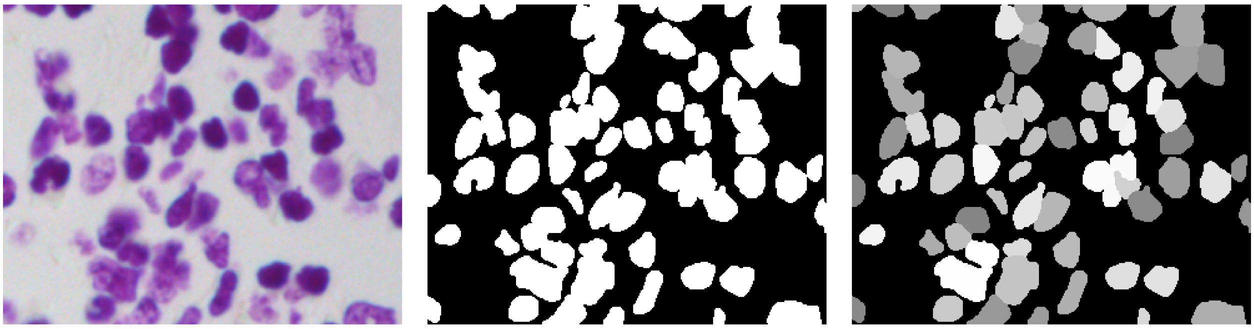


Figure 1: Example of segmentation left-to-right: original image, semantic segmentation, instance segmentation. The source of the image and segmentations: Data Science Bowl 2018 dataset [11].

1.2 Specific aims of the thesis

1.2.1 Review existing methods for cell segmentation

Image pre-processing and nucleus (or cell) segmentation are usually the first steps of the analysis of single-cell images. The accurate segmentation affects the quality of the following downstream analysis, so this step is crucially important.

The author of the thesis contributed to the review paper [7], which puts together the state of the field of nucleus segmentation in 2020-2021. Besides the segmentation methods for 2D and 3D data, it also covers the pre- and post-processing methods, existing datasets and tools for annotation of cellular images.

1.2.2 Deep-learning assisted nuclei annotation

To train a single-cell (nuclei) segmentation based on deep learning, annotated data is needed and the bigger the dataset is, the more robust the model will be. Manual annotation is an expensive process as it requires a significant amount of time and effort from biology experts. To make the annotation process faster and more accurate, a plugin AnnotatorJ [12] for ImageJ/FIJI [13] (the software for bioimage analysis) was developed which combines single-cell identification with deep learning and manual annotation.

1.2.3 Evaluate test-time augmentation approach for nuclei segmentation

Test-time augmentation was an existing approach to improve image classification [14]. In this thesis, test-time augmentation for nuclei segmentation is evaluated. The trained deep learning model for segmentation processes the original input image and several transformed variants of the same image. The obtained segmentation results are then merged. The core idea is that the combination of segmentation results from the original image and its transformed variants will perform better than the segmentation of just the original image or at least will give us hints about uncertain segmentations. The final result is an experimental evaluation of this approach for two popular segmentation deep learning networks.

1.2.4 Image-based morphological profiling with deep learning

The use of deep learning models for image-based profiling (phenotyping of single cells) is investigated. Those deep learning models can be either pre-trained (with ImageNet dataset [15]) or trained (weakly supervised) for the particular single-cell dataset. Using those models, it is possible to extract features (profiles) of the single cells. The obtained features are used in the downstream analysis afterwards (for instance, to predict the mechanisms of action of drugs). We investigate if the features obtained with deep learning networks provide better results in the downstream analysis than classical morphological features [16], particularly for the images obtained with Cell Painting [10] (also see in 2.2).

1.2.5 Assess different sources of features for drug screening

The relative predictive power is compared for three high-throughput sources of features: representations of chemical structures [17] of compounds, gene expression phenotypic profiles obtained with L1000 assay [18] and image-based morphological profiles obtained from Cell Painting [10] images processed with CellProfiler [19] for the task of assay-compound activity prediction.

1.3 Importance of the presented work

The review [7] (Aim 1.2.1) of the most recent 2D and 3D segmentation methods provides insights for practitioners about usage and the most suitable methods for different microscopy modalities. As the end-users of the segmentation pipelines are usually biologists, the guidance for the most effective and easy-to-use framework might be helpful to the community, as accurate segmentation is crucially important for the following downstream tasks.

The usage of deep learning-based algorithms is not possible without accurately annotated image datasets and in the field of nuclei segmentation, such datasets are usually built by experts. We have developed a tool [12] (Aim 1.2.2) to make the creation of annotated nuclei datasets faster, more comfortable and, thus, cheaper.

One of the possible ways to obtain better segmentation is to apply post-processing methods. One of such potential methods is test-time augmentation, which is traditionally used for image classification. The systematic evaluation [20] (Aim 1.2.3) of this method for the task of segmentation of nuclei for the most popular deep learning frameworks and the most popular nuclei dataset so far provides insights into its usefulness.

The main goal of image-based morphological profiling is to get such feature representation that accurately captures the cell state [21]. Deep learning networks for image classification might be able to capture such representations, especially with post-processing steps, such as aggregation. Deep learning image-based morphological profiling combined with a cost-efficient Cell Painting assay [10] can be used in drug discovery and other biologically relevant questions (Aim 1.2.4).

Besides morphology, gene expression profiles and information and representations of chemical structures [17] are useful for extracting useful information in the drug discovery task. The comparison (Aim 1.2.5) of their predictive power can provide insights and demonstrate the usefulness of machine learning models for early-stage drug discovery processes.

1.4 Publications

Papers related to the research topic:

- **Moshkov N.**, Mathe B., Kertesz-Farkas A., Hollandi R., Horvath P. Test-time augmentation for deep learning-based cell segmentation on microscopy images. Scientific Reports. 2020. Vol. 10, 5068. Q1 journal, IF 3.998 (2020). DOI: `https://doi.org/10.1038/s41598-020-61808-3`

- Hollandi R.*, **Moshkov N.***, Paavolainen L., Tasnadi E., Piccinini F., Horvath P. Nucleus segmentation: towards automated solutions. Trends in Cell Biology. 2022. Q1 journal, IF 20.808 (2021). DOI: `https://doi.org/10.1016/j.tcb.2021.12.004`

- Hollandi R., Diosdi A., Hollandi G., **Moshkov N.**, Horvath P. AnnotatorJ: an ImageJ plugin to ease hand-annotation of cellular compartments. Molecular Biology of the Cell. 2020 Vol. 31. № 20. P. 2157-2288. Q1 journal, IF 3.791 (2020). DOI: `https://doi.org/10.1091/mbc.E20-02-0156`

Preprints related to the research project:

- **Nikita Moshkov**, Tim Becker, Kevin Yang, Peter Horvath, Vlado Dancik, Bridget K. Wagner, Paul A. Clemons, Shantanu Singh, Anne E. Carpenter, Juan C. Caicedo. Predicting compound activity from phenotypic profiles and chemical structures bioRxiv 2020.12.15.422887, DOI: `https://doi.org/10.1101/2020.12.15.422887`

- **Nikita Moshkov**, Michael Bornholdt, Santiago Benoit, Matthew Smith, Claire McQuin, Allen Goodman, Rebecca Senft, Yu Han, Mehrtash Babadi, Peter Horvath, Beth A. Cimini, Anne E. Carpenter, Shantanu Singh, Juan C. Caicedo. Learning representations for image-based profiling of perturbations. bioRxiv 2022.08.12.50378, DOI: `https://doi.org/10.1101/2022.08.12.503783`

Conferences, related to the research project:

- HEPTECH AIME19 AI & ML (2019). Test-time augmentation for deep learning-based cell segmentation on microscopy images (poster). Link: `https://indico.wigner.hu/event/1058/contributions/2542/`

Papers unrelated to the research topic published in 2017-2022:

- **Moshkov N.***, Smetanin A.*, Tatarinova T. Local ancestry prediction with PyLAE. PeerJ. 2021. Article 12502. Q2 journal, IF 2.816. DOI: `https://doi.org/10.7717/peerj.12502`

- Piccini F., Balassa T., Carbonaro A., Diosdi A., Toth T., **Moshkov N.**, Tasnadi E. A., Horvath P. Software tools for 3D nuclei segmentation and quantitative analysis in multicellular aggregates. Computational and Structural Biotechnology Journal. 2020. Vol. 18. P. 1287-1300. IF 6.018 (2020), Q1 journal. DOI: https://doi.org/10.1016/j.csbj.2020.05.022

- Grexa I., Diosdi A., Harmati M., Kriston A., **Moshkov N.**, Buzas K., Pietiäinen V., Koos K., Horvath P. SpheroidPicker for automated 3D cell culture manipulation using deep learning. Scientific Reports. 2021. Vol. 11, 14813. Q1 journal, IF 4.379 (2021). DOI: https://doi.org/10.1038/s41598-021-94217-1

- Kornienko I. V., Faleeva T. G., Schurr T. G., Aramova O. Y., Ochir-Goryaeva M. A., Batieva E. F., Vdovchenkov E. V., **N. E. Moshkov**, Kukanova V. V., Ivanov I. N., Sidorenko Y. S., Tatarinova T. V. Y-Chromosome Haplogroup Diversity in Khazar Burials from Southern Russia. Russian Journal of Genetics. 2021. Vol. 57. No. 4. P. 477-488. IF 0.581. DOI: https://doi.org/10.1134/S1022795421040049

Conferences, unrelated to the research project:

- HEPTECH AIME ML&VA on Clouds (2018). Image database generation techniques for DIC brain tissue cell segmentation (poster). Link: https://indico.wigner.hu/event/904/contributions/1874/

2 Background

2.1 Neural networks for segmentation of nuclei and single cells

The history of automated approaches to segment cells and nuclei starts around 60 years ago and those very first approaches were solely based on intensity thresholding [22]. For a very long time, the intensity thresholding (example in Figure 2) was a dominant approach, being the only part of the segmentation pipelines or combined with other classical approaches. Later on, just before the deep learning era, there were other approaches for nuclei segmentation based on classical machine learning [23], active contours [24] [25] and the multilayer gas of circles model [26]. The complexity of biological questions together with the data to be analyzed (developmental biology [27], drug discovery [28], functional genomics [29] and pathology [30]) have started to demand more accurate cell segmentation, and the field has started to seek general solutions to nuclei segmentation task. The adoption of convolutional neural networks and the availability of computational resources to train convolutional deep learning models allowed us to leap toward such solutions.

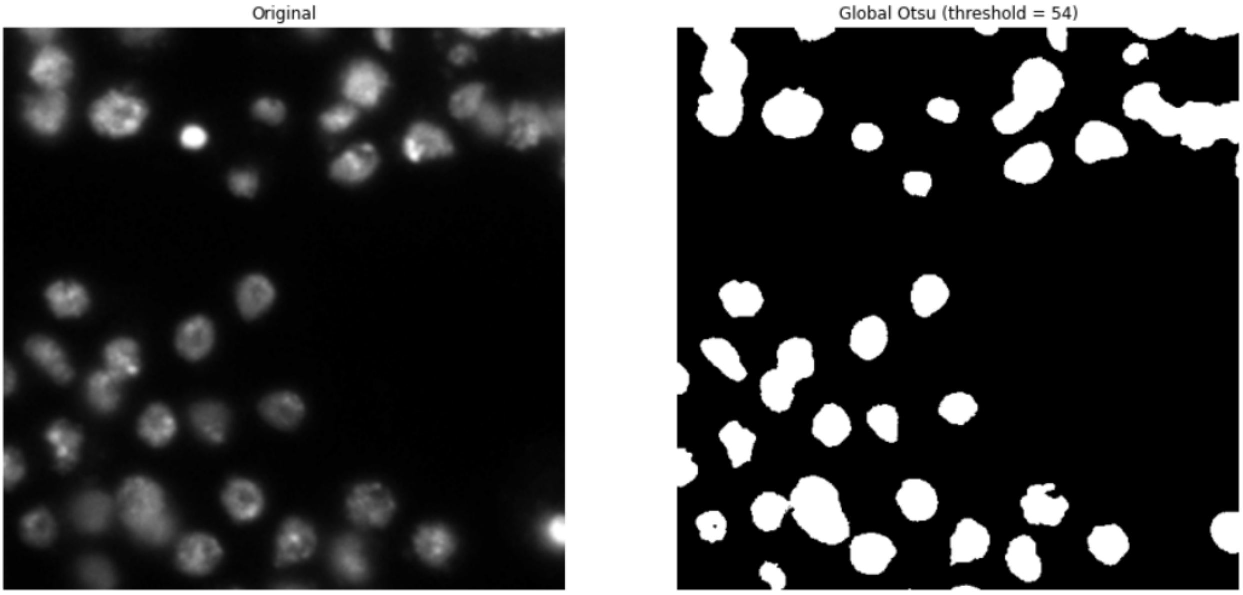


Figure 2: Microscopy image and segmentation mask produced by Otsu thresholding [31] from Scikit-image [32]. The source of the image: Data Science Bowl 2018 dataset [11].

One of the first steps of the image-based analysis of single cells is cell/nuclei detection/segmentation. ***Single-cell segmentation*** (and image segmentation in general) is a vastly developing field: with increased performance of GPUs (graphical processing units) and deep learning neural networks like U-Net [33] (see also 2.1.1), which was the breakthrough for deep learning-based segmentation for biological images (and in the field of deep learning-based segmentation in general). This approach still serves as a baseline for semantic segmentation tasks (i.e. pixel classification) and is used as part of the recent general nucleus/cell segmentation pipelines such as CellPose [34], and StarDist [35] and their derivatives. Besides specialized methods for cell segmentation, methods initially developed for natural image segmentation, like Mask R-CNN [36] (see also 2.1.2) are also applied to single-cell segmentation tasks either by simple fine-tuning or as a part of a complex segmentation pipeline [37].

In addition to deep learning networks themselves, there are common training techniques for regularization, and therefore to train more robust models such as data augmentation for training (modification of original training data by rotating, flipping or adding noise) [38], dropout layers [39], L1 or L2 regularization [40]. Single-cell (nuclei) segmentation task is not an exception to using those techniques.

2.1.1 U-Net

U-Net [33] (Figure 3) is a deep learning-based architecture, developed primarily for biological-image semantic segmentation in 2015 (also was a winner of the ISBI cell tracking challenge). It takes its name from the U-shape encoder-decoder architecture: the input data is firstly compressed by convolutional layers and then expanded back to its original size. U-Net is still widely used as a baseline in nuclei segmentation and there are numerous pipelines based on it for different datasets [7].

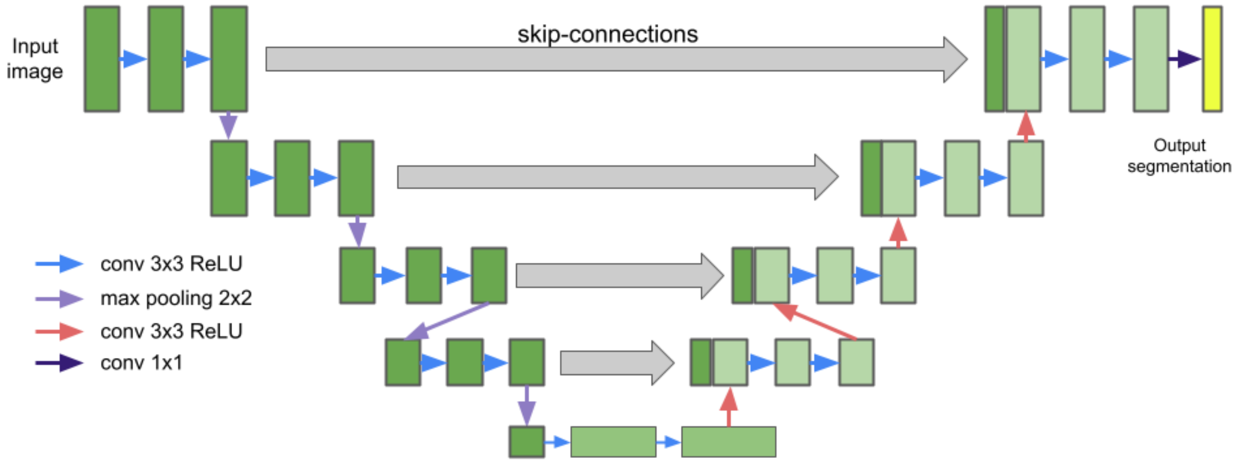


Figure 3: Standard U-Net architecture.

2.1.2 Mask R-CNN

Mask R-CNN [36] (Figure 4) was developed in 2017 for instance segmentation (each pixel in the image is assigned to a separate object) of natural images. Mask R-CNN uses a ResNet [41] architecture as a backbone (usually ResNet50 or ResNet101), which is followed by a region proposal network (RPN). This is stage one of Mask R-CNN, which finishes with a set of proposed regions with objects.

RoIAling (RoI - region of interest) is one of the key enhancements of Mask R-CNN over Faster R-CNN [42], which uses RoI pooling. Both of those operations in principle extract RoIs from the feature maps, RoIAling is more precise. It is followed by the head layers: they predict the class, box offset and an output binary mask for each region of interest (RoI).

Classes are not taken into account for mask generation. RoIAling and the head layers are stage two of Mask R-CNN.

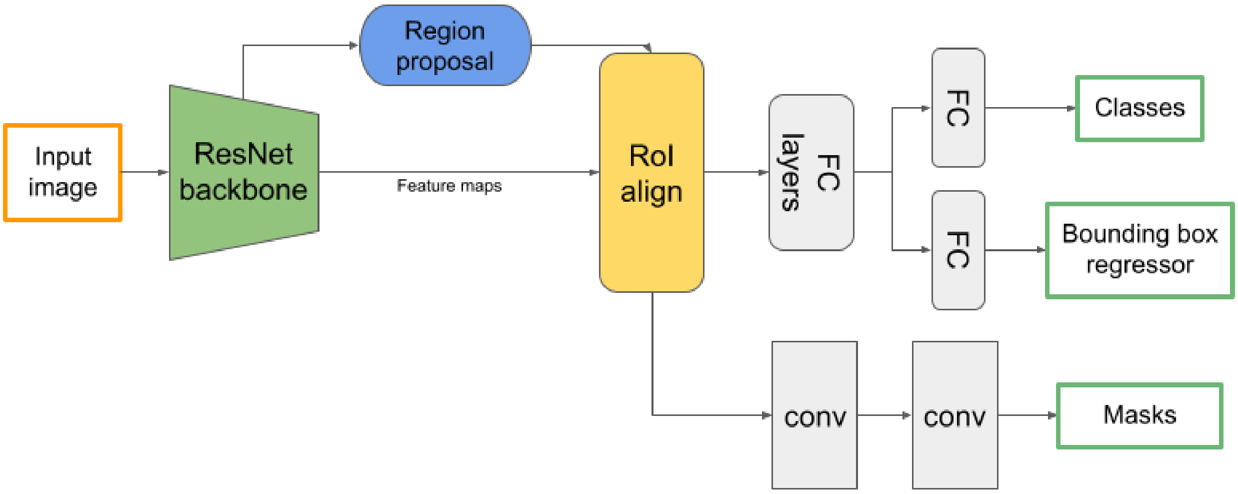


Figure 4: Standard Mask R-CNN architecture.

2.2 Cell Painting and phenotypic profiling

The *target-based approach* used to be dominant in drug discovery, but currently, the *phenotypic approach* to drug discovery takes advantage [43]. Target-based drug discovery focuses on the search for drug targets – gene products, which are the starting point for investigation, and then researchers come up with an idea of how to affect it [44]. The phenotypic approach to drug discovery is empirical: a large number of compounds are tested in target-agnostic assay and the phenotypic variation is monitored [45]. Phenotypic drug discovery expanded the search space of drugs, targets and mechanisms of action making their discovery possible [46].

One way to identify phenotypic variation is through the quantification of cell morphology, which might demonstrate the differences between treated and not treated cells in drug screening experiments [8] [9]. An effective assay for phenotypic-based drug discovery is Cell Painting. This assay was designed to capture as many biologically meaningful morphological features as possible while maintaining the protocol compatible with existing microscopy systems and at the same time keeping it relatively cheap [10]. The output images are five-channel and capture eight cellular compartments (see Figure 5).

Cell morphology might be described by a vector of features - or *profile* (either for individual cells or aggregated for a population of cells), extracted by a multi-stage pipeline [48]. This task can be referred to as ***morphological profiling*** [9] [48] or with broader term ***phenotypic profiling*** [49]. The extracted profiles are processed in downstream analysis of interest. The most popular software to create pipelines to obtain morphological profiles of the cells is CellProfiler [19], the features are hand-crafted, though features obtained with deep learning models are to be researched [50] [51] [52].

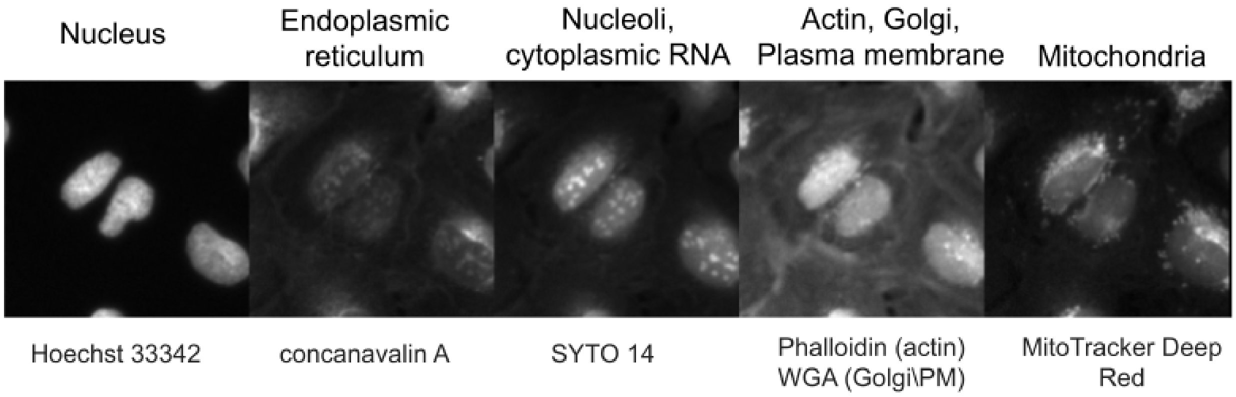


Figure 5: Example of an image obtained with Cell Painting with compartments (labels on top) and stains (labels at bottom). The image is from BBBC022 dataset [47].

CellProfiler [19] is open-source software for the quantitative analysis of cell phenotypes. It is designed for biologist-analysts, so it does not require particular experience in the field of computer science, the biologist-analyst develops only the pipeline with the modules and their settings and best practices pipelines are available for certain types of data (`https://cellprofiler.org/published-pipelines`). The output of the CellProfiler is the feature vector with human-readable features, which could be organized in the groups, such as intensity, texture and shape.

2.3 Computational methods in chemical biology

The field strongly tied to drug discovery is chemical biology, which studies the interaction of small molecules (drugs are usually small molecules) with biological systems (for instance individual cells, tissues and organisms). Like any other field, chemical biology has its own set of computational methods for different tasks [53] [54].

The first problem to solve is to represent chemical molecules conveniently and efficiently way for computational methods. There are different approaches for doing this, the one of the simplest ones is SMILES (Simplified Molecular Input Line Entry System) [55], which is simple, yet very efficient and widely used nowadays. The example is in Figure 6.

Another class of representation of molecules are the fingerprints - binary or numerical vectors of size between 16 to 16384. Fingerprints can be rule-based or obtained with deep learning methods and the efficacy of those representations is not equal [56]. The most commonly used molecular fingerprints are Morgan fingerprints [57], which are binary vectors.

Another term related to the representation of compounds is a scaffold. A scaffold is a core structure of a compound, which consists of all ring structures and links between them and was proposed by Bemis and Murcko [58], example is in Figure 6.

In the last few years, different deep learning-based approaches for computational chemistry have emerged based on convolutional or recurrent neural networks, autoencoders and graph convolutional networks [54].

One of the notable recent methods, based on graph convolutional networks is Chemprop

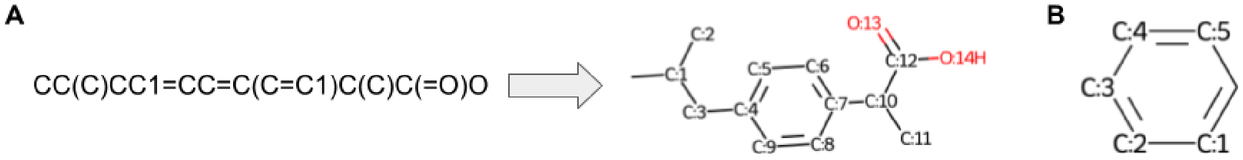


Figure 6: A. SMILES representation of Ibuprofen and its generated graphical representation. B. Bemis-Murcko scaffold of Ibuprofen. Graphical representations and the scaffold were generated with RDKit software (`https://www.rdkit.org/`).

(`http://chemprop.csail.mit.edu/`) [17] [59] [60]. It takes SMILES strings as an input (other feature vectors can be used) and reconstructs molecular graphs, where atoms are nodes and bonds are edges. Then a series of message passing steps are applied to aggregate information from neighboring atoms and bonds, to refine the representation of the molecule.

3 Summary of the research

This section contains a brief description of research projects and the results. Some details are omitted, though can be found in the related publications.

3.1 Nucleus segmentation: towards automated solutions

This section briefly discusses the content of [7].

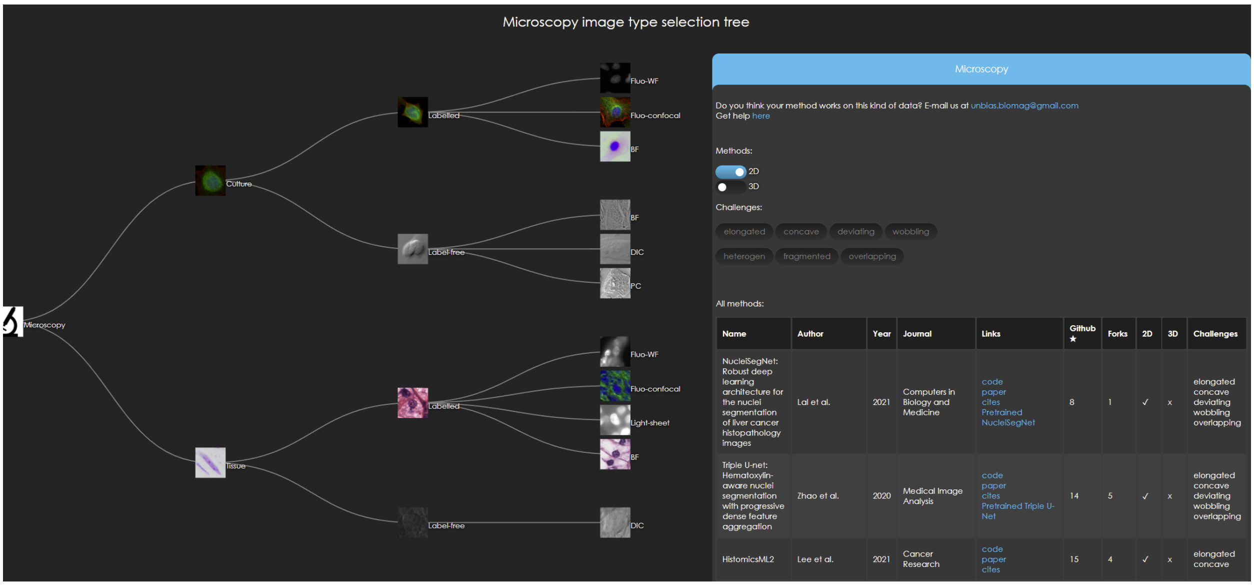
The field of nucleus segmentation was developing over the last few years with the help of deep learning. Practitioners started to use widely deep learning-based segmentation methods, especially after the DSB 2018 challenge [11], which clearly showed the superiority of deep learning-based methods over the classical ones. Besides, the computational resources have become more affordable, and the methods tend to be more user-friendly by providing guides for the tools and sometimes by providing graphical user interfaces. The review is aimed to provide an overview of the methods and datasets related to nuclei segmentation and guide practitioners in the field.

As deep learning methods require the data for training, we start the review, with the description of the openly available annotated nuclei datasets, both in 2D and 3D and for different microscopy modalities. The annotations for those datasets are shared as background-foreground (BG-FG) masks or as object masks (when each object is outlined separately). The first observation is that not so many annotated datasets are shared particularly for 3D data. The possible reason is that the laboratories started to massively switch to 3D not long ago, besides the usage of 3D over 2D is not always a necessity. An example of the 3D dataset, which might be used as a benchmark (and in fact is already used) is A549-Dataset [61]. Another observation - very few imaging modalities are well represented even in the case of 2D datasets. Most of the datasets have only fluorescent, brightfield or hematoxylin and eosin stained (H&E) images. The notable exception is the LIVECell [62] large-scale label-free dataset.

The review part about datasets is then followed by the part about annotation tools. Most of those annotation tools were released recently. We observed the presence of open-source and free tools for annotation of both 2D and 3D data.

The last part of the review is about segmentation methods and tools. The reviewed segmentation methods were classified using meaningful criteria for practitioners. First, the methods were classified by the dimensionality of the input image (2D, 3D or both 2D and 3D). Next, for each method, the availability of the code was checked. Another important criterion is the availability of extended versions of tutorials, as the users of the segmentation methods for biological tasks do not necessarily have computer science expertise and need a clear step-by-step guide to use those methods. The last important criterion is if the tool runs or can be run in the cloud, which has become a very common scenario for running computationally demanding tasks.

Another contribution of this review is the assistant tool for nuclei segmentation method selection (called *unbiased*) which is available online at GitHub Pages `https://biomag-lab.github.io/microscopy-tree/`. It is supposed to help in choosing potentially useful methods based on microscopy modality, the dimensionality of images and potential challenges in the data of interest.



Microscopy

Do you think your method works on this kind of data? E-mail us at unbias.biomag@gmail.com
Get help here

Methods:

2D
3D

Challenges:

elongated concave deviating wobbling
heterogen fragmented overlapping

All methods:

Name	Author	Year	Journal	Links	Github ★	Forks	2D	3D	Challenges
NucleiSegNet: Robust deep learning architecture for the nuclei segmentation of liver cancer histopathology images	Lal et al.	2021	Computers in Biology and Medicine	code paper cites Pretrained NucleiSegNet	8	1	✓	x	elongated concave deviating wobbling overlapping
Triple U-net: Hematoxylin-aware nuclei segmentation with progressive dense feature aggregation	Zhao et al.	2020	Medical Image Analysis	code paper cites Pretrained Triple U-Net	14	5	✓	x	elongated concave deviating wobbling overlapping
HistomicsML2	Lee et al.	2021	Cancer Research	code paper cites	15	4	✓	x	elongated concave

Figure 7: The interface of the assistant tool for the selection of segmentation methods. On the left, there is a tree of microscopy modalities. In the top-right, there are controls for filtering for choosing 2D/3D methods and specific methods for segmentation challenges. In the bottom-right, there is a list of segmentation methods.

The main result of the review turned out to be the raising of concerns and questions about the current state of the field. The first concern is related to the lack of diversity of existing datasets in terms of microscopy modalities. Turns out most of those openly published annotated datasets are either for H&E images or fluorescent images. Other microscopy modalities (e.g, DIC (differential interference contrast), light-sheet or phase contrast) are poorly represented in publicly available datasets. Besides, the size of the published datasets also matters, most of the datasets do not contain many objects and images.

Another point is a call for a solution to the common challenges in nuclei segmentation, such as touching, overlapping and irregularly shaped nuclei [35] [63] [64] [65]. Current deep learning methods can partially address those challenges, but more progress is desired. Both novel model architectures and high-scale training datasets might positively impact in this regard.

The real problem, which is on the surface, but rarely discussed, is the lack of a unified approach for the evaluation of nuclei segmentation methods. After inspecting all the methods eventually presented in the review, it has become clear that the evaluation methods and the datasets don't overlap. Even though there are datasets that are supposed to be the standards, different subsets of the test sets are getting used in different articles. The problem could be solved by discussions inside the community and enforcing the standards. Two candidate platforms to host such standardized tests could be Kaggle and BIAFLOWS [66].

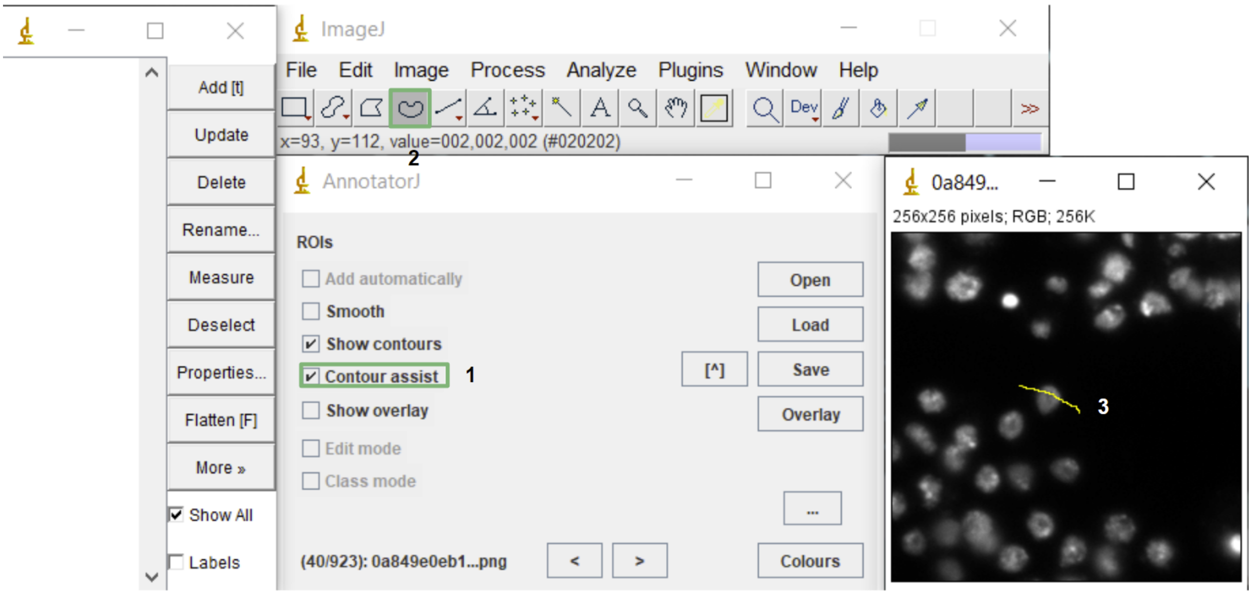
The last conclusion of the paper is that the field could try to move towards the general models which can segment nuclei from images of diverse modalities. Some models are already capable of doing this, though with a limited amount of modalities, for instance, the models obtained during the DSB 2018 challenge [11] [67].

3.2 AnnotatorJ: an ImageJ plugin to ease hand annotation of cellular compartments

This section briefly discusses the content of [12].

To train a single-cell (nuclei) segmentation based on deep learning, annotated data is needed. To train more robust models, bigger datasets are desired, but manual annotation is an expensive process as it requires a significant amount of time and effort from biology experts. To make the annotation process faster and more accurate, a plugin AnnotatorJ [12] for ImageJ/FIJI [13] (the software for bioimage analysis) was developed which combines single-cell identification with deep learning and manual annotation.

The main feature of AnnotatorJ is a contour assistant. Contour assistant uses the pre-trained U-Net model to predict the area covered by the object of interest. After that, the user can refine the contours of the object if needed.



x=93, y=112, value=002,002,002 (#020202)

ROIs

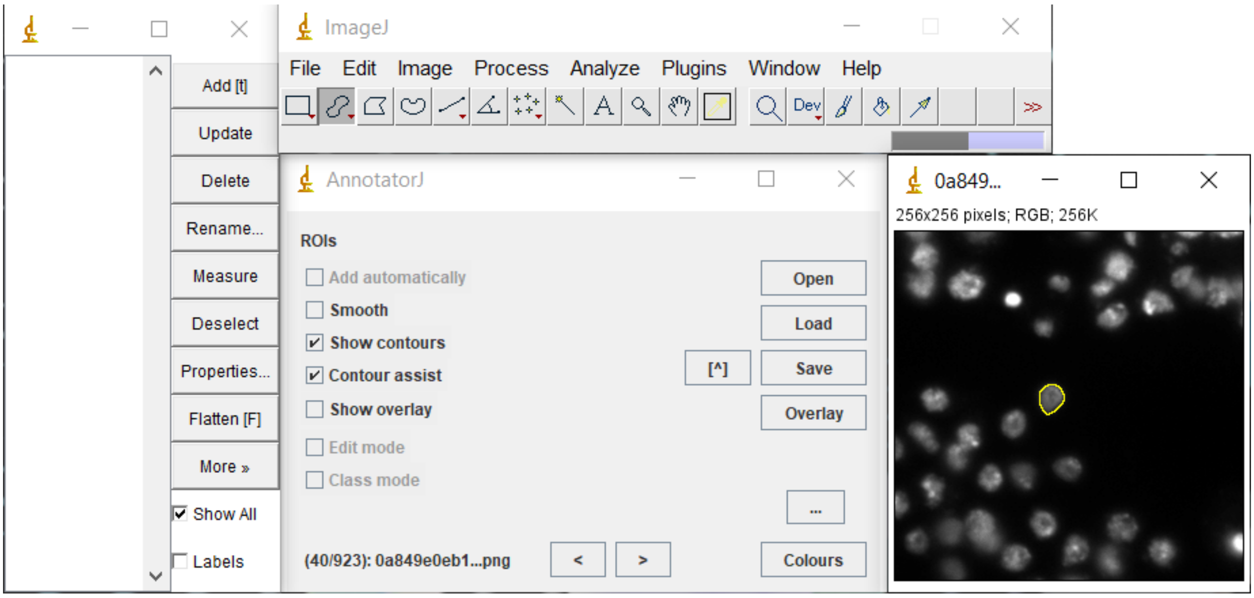
☐ Add automatically
☐ Smooth
☑ Show contours
☑ Contour assist
☐ Show overlay
☐ Edit mode
☐ Class mode

(40/923): 0a849e0eb1...png

0a849...
256x256 pixels; RGB; 256K

☑ Show All
☐ Labels

Figure 8: First step of annotation with contour assist: initialize contour by drawing a line on the object. The numbers and green boxes show the steps to perform in the interface. The source of the microscopy image: Data Science Bowl 2018 dataset [11].



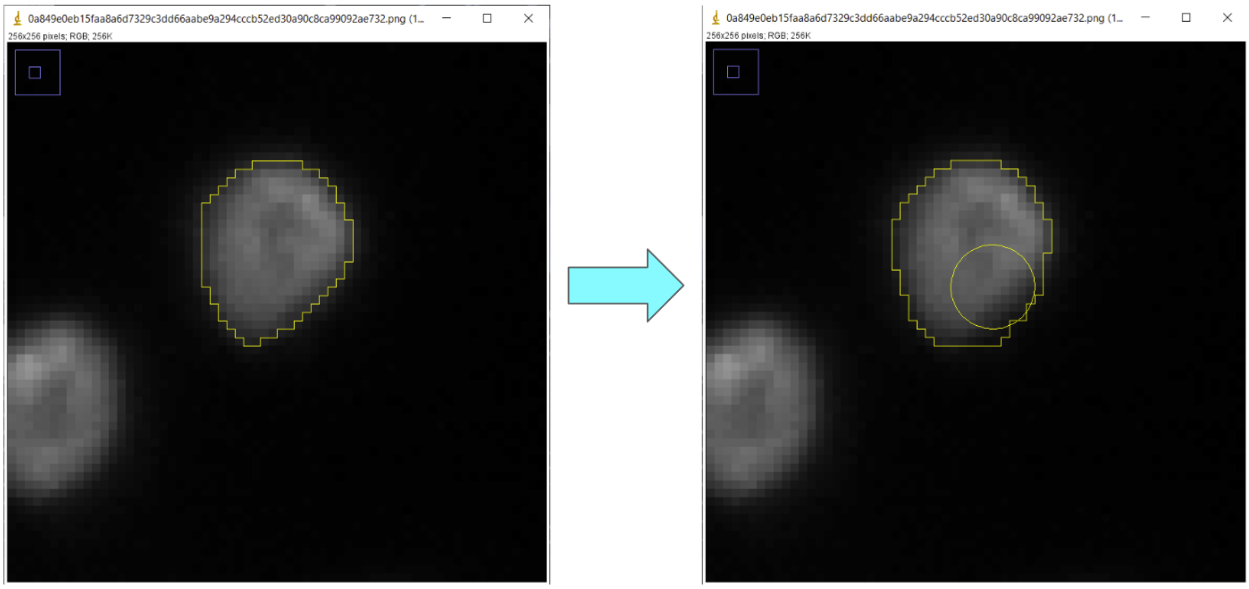
☑ Show All
☐ Labels

ROIs
☐ Add automatically
☐ Smooth
☑ Show contours
☑ Contour assist
☐ Show overlay
☐ Edit mode
☐ Class mode

(40/923): 0a849e0eb1...png

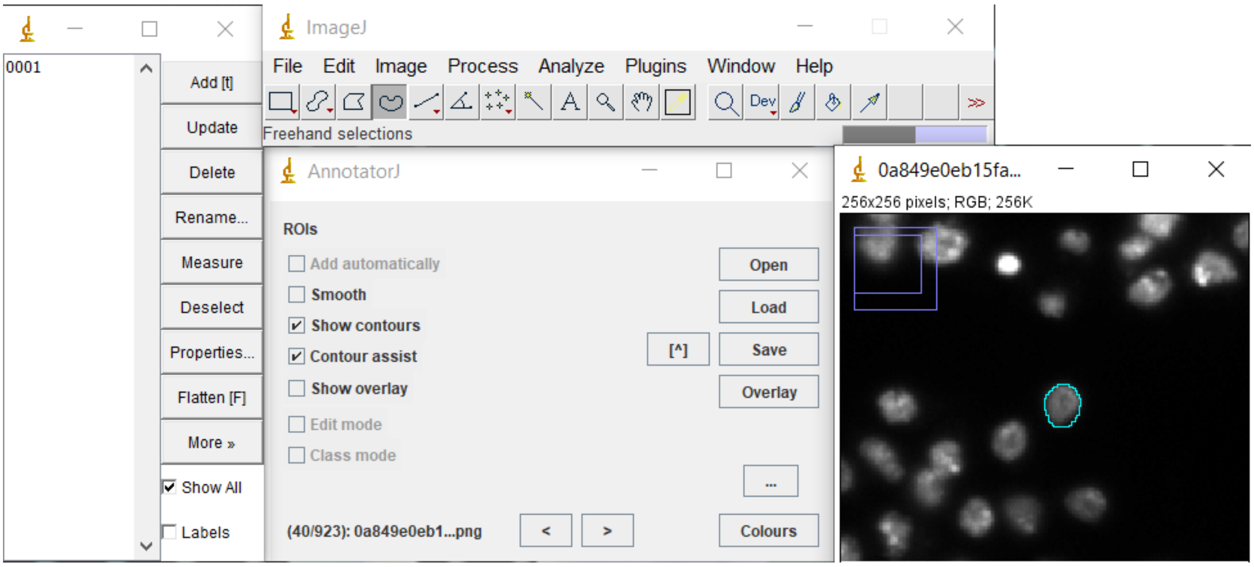
256x256 pixels; RGB; 256K

Figure 9: Initialized contour by pre-trained deep learning segmentation model (in the right). The source of the microscopy image: Data Science Bowl 2018 dataset [11].



0a849e0eb15faa8a6d7329c3dd66aabe9a294cccb52ed30a90c8ca99092ae732.png (1...
256x256 pixels; RGB; 256K
0a849e0eb15faa8a6d7329c3dd66aabe9a294cccb52ed30a90c8ca99092ae732.png (1...
256x256 pixels; RGB; 256K

Figure 10: Refining the contour of the object. The source of the microscopy image: Data Science Bowl 2018 dataset [11].



☑ Show All
□ Labels

□ Add automatically
□ Smooth
☑ Show contours
☑ Contour assist
□ Show overlay
□ Edit mode
□ Class mode

(40/923): 0a849e0eb1...png

Figure 11: Refined object is added as a region of interest after refining the borders and pressing ‘Q‘ key. The source of the microscopy image: Data Science Bowl 2018 dataset [11].

To make trained models compatible with ImageJ/Fiji, which is developed in Java, we used the library DL4J and ND4J (`http://deeplearning4j.org/`). AnnotatorJ is openly available at `https://github.com/spreka/annotatorj`.

3.3 Test-time augmentation for deep learning-based cell segmentation on microscopy images

This section briefly discusses the content of [20].

Deep learning-based nuclei segmentation heavily relies on manually annotated data, which in most cases is annotated by domain experts. To increase the amount of training data and train more robust models, data augmentation [38] (see 2.1) has become a common technique in deep learning. Data augmentation is frequently used in the case of diverse or limited datasets, which is often the case in the field of nuclei and cell segmentation.

While the usual data augmentation approach is performed during the training time, the idea of another approach, test-time augmentation (TTA) (Figure 12) is to perform predictions on the original and the augmented versions of the data samples and then merge the predictions. This technique existed for some time and was successfully used in image-analysis tasks [68] [69] [70]. The experiments with test-time augmentation were conducted in the setting of the nucleus segmentation task.

3.3.1 Test-time augmentation

The pipeline of test-time augmentation includes four steps:

1. Augmentation of the original image.

2. Inference of original and augmented versions of the image.

3. Dis-augmentation: if the original image was flipped or rotated, the transformation should be reverted to the original orientation to allow further correct merging of the predictions.

4. Final merging: this step is different for Mask R-CNN and U-Net and discussed further.

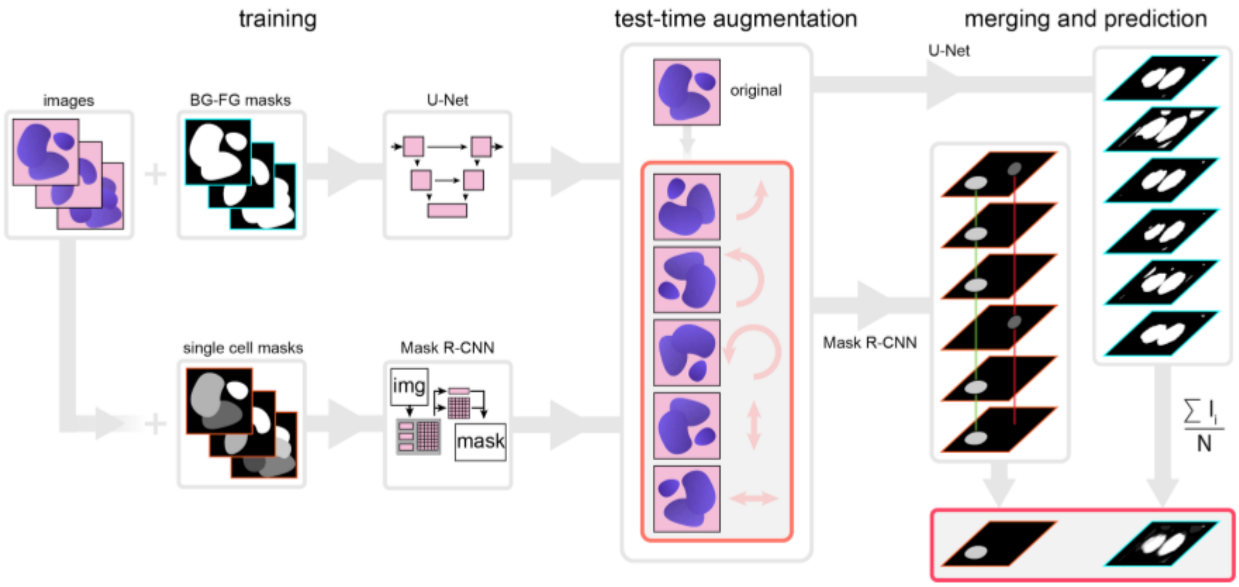


Figure 12: Proposed test-time augmentation techniques. Input: Run inference on several augmented instances of the same test images with trained models. To merge predictions, pixel-wise majority voting was used for U-Net and object matching with majority voting was used for Mask R-CNN. The source of the figure [20].

For U-Net predictions step (4) is straightforward, just sum and average all the dis-augmented probability maps. The resulting probability map is then converted to a binary mask by thresholding (0.5) which is further used for evaluation of the segmentation (Figure 12, right).

Mask R-CNN, as an instance segmentation framework, requires more post-processing. Here, each object is processed separately: for each detected object the majority voting is done. Before majority voting the object alignment should be done: the objects from the predictions of original and augmented versions of the input image are checked if those can be considered the same object. In this setup, two objects (each from different versions of the input image) are considered to be the same object if the intersection over union (IoU, also known as Jaccard Index, (Eq. 1)) between them is at least 0.5. If the same detected object is present in the majority of the predictions, then it will be included in the final prediction mask. The mask of the included object is corrected by majority voting on the pixel level.

$$IoU(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

3.3.2 Materials and methods

For the experiments the popular neural networks for segmentation were chosen: U-Net [33] (for semantic segmentation) and Mask R-CNN[34] (for instance segmentation) and the data for experiments mostly comes from Data Science Bowl 2018 dataset [11] with additional sources [71] [72] [73] [74] [75] [76] [77]. The original images were cropped to the size of 512 × 512 pixels. Images with a resolution lower than 512 × 512 were resized. This primary dataset was split into two datasets: one with fluorescent images (further referred to as Fluorescent or Fluo) and tissue images (further referred to as Tissue). For both of those datasets the following train-test splits were done:

- 95% images in the train set and 5% in the test set (referred to as Fluo_5 or Tissue_5)
- 85% images in the train set and 15% in the test set - repeated 6 times in cross validation setting (cross-validation split 1 is referred to as Fluo_15 or Tissue_15)
- 70% images in the train set and 30% in the test set (referred to as Fluo_30 or Tissue_30)

Separate models were trained for each holdout set. For training, the augmentation was used using horizontal and vertical flip, 90°, 180° and 270° rotations. Augmentations were done before the training (not on-the-fly), which means that the training set size was equal for each split was $6 * number\ of\ unique\ images\ in\ the\ training\ set.$

In the experiments with the U-Net (the architecture was described in 2.2.1) the widely-used implementation [78] based on Tensorflow [79] and Keras was used. The models were trained for 200 epochs with a constant learning rate of 3×10^{-4}. The initial parameters were initialized randomly. A binary cross-entropy loss function with ADAM [80] optimizer were used. Batch size was set to 1 due to GPU memory limitations. Additionally, trainings with and without the use of augmentations in training time were run with U-Net. For the experiments with Mask R-CNN, Matterport's codebase was used [81], also based on Tensorflow and Keras. Evaluation scripts were used from [37].

Mask R-CNN models were trained for 3 epochs for different layer groups in the following order:

- Initialize with COCO weights (`https://github.com/matterport/Mask_RCNN/releases/download/v1.0/mask_rcnn_coco.h5`)
- Epoch 1: all network layers were trained at a learning rate of 10^{-3}.
- Epoch 2: training of ResNet stage 5 and head layers at a learning rate of 5×10^{-4}.
- Epoch 3: Train only head layers at a learning rate of 10^{-4}.

The loss function was binary cross-entropy with ADAM [80] optimizer, batch size 1. This training strategy replicates the one from [37]. Mask R-CNN models were trained only with the use of augmentations in training.

mAP_{DSB} for an image is calculated as follows: calculate the average precision over all test images at IoU threshold t (IoU is calculated between predicted and ground-truth objects) and average over all IoU thresholds T (2). In this equation, $TP(t)$, $FP(t)$ and $FN(t)$ stand for a number of true positive, false positive and false negative objects, respectively:

$$mAP_{DSB} = \frac{1}{|T|}\sum_{t\in T}\frac{TP(t)}{TP(t)+FP(t)+FN(t)},\\ T = \{0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95\} \tag{2}$$

U-Net predictions were evaluated using the intersection over union metric (Jaccard Index) (Eq. 1). TTA's performance is evaluated by calculating the difference between the prediction scores obtained after applying TTA (*merged*) and after regular prediction (*original*). Next, TTA's performance was evaluated by calculating the difference:

$$delta = merged - original \tag{3}$$

3.3.3 Results

Test-time augmentation improved the performance for all the train-test splits on average, if used together with Mask R-CNN models. The mean gain in the mAP_{DSB} metric is between 0.01 and 0.02. While in most of the test images mAP_{DSB} improved, there are a few images with degraded performance (Figure 13).

Test-time augmentation used together with U-Net models also provided improvement in the IoU metric. We can observe that for most model checkpoints in every training scenario, except at the beginning of the training, when the model is underfitting (Figure 14).

In some test examples, test-time augmentation could change the prediction quality by a large margin (see examples in Figure 15).

Test-time augmentation combined with the method [37] (the best performing method for the DSB 2018 test set according to the Kaggle scoreboard at the time of publishing of the paper [20]) further increases the performance by 0.011 in mAP_{DSB}.

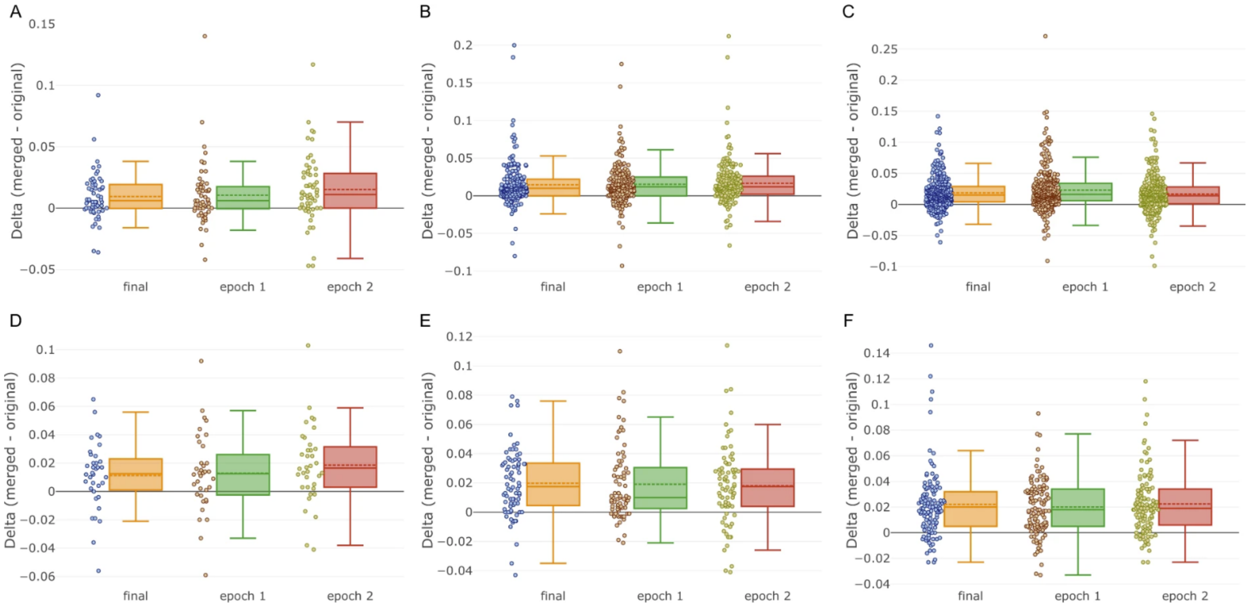


Figure 13: Test-time augmentation impact on segmentation performance (*delta* of mAP). Each point represents an image. Bars: training epochs. A dashed line in bars: mean, a solid line in bars: median. Sets: A. Fluorescent_5. B. Fluorescent_15 (cross-validation 1) C. Fluorescent_30. D. Tissue_5. E. Tissue_15 (cross-validation 1) F. Tissue_30. The source of the figure [20].

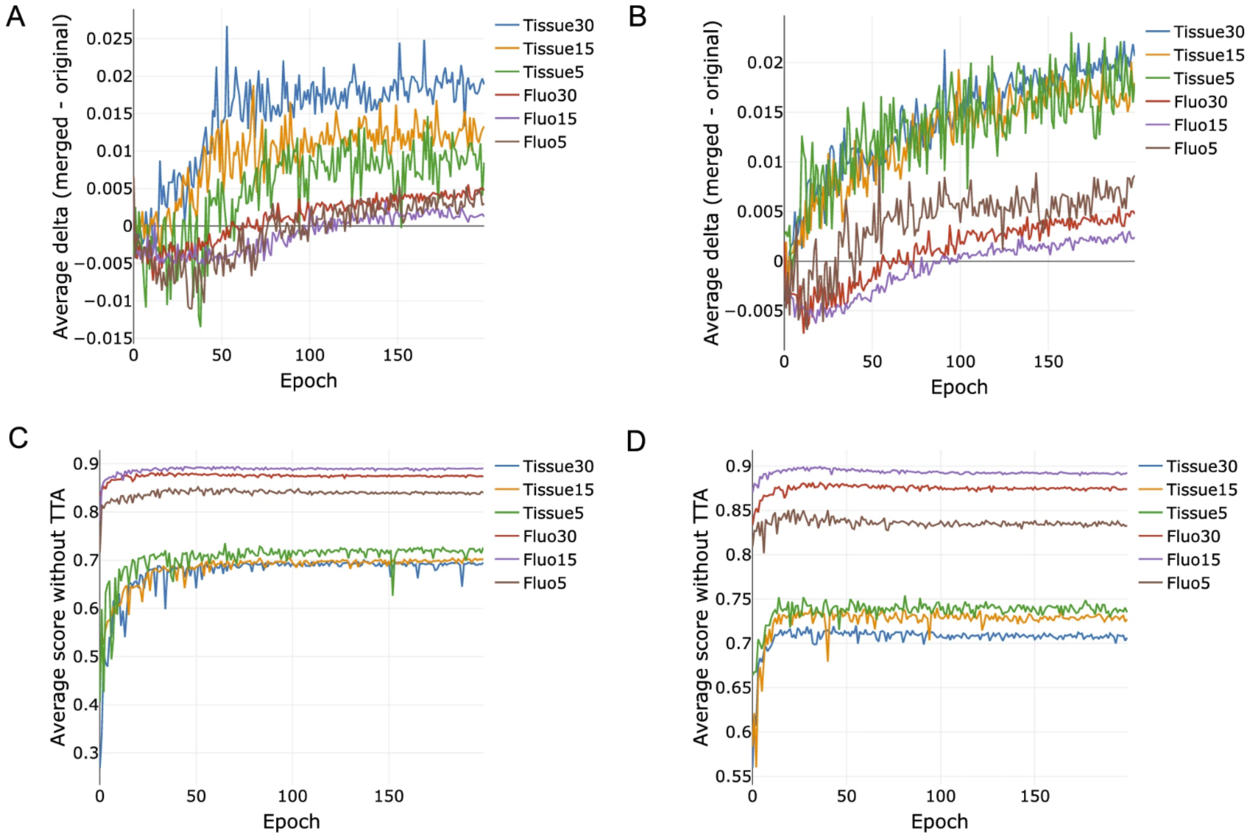


Figure 14: Mean Jaccard index in the test sets and impact of TTA for U-Net. A. Mean *delta* of Jaccard index in models trained without augmentations. B. Mean *delta* of Jaccard index in models trained with augmentations. C. Mean Jaccard index in test sets in models trained without augmentations. D. Mean Jaccard index in test sets in models trained with augmentations. The source of the figure [20].

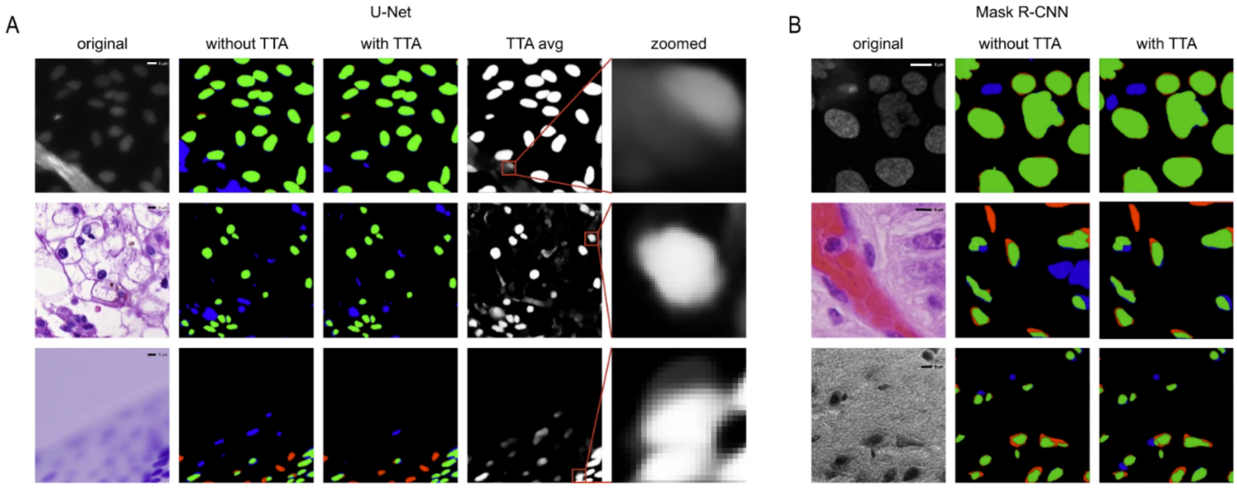


Figure 15: Comparison of predictions with and without TTA on example images. A. U-Net. First column: original image, the second: predictions without TTA, the third: predictions with TTA. Colors: false negative predictions (red), true positive (green), and false positives (blue). The fourth column – averaged TTA predictions before thresholding and the fifth: zoomed insets from the previous column. Rows are example images. B. Mask R-CNN. Columns are as first three in A, rows are example images. The source of the figure [20].

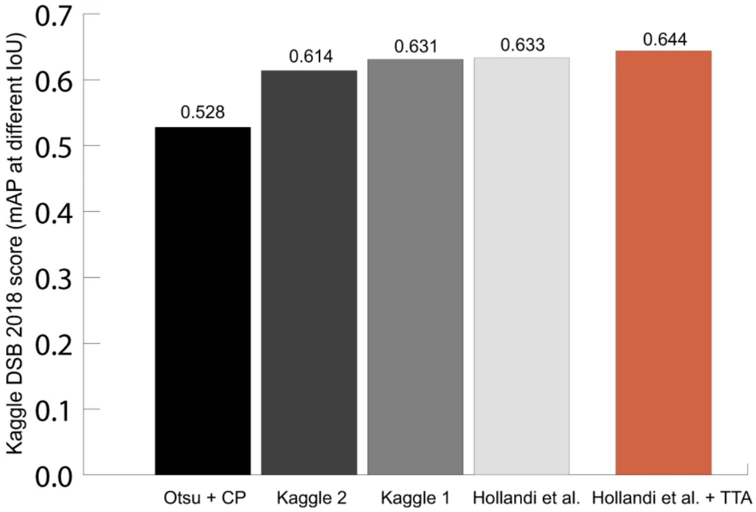


Figure 16: DSB 2018 Stage 2 test scores for different methods, compared to [37] + TTA. The source of the figure [20].

3.4 Learning representations for image-based profiling of perturbations[1]

This section briefly discusses the content of [82].

Phenotypic drug discovery is based on observations of drug effects on treated subjects, in our particular case, we consider single-cells. This problem not only requires significant

[1] The article is online as a pre-print and is being submitted to a journal

wet lab efforts but also computational approaches to process the output data. One of the first attempts to measure treatment effects using features extracted from fluorescent imaging data was [9]. Later on, CellProfiler [19], the standard approach to extract representations of single-cells was released. It produces features which are human-readable and their usefulness was proven in different downstream tasks [16].

Now, the question is, what if we can extract even more biologically relevant representations of cells from images using deep learning? With inspiration from representation learning and popular deep learning architectures for image classification, researchers have started to seek a methodology that could allow them to extract such biologically relevant representations.

One of the first attempts of using transfer learning (usage of pre-trained image classification networks with ImageNet dataset [15]) for morphological profiling was performed in [51] on full images, meaning that the full image was resized to the input size of the network and was run in inference mode.

Training models directly on images of single-cells have been explored in proof-of-concept experiments [50]. It was based on weakly-supervised learning (WSL), which does not require manually annotated data to learn feature representations. Instead, it uses treatment labels as a proxy for the phenotypes of interest. These treatment labels are weak because there is no certainty that all the treatments have a phenotype sufficiently different from the untreated cells (negative controls) or resulting phenotypes are not similar for different treatments.

Here, a systematic evaluation of three large-scale Cell Painting public datasets is conducted. Those datasets contain thousands of perturbations, hundreds of plates, and millions of single cells. The tested representations are extracted by pre-trained models and models trained in a weakly-supervised setting and compared against classical features. To run training and feature extraction experiments, the publicly available tool *DeepProfiler* was developed.

The current best practices found for making deep learning methods improve the quality of downstream analysis, which are reported below. For interpretation of the obtained results with trained models and reasoning about challenges, a causal modeling framework is used [83] [84].

3.4.1 Cell Painting datasets

In this study, five datasets were used in total:

- BBBC037 (also known as TA-ORF) dataset [85], published in 2017 to test morphological profiling using overexpression in human cells as a general approach to annotate gene and allele function.
- BBBC022 dataset [47], published in 2013, screened 1600 bio-active compounds.
- BBBC036 dataset (also known as CDRP-Bioactivies) [86], published in 2017, screened 2000 bio-active compounds.

- BBBC043 dataset (also known as LUAD) [87], testing lung adenocarcinoma variants (375 in total).
- LINCS dataset [88] screened 1300 compounds.

Images in all datasets above were taken with 20X magnification and five-channel (all captured with Cell Painting assay). The first three datasets in the list above are used as benchmarks, the latter two are only used to construct a combined Cell Painting dataset (discussed in section 3.4.5).

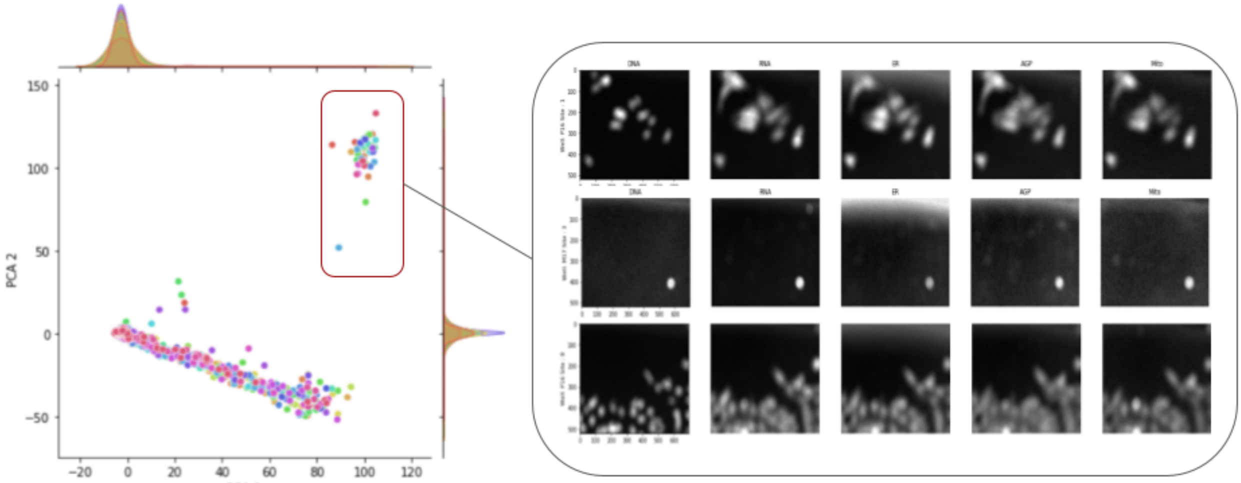


Figure 17: Example of quality controls for BBBC022 dataset. On the left: PCA plot for the first two PCs. Each point is a well, colors stand for plates. The outlier cluster is observed. On the right: examples of images from outlier wells. We see that those images are out of focus.

For the datasets above, the quality control was done to remove very noisy or out-of-focus images, as those are not able to preserve reliable phenotypic information and at the end of the day may distort the aggregated features. To do so, the features were extracted with DeepProfiler (EfficientNet-B0 model pre-trained with ImageNet dataset, see 3.4.3), then those features were aggregated by site, by well (as described in [48]) and sphering transformation (see 3.4.4.2) was applied. Then PCA was used on those aggregated profiles. The outliers observed in the PCA plots were checked manually for technical problems (Figure 17).

The datasets have class labels, such as treatments (gene perturbations in BBBC037 and compounds with concentrations for BBBC022 and BBBC036). In the case of BBBC036 and BBBC022 datasets, the treatments which were present more than once were filtered, leaving only entries with maximum concentrations. In the downstream analysis, the superclass annotations matter: gene signaling pathways (BBBC037 dataset) or mechanisms of action (for BBBC036 and BBBC022 datasets). The superclass annotations were used from [85] and then refined. Cell locations were obtained with CellProfiler.

3.4.2 DeepProfiler

A pipeline called DeepProfiler was developed which helps to train weakly-supervised models and extract representations of single-cells from high-throughput imaging experiments. DeepProfiler introduces a standardized workflow for utilizing convolutional neural networks for extracting single-cell features from large-scale image collections.

With DeepProfiler it is possible either to train the network and then perform feature extraction or to use a pre-trained network for feature extraction. The inputs of DeepProfiler are the images, corresponding metadata and the experiment configuration. DeepProfiler extracts the single-cell images (simple crops, DeepProfiler does not do segmentation on its own, but can cut objects if the segmentation mask is provided) from the full-sized images of the predefined size, and those are the inputs of deep learning networks. The workflow is shown in Figure 18. The extracted features can be used in the downstream analysis which is usually unique for a dataset and depends on the biological questions. Besides training and feature extraction, DeepProfiler has additional features for image compression and extraction of single-cell crops from full-sized images into separate image sets.

The framework is implemented in Tensorflow [79] (for both versions 1 and 2). The source code, documentation and discussions are available on the GitHub page (`https://github.com/cytomining/DeepProfiler/`).

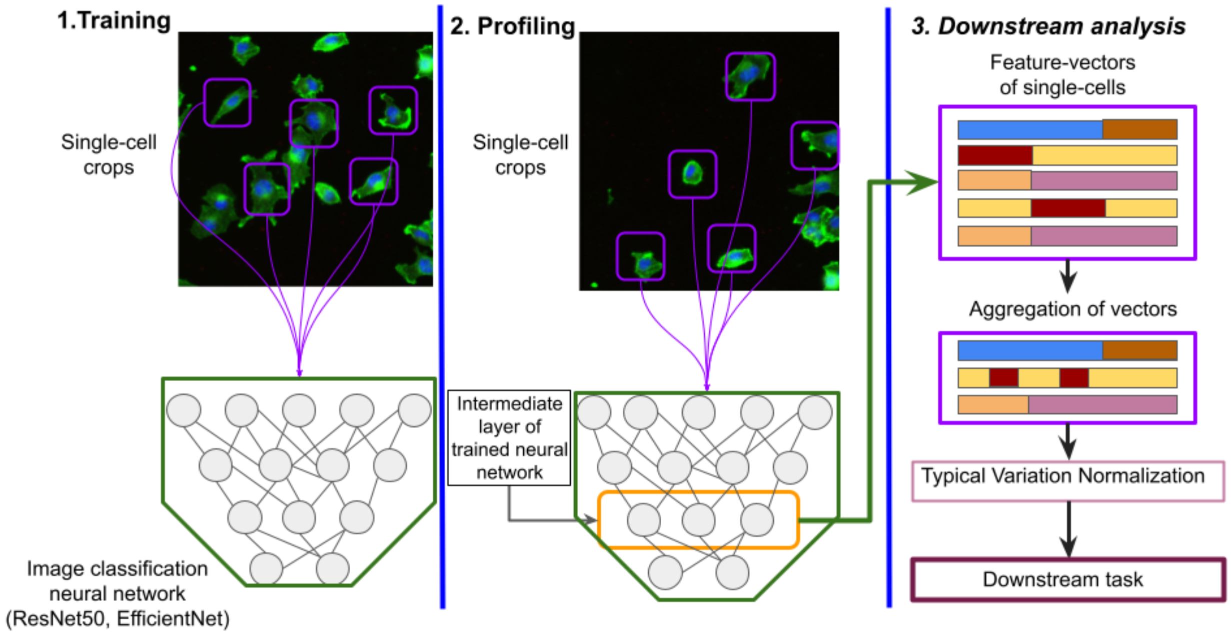

1.Training
Single-cell crops
Image classification neural network (ResNet50, EfficientNet)
2. Profiling
Single-cell crops
Intermediate layer of trained neural network
3. Downstream analysis
Feature-vectors of single-cells
Aggregation of vectors
Typical Variation Normalization
Downstream task

Figure 18: Typical usage of DeepProfiler. 1. Perform training of image classification network 2. Use the trained model to extract the representations 3. Use the representations for downstream analysis tasks. Steps 1 and 2 in the image are performed with DeepProfiler, step 3 is a user preference. The microscopy images used from the BBBC021 dataset [72].

3.4.3 Experimental setup

3.4.3.1 EfficientNet

For deep learning experiments EfficientNet [89] architecture was used, in particular the base one EfficientNet-B0. The choice was motivated by its computational efficacy and demonstrated accuracy on the ImageNet dataset [15] superior to ResNet50 [41]. EfficientNet was used in several prior publications related to cell imaging, for feature extraction and image-based profiling [87], and for training a model on a combined dataset of cellular images [90]. Some of the solutions to Recursion Pharmaceuticals cellular image classification challenge `https://www.kaggle.com/competitions/recursion-cellular-image-classification/` were based on different modifications of EfficientNet.

3.4.3.2 Experiments with pre-trained models

In this approach, pre-trained on ImageNet dataset [15]. As pre-trained networks require 3-channel input, each of the channels is replicated three times and sent to the model separately. As an input, single-cell crops of size 128 × 128 were used. The preprocessing for the used model also required a resize to 224×224 and min-max normalization adjusted to have a final input in the range $[-1, 1]$. The features were extracted from the $block6a\_activation$ layer. For each channel, the output dimensionality is 672 features, thus the full feature vector for the cell is 3360 features.

3.4.3.3 Experiments with weakly supervised learning

Training and the following feature extraction were conducted with DeepProfiler. The inputs are pre-cropped images of single-cells, saved as a stripe of five channels and reshaped during training, so the input to the network is 128 × 128 × 5. During training the augmentations were used:

- Random crop and resize with 50% probability, the size of the crop is not less than 80% of the original size and then it is resized back.
- Random horizontal flip and then random rotation (90 degree-based).
- Color changes: brightness (up to 10% deviation from the original) and then contrast (up to 20% deviation from the original). Each channel is processed separately in both steps.

As the number of single cells varies from treatment to treatment, auto-balancing is done in each epoch of training. For all datasets, the parameters were: categorical cross-entropy loss, batch-size 32, a constant learning rate of 0.005 with SGD optimizer, augmentations on, no label smoothing and 30 epochs. The models are initialized with ImageNet pre-trained weights.

Two setups for splitting the data to training and validation were used:

- Leave-plates-out - the single-cells from one subset of plates are used for training, and from another for validation.

- Leave-cells-out - the single-cells from each plate and each well are used both in training and validation, approximately 60% of cells from each well are used in training, 40% in validation.

Using trained models, features were extracted from *block6a_activation* layer (feature vector size is 672).

3.4.3.4 Computational efficacy

The computational efficacy was estimated in terms of computation clock-time and storage space needed versus classical features. The proposed approach is faster, than the classical as it utilizes GPU parallelization. NVIDIA V100 was used for all deep learning experiments. Training time on average across the datasets takes 3.3 hours, profiling approximately takes 0.58 hours with the pre-trained model and 0.22 hours per plate with trained models. The pre-trained model takes more time as five inference passes are needed for an image. Comparison is available in Figure 19. The price is not compared here, though commonly cloud GPU computation is more expensive than on CPUs.

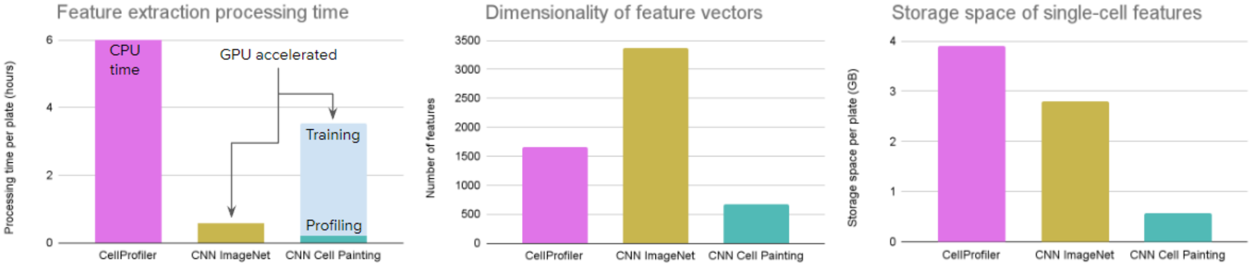


Figure 19: Computational cost of profiling strategies. The source of the figure [82].

3.4.4 Profiling workflow and evaluation

3.4.4.1 Feature aggregation and similarity matching

The feature aggregation is a pipeline to get treatment-level profiles from single-cell profiles [48]. There are intermediate levels, such as field-of-view (image)-level and well-level. The feature vectors of single-cells are aggregated using the median to image-level, and then, image-level profiles are aggregated using mean to well-level profiles. In this work, feature aggregation steps are the same, disregarding the source of the features either CellProfiler or deep-learning models.

To assess the similarity between treatments different metrics can be used [48], here the cosine similarity is used (also used in other works [91]).

3.4.4.2 Batch correction using sphering transform

One attempt for reduction of unwanted technical variation is Typical Variation Normalization (TVN) proposed in [92], also used in [93]. It computes axes of variation using principal component analysis on negative control well-level profiles. The obtained axes are normalized, which makes axes of large variation be reduced and axes of low variation to be amplified. The normalization transformation is then applied to all well-level profiles.

Here, the ZCA-sphering transformation is used similarly as TVN. As an input, the matrix of well-level negative control features $X^{n \times d}$ is used, where n is the number of control wells and d is the feature vector dimension. The covariance matrix for X is $\Sigma = \frac{X^T X}{n}$, its eigendecomposition is $Q = U \Delta U^T$, where Δ are eigenvalues. To obtain a final ZCA-transformation [94] (sphering), is the following $U(\Delta + \lambda)^{-1} U^T$, where λ is a regularization parameter.

3.4.4.3 Evaluation and metrics

As described in [16], the evaluation task is to check if the most similar treatments according to the similarity metric belong to the same gene pathway or mechanism of action (MoA). Several metrics were used for evaluation, all of them briefly described below. In further text, *query treatments* are referred to as treatments which have at least two treatments in the same MoA or pathway and are used as queries in the ranking task.

First metric that was used is folds of enrichment. The odds ratio is calculated, similar to [16], the main difference is that here it is done only for 1% threshold and this is done for each query treatment separately. Then, the simple mean of obtained values is computed.

As another metric, an interpolated precision-recall curve and mean average precision for the ranking task was used. This metric is calculated in the following way: each treatment is a query, and the top similar treatments to the query treatment are checked. Precision@K in this ranking task is the ratio of treatments that belong to the same MoA/pathway as the query out of the top K most similar treatments. The same intuition is applicable for *Precision@Recall*: for one treatment (query) we go through all the treatments ranked by distance until we reach a recall of 1 (find all positive matches). As each MoA/pathway has a different number of associated ground truth treatments, *Precision@Recall* is interpolated to cover the max number of recall points, interpolated precision is defined as $p_{interpolated}(r) = max_{r' \geq r} p(r')$ [95]. Average precision (area under interpolated Precision-Recall curve) is a mean of $p_{interpolated}$ at all recall points. mAP here is a simple mean of average precisions for individual queries.

Hits in the top 1% metric simulates the task of finding a 'hit' in the most promising candidate treatments. The metric is applicable on several levels of profiling:

- Treatment-level: measure the number of query treatments which have a treatment (response) with the same MoA/pathway among the top 1% of most similar response treatments.

- Well-level: The well profile is used as a query (all treatments can be used). The number of treatments, which have query wells with the response wells of the same treatment among the top 1% of most similar wells is computed.

- Image-level: The image profile is used as a query (all treatments can be used). The number of treatments, which have query images with the response images of the same treatment among the top 1% of most similar images is computed. The images of the same well as query image are excluded from the possible responses.

3.4.5 Strong treatment selection and combined Cell Painting dataset

To expand the potential feature-space both with biological and technical variation the treatments resulting into a strong phenotypes were collected from five Cell Painting datasets. Strong treatment here is defined as one to produce a phenotype which is different to a phenotype of untreated cells. To estimate the strength of the phenotype, CellProfiler feature space is used (batch-corrected with regularization parameter $1e-2$) and measure the Euclidean distance between the well-level profiles of treatments and negative controls (Algorithm 1).

Algorithm 1 Strong treatments selection

```
for each $p$ in $Plates$ do
    Calculate median profile of negative controls in the plate - $MCP_p$
    Calculate Euclidean distance between the treatment well-level features and $MCP_p$, get the distances $EDT_p$
    Calculate Euclidean distance between the negative control well-level features and $MCP_p$, get the distances $ECT_p$
    Calculate $\mu$ and $\sigma$ of $ECT_p$
    Use $\mu$ and $\sigma$ to Z-score $EDT_p$
end for
for each $t$ in $Treatments$ do
    $Z(t) \leftarrow \sum_p^{Plates} EDT_p(t)$, where $Z$ stores the final distances for each treatment
end for
```

Selection of the strong treatments for the combined Cell Painting dataset did include the following steps:

- Select top 500 strongest treatments according with Algorithm 1 from BBBC022.

- Intersect those with BBBC036, include the intersection into the combined Cell Painting dataset.

- Additionally select 50 from BBBC022 and 62 from BBBC036 strongest treatments and add them to the dataset.

- Select 7 random treatments from LINCS, from the top 20 (by a number of associated treatments) MoAs, and add them to the dataset.
- Select 28 overlapping wildtype genes between BBBC043 and BBBC037 dataset and add to the dataset.
- Additionally select 29 strongest treatments from BBBC037 and 32 from BBBC043 and add them to the dataset.
- Filter out classes with less than 100 cells.
- Add controls one class for compound screening datasets (BBBC022, BBBC036, LINCS) and another for gene overexpression datasets (BBBC037, BBBC043). Control cells from BBBC036 and LINCS are partially selected.

Resulting dataset contains 8.3 million single cells from 232 plates, 488 treatments and 2 types of negative controls. More information about the dataset is in the Figure 20.

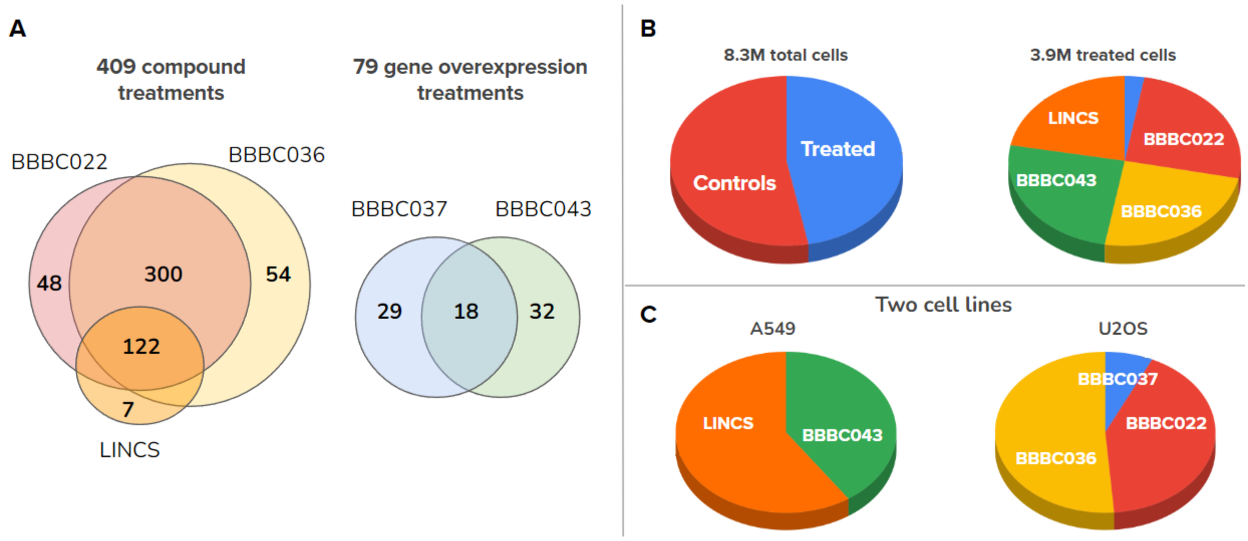


Figure 20: Description of combined Cell Painting dataset. A. Treatment sources in the combined dataset. B. Treated vs control cells distribution and sources of treated cells. C. Sources of cells inside per cell line. The source of the figure [82].

3.4.6 Causal relations in screening experiments

By applying different treatments to cells, biologists are trying to perturb their state and observe the response. The causal graph for that kind of experiment includes four variables: treatments T, images O, phenotypes Y and batch-effects C. In causality modeling terms, those are interventions, observations, outcomes and confounders respectively. T and O are observed variables, while Y and C are latent variables. The goal is to learn Y, a multidimensional representation of treatment, which could be used in the further downstream task. To be useful in the downstream analysis task, Y should encode biologically relevant representation, though the reality is that technical variation, the batch-effects C affect all other elements of this causal model. C affects images by technical variation in the image acquisition process, treatments by plate-layout design (the template of the positioning of treatments in plates in the screening experiment) and phenotypes by environmental conditions. The relations are shown in the graph (Figure 21).

Treatment is expected to be the main cause to change in the phenotype of the cell. To extract the representation of phenotypic outcome, WSL is used with the pretext task of treatment classification. The representations extracted from the intermediate layers of CNNs encode all visual variation, in this case, both batch-effects and phenotypes. WSL together with batch correction would help to disentangle phenotypic variation from technical.

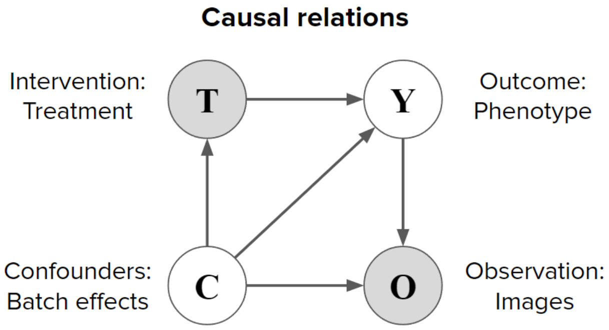


Figure 21: Causal model for screening experiment. T stands for treatments (interventions), O for images (observations), Y for phenotypes (outcomes) and C for batch-effects (confounders). The source of the figure [82].

3.4.7 Results and observations

The subsection discusses the results obtained with WSL on the combined Cell Painting dataset *CNN Cell Painting* model and models trained on the benchmark datasets. Pretrained model on ImageNet (also referred to as *CNN ImageNet*) dataset and classical features extracted with CellProfiler serve as baselines.

3.4.7.1 Learned representations sharpen biological features

CNN Cell Painting model performs better in quantitative evaluation than both baselines in the evaluation task (Figure 22, cyan points). That was expected as manually engineered features might miss some information and the ImageNet model is trained on a completely different domain and not optimized for the images of cells. The models trained only on the corresponding benchmark datasets did not show a consistent improvement in their performance against the baselines (Figure 22, green points).

For qualitative assessment, UMAP projection [96] of feature space obtained with *CNN Cell Painting* was used (Figure 23). In BBBC037 dataset, treatments are grouped together according to their pathway annotations, reproducing observations from [85]. In BBBC022 and BBBC036 projections, many treatments are also grouping together according to their MoAs.

CNN ImageNet demonstrates similar or lower performance compared to CellProfiler features (Figure 22, yellow and pink points).

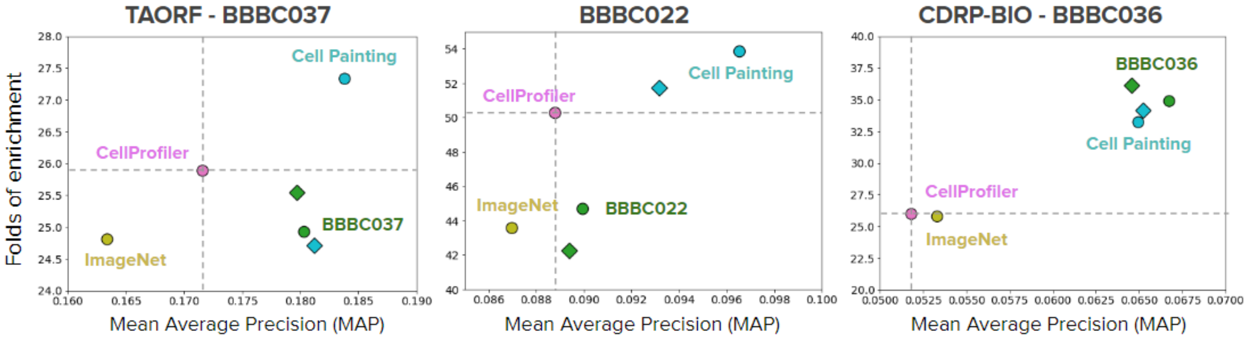


Figure 22: Quantitative performance of feature representations for three benchmark datasets in two metrics: mean average precision (X-axis) folds of enrichment (Y-axis). On the plot, the baselines are CellProfiler (pink) and CNN ImageNet (yellow), trained models: CNN Cell Painting model (cyan), trained on corresponding benchmark dataset (green). Leave-cells-out training-validation scheme shown with circles and leave-plates-out with diamonds. The source of the figure [82].

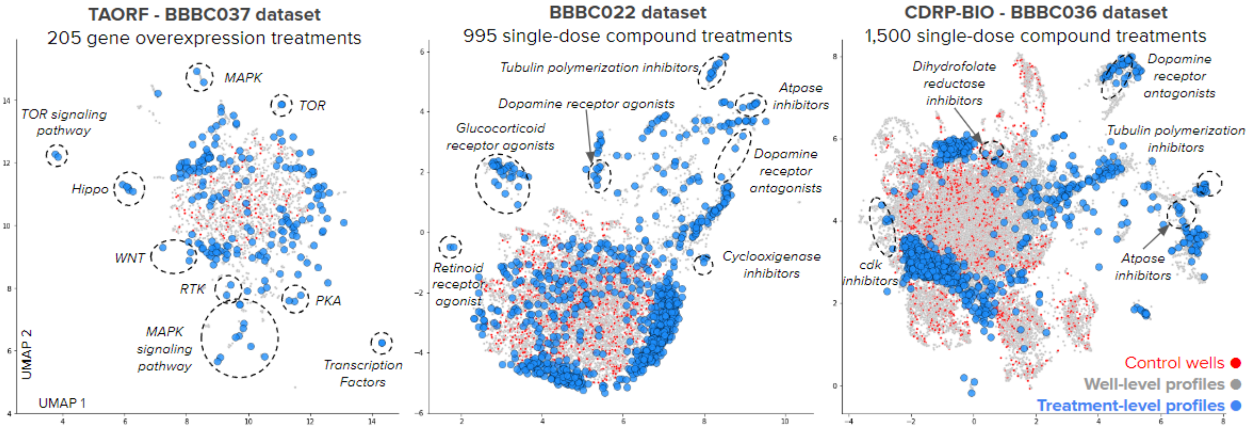


Figure 23: UMAP plots of well-level features extracted with Cell Painting CNN for three benchmark datasets. Gray points: well-level profiles of treatments, red points: well-level profiles of negative controls, blue points: treatment-level profiles. Dashed ellipses highlight clusters of treatment-level profiles with the same biological annotation. The source of the figure [82].

3.4.7.2 WSL learns both the phenotypes and the batch-effects

Different validation schemes leave-plates-out and leave-cells-out (see Experimental setup 3.4.3) help to understand the information contained in features learned from Cell Painting images. In leave-cells-out validation scheme the model as access to the full distribution of biological variation (treatments T) and technical variation (batch-effects C), yet with leave-plates-out scheme, the model still has access to the full distribution of biological variation, but only to a part of technical variation.

Major performance difference was observed in the pretext classification task for those two validation schemes. In leave-cells-out setup, the trained CNN can accurately classify single-cells from both training and validation sets, while in leave-plates-out setup, the trained model completely fails to classify single-cells in validation set (Figure 24). Nonetheless, two models trained with different validation schemes demonstrate similar performance in the downstream task (Figure 22). This observation leads to a conclusion that WSL models try to take advantage of any information that can explain the link between the images and treatments, including batch-effects. The validation performance in leave-cells-out is too optimistic (batch-effects are heavily used to build the link between observation and intervention), on the contrary, leave-plates-out validation performance is too pessimistic as in this case the model is not aware of confounding variation in validation plates.

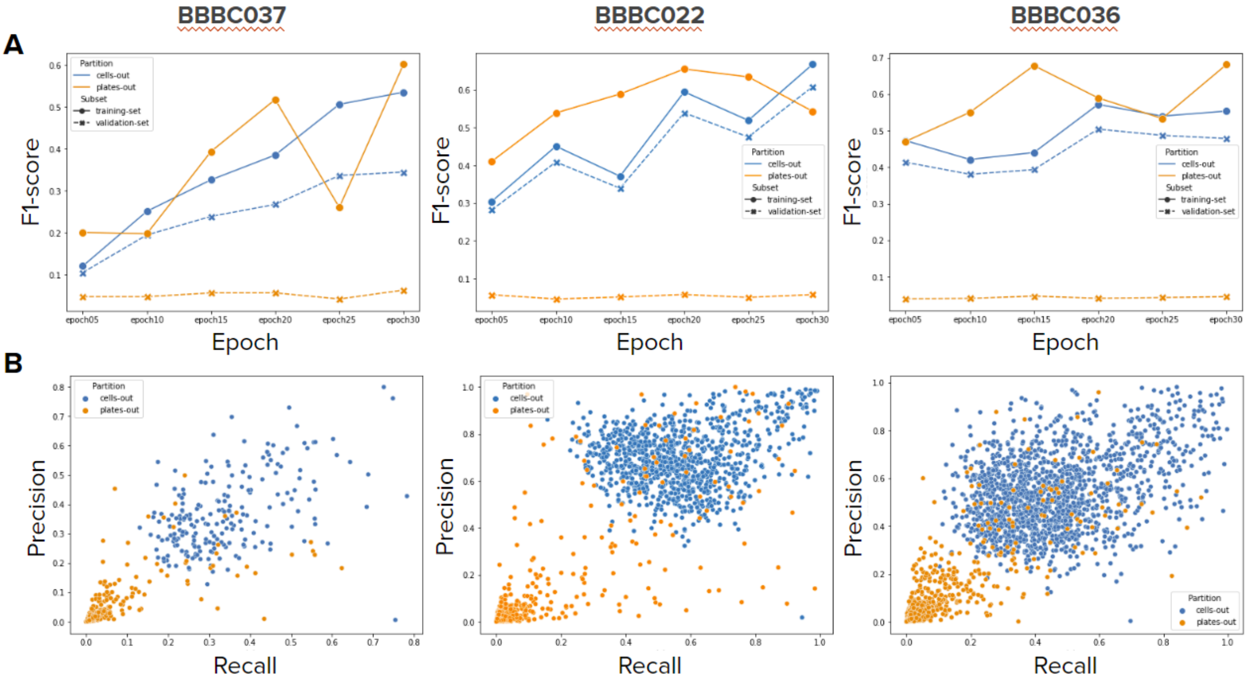


Figure 24: Classification performance in the pretext task (treatment classification) in the benchmark datasets for leave-plates-out (orange) and leave-cells-out (blue) training-validation schemes. A. F1-score for the training set (solid line) and validation set (dashed line) for every fifth epoch. B. Recall (X-axis) and precision (Y-axis) for the final checkpoint. Every point is a class (treatment, including negative control). The source of the figure [82].

3.4.7.3 Learning with strong phenotypes improves performance in the biological task

As in the previous section it was observed that controlling the distribution of confounding factors C does not change the downstream performance, now it is time to explore what happens if the phenotypic distribution Y is restricted. The intuition is that WSL minimizes an error in the pretext task by exploiting confounding factors to correctly classify treatments with a weak phenotypic response. Such treatments might have a stronger technical signal rather than a biologically relevant phenotypic signal.

The strong treatments were selected by measuring Euclidean distance between negative control and treatment profiles, obtained with CellProfiler (see section 3.4.5). That is an approximation of average treatment effect (ATE), a causal parameter for intervention outcomes. As we cannot observe the untreated (control) and treated conditions in the same cell, this can be considered only as an approximation of ATE. CellProfiler features were chosen to estimate ATE as those are non-trainable, thus can serve as independent prior.

WSL training only on strong treatments only in benchmark datasets was evaluated in leave-plates-out training-validation scheme. The results demonstrate minor performance improvement against training on full datasets (Figure 25, blue points).

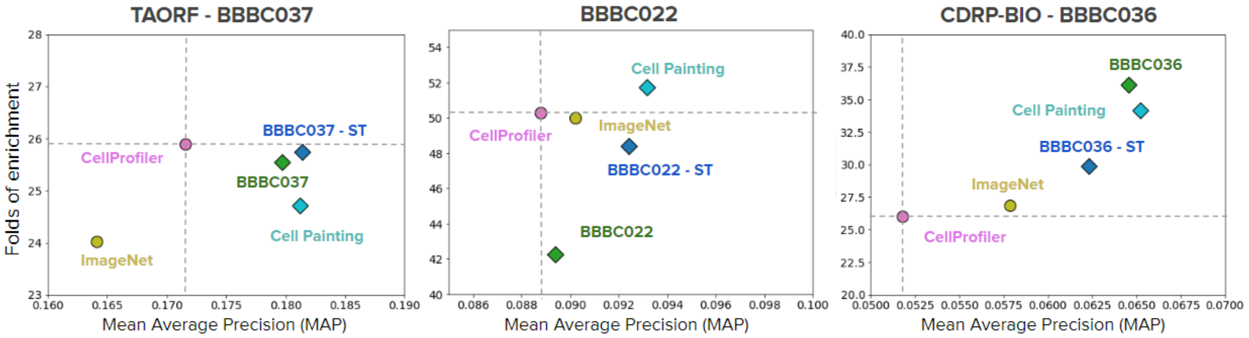


Figure 25: Quantitative performance of feature representations for three benchmark datasets in two metrics: mean average precision (X-axis) folds of enrichment (Y-axis). On the plot the baselines are CellProfiler (pink) and CNN ImageNet (yellow), trained models: CNN Cell Painting model (cyan), trained on corresponding benchmark dataset (green), trained on strong treatments from corresponding benchmark dataset (blue). All training experiments used leave-plates-out training-validation scheme. The source of the figure [82].

3.4.7.4 Diverse experimental conditions result in improved representations

The combined Cell Painting dataset was created to maximize both phenotypic (Y) and technical (C) variation by combining the treatments with the strongest resulting phenotypes from five datasets. Training on this dataset consistently improves performance over other approaches (Figure 22, cyan points), which means that this model can disentangle Y and C more efficiently. The most important outcome is that this model was trained once and could be used at all benchmarks without additional training.

3.4.7.5 Batch-correction is a crucial post-processing step

The role of batch-correction (see Batch correction using sphering transform 3.4.4.2) is to reduce the impact of confounding technical factors C. It is crucial for all representations tested: classical features, features extracted with pre-trained and trained CNNs. Mean average precision improves up to 90% versus raw features (see Figure 26). Also, using the effect of batch-correction can be observed qualitatively (Figure 27). Still, this does not mean that the batch-effects are eliminated and further research is needed to learn how to disentangle technical and biological information in representations.

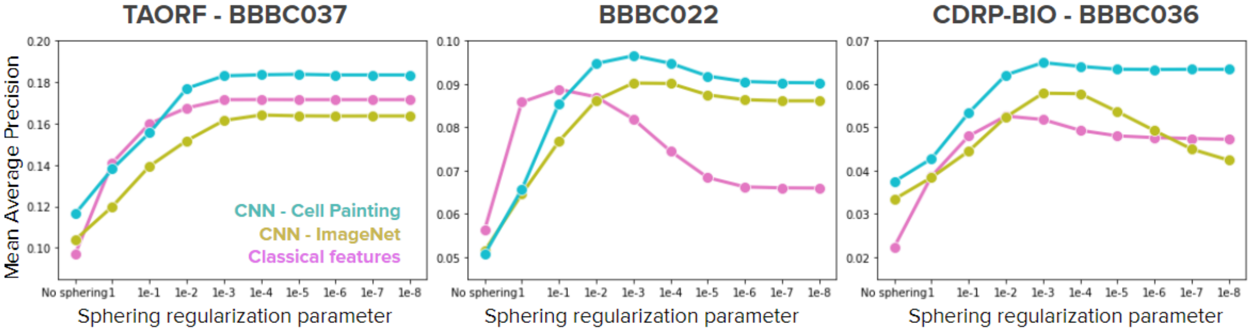


Figure 26: Mean average precision for sphering with different regularization parameters (smaller regularization term, more correction applied) for three datasets. For each dataset CellProfiler features (pink), ImageNet CNN (yellow) and Cell Painting CNN (cyan) are evaluated. The source of the figure [82].

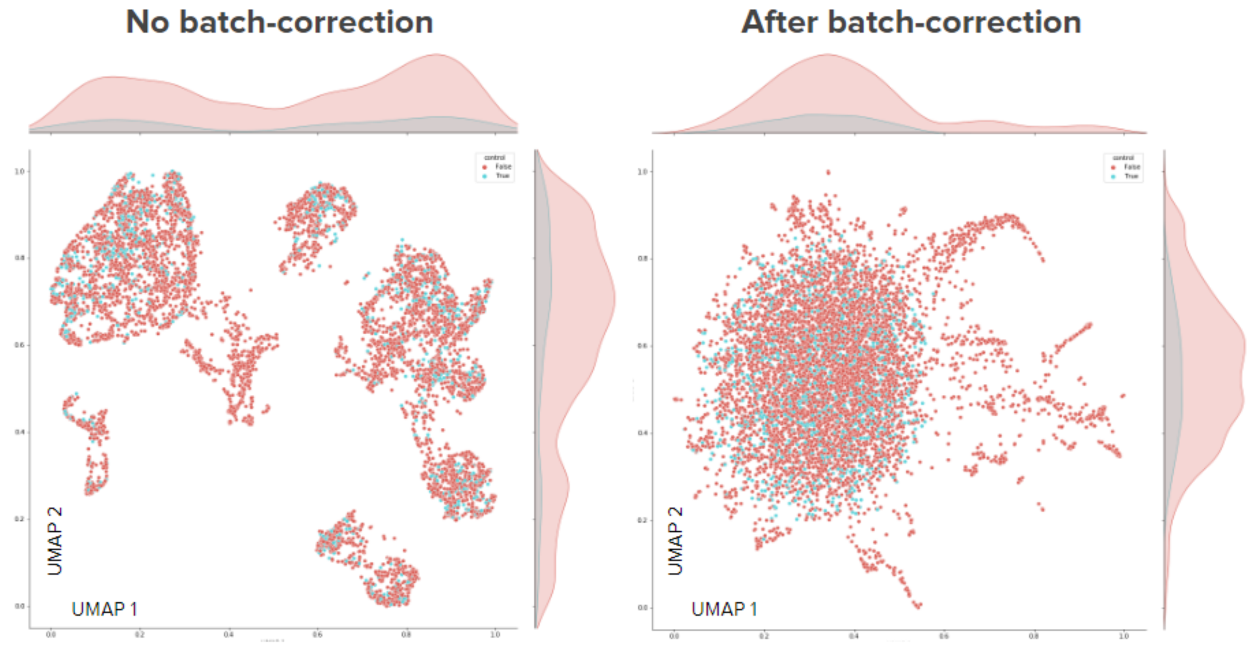


Figure 27: The qualitative effect of batch-correction in the UMAP plots. The left plot shows the UMAP representation of the BBBC022 dataset without batch-correction and the right plot after batch-correction. The points are the embeddings of well-level profiles (cyan - negative controls, red - treatments). Density plots are on the top and the right sides of the plots. Features were extracted with *Cell Painting CNN* model.

3.5 Predicting compound activity from phenotypic profiles and chemical structures[2]

This section briefly discusses the content of [97].

Drug discovery is an expensive and very slow process, there are too many theoretically possible compounds to test in a real physical experiment. Even though pharmaceutical companies may afford to test millions of compounds in their experiments, this only covers a small fraction of possible compounds. Besides, to test those compounds the expensive (as those contain valuable biological materials: primary cells, antibodies, etc.) phenotypic assay systems are used to identify candidate compounds. Finally, this process is time-consuming and requires the time of experts to run the assays.

To reduce the costs of screens in drug discovery, there is possible room for computational methods, for instance, modern deep learning might allow accurate prediction of assay activations for compounds. The previous works tried to use machine learning methods with morphology data only [98] [99].

In this project, the aim is to evaluate the predictive power of the representations of chemical structures, cell morphology profiles and gene expression profiles, to predict assay outcomes computationally at a large scale. The hypothesis is that the predictive capabilities of those data sources are complementary and those data sources could be used together to further increase the success rate of the drug screening process. Besides, the basic data fusion techniques are tested, although it is not the focus of the project and this question might be investigated further.

3.5.1 Materials and methods

The dataset is composed of four parts: assay-compound interaction matrix, morphology profiles, gene expression profiles and representations of chemical structures. All the information was collected from assays from the drug discovery experiments conducted at Broad Institute [86].

Assay-compound interaction matrix is the main piece of the dataset. Rows are compounds (represented as SMILES strings) and columns are assays. The cells are filled with 1 (hit) and 0 (no hit) and can be blank (this compound was not tested with the assay). “Hits” and “no hits” combined are also referred to as readouts. Only a fraction of compounds was tested in each particular assay, which means that the matrix is quite sparse. Initially, the matrix contained 496 assays, but filtered using the following procedure:

- Applied all pan-assay interference (PAINS) filters [100] implemented in RDKit, which removed 786 compounds, resulting in 16,210 compounds.

- Removed all assays without hits, thus the number of assays decreased from 496 to 437.

[2] The article is online as a pre-print and submitted to a journal

- Calculate intersection-over-union (IoU) for the hits between assays to find out the assays which carry the redundant information. The IoU matrix (437 × 437) was thresholded by 0.7 and then hierarchical clustering was applied with the cosine distance metric, which was used for filtering.
- Final removal of frequent hitters, defined as compounds that are positive hits in at least 10% of the assays (30 assays or more) and final cleaning of assays without any hit. In the end, the final dataset consists of 16,170 compounds and 270 assays.

Most of the assays in the final dataset are cell-based, other represented types of assays are biochemical, bacterial and yeast assays and also there are poorly represented categories of assays, such as fungal, homogeneous, viral and worm (Figure 28).

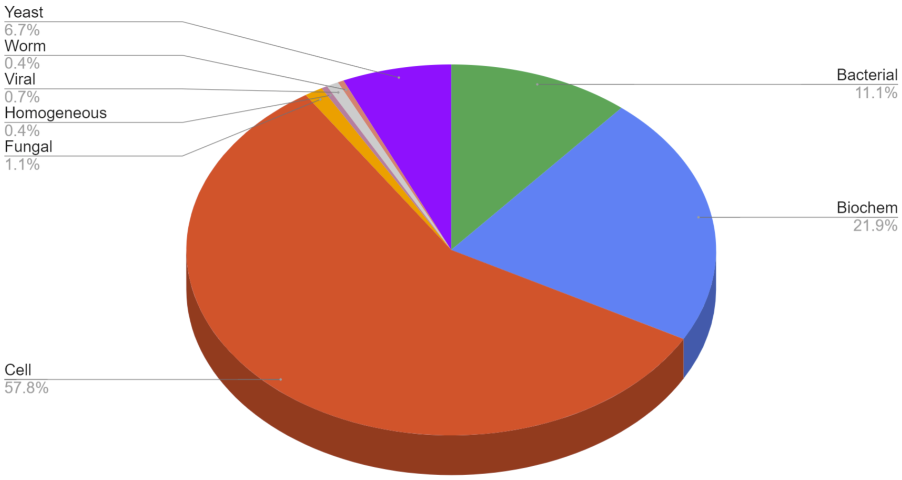


Figure 28: Distribution of the assay types in the final dataset. The source of the figure [97].

The Cell Painting assay [10] [47] [101] [102] experiments were run to obtain high-resolution five-channel images. Those images were processed with CellProfiler software to segment and obtain $\sim$ 1700 morphological features at the single-cell level. Those were then aggregated to the well-level as in [48]. On the well-level profiles, sphering (see also 2.2) was applied to correct for batch effects. To calculate the sphering transformation, DMSO wells from all plates were used. Then the profiles were aggregated to the treatment level (referred to as MO, except for Table 3.5.2 and Table 2). The experiments were also performed with the features without sphering, though the additional performance boost gained for the morphological features, in that case, may be biased by batch effects (Figure 29).

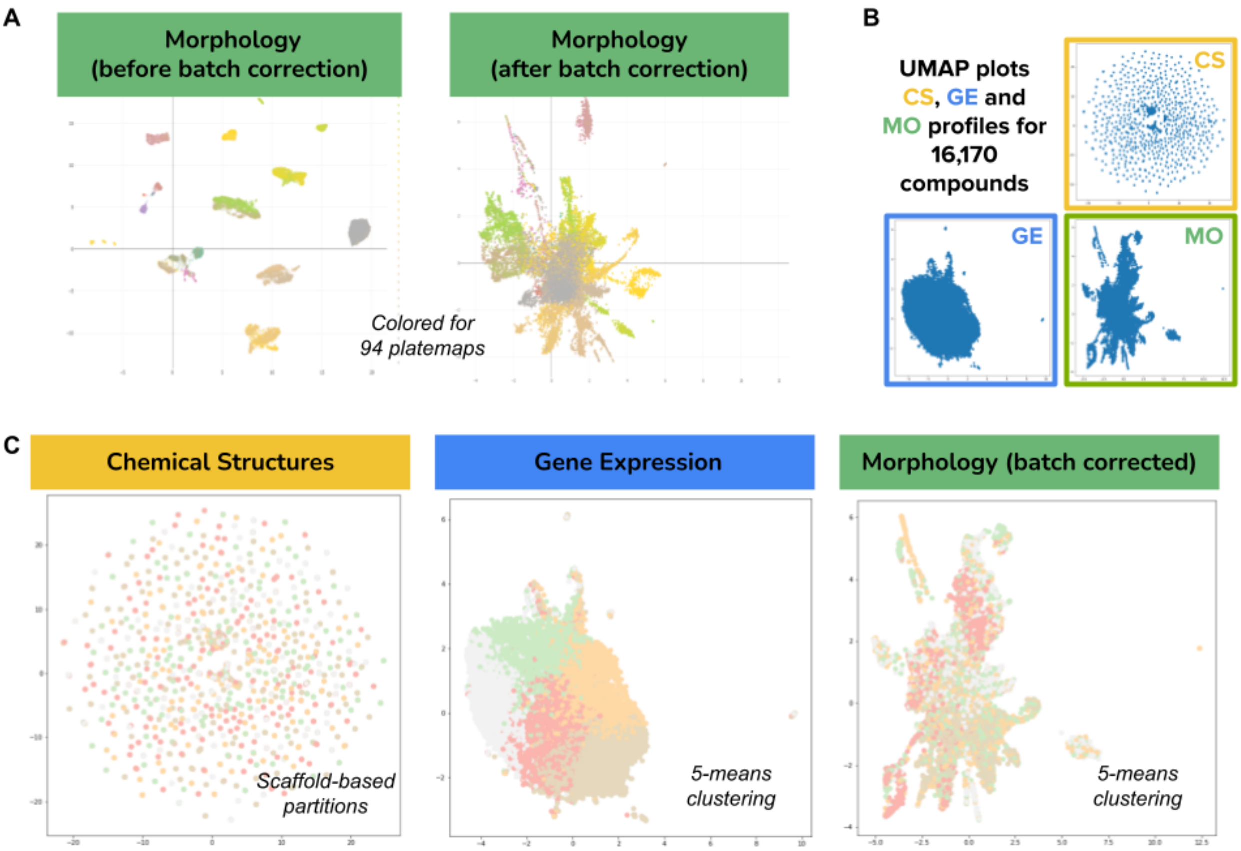


Figure 29: Compound embeddings in three different modalities. Visualizations are built with UMAP. A. The morphology feature space originally was grouped by technical variation (plate maps), which was corrected using the sphering. The color palette for the 94 plate maps is continuous and may have similar tones for consecutive plates. B. Compound embeddings in three different modalities C. The same embeddings as in B, colored by clusters obtained for cross-validation experiments (see “Experiments and results section”). The source of the figure [97].

3.5.2 Experiments and results

The experiments were conducted for several train-test split approaches. All the train test approaches share the same idea that we want to predict assays-compound interaction for compounds that are distinct relative to training data. From the practical perspective, there is little value in searching for similar chemical structures for the one with known activity. The closest train-test split to such a real-world scenario is a scaffold-based split (for 5-fold cross-validation) achieved with Bemis-Murcko clustering [58] [103].

In addition to scaffold-based train-test splits, the splits based on morphological and gene expression features are constructed. For gene expression-based splits the gene expression features were clustered and for morphology-based splits the batch-corrected morphology features were clustered (for 5-fold cross-validation) using same-size K-Means clustering (implementation [104]), see clustering in Figure 29.

As a primary metric, the area under receiver operating characteristic curve (AUROC) was used. The main results are reported for 0.9 threshold as it was used in earlier works about assay-compound activity prediction [60] [105] [106]. As a secondary metric, an area under the precision-recall curve (AUPRC) was used.

The models were trained with a logistic regression loss function for each assay, and total loss is a sum of losses for each assay. The mini-batch contains information about 50 compounds. If there is no ground-truth readout for assay-compound interaction, it is ignored for gradient update. In each training, the hyperparameter optimization was run before the training (see 3.5.1).

Our results show that morphology could accurately predict the largest number of assays with the median $AUROC > 0.9$ over cross-validation splits (28 for morphology, 19 for gene expression and 16 for chemical structures), see Figure 31. Although, for lower AUROC thresholds (0.7) chemical structures tie with morphology (also see Figure 33). Interestingly, all three modalities share zero well-predicted assays (Figure 31) and each pair of modalities share a few common well-predicted assays, which means that different data sources contain significantly complementary information.

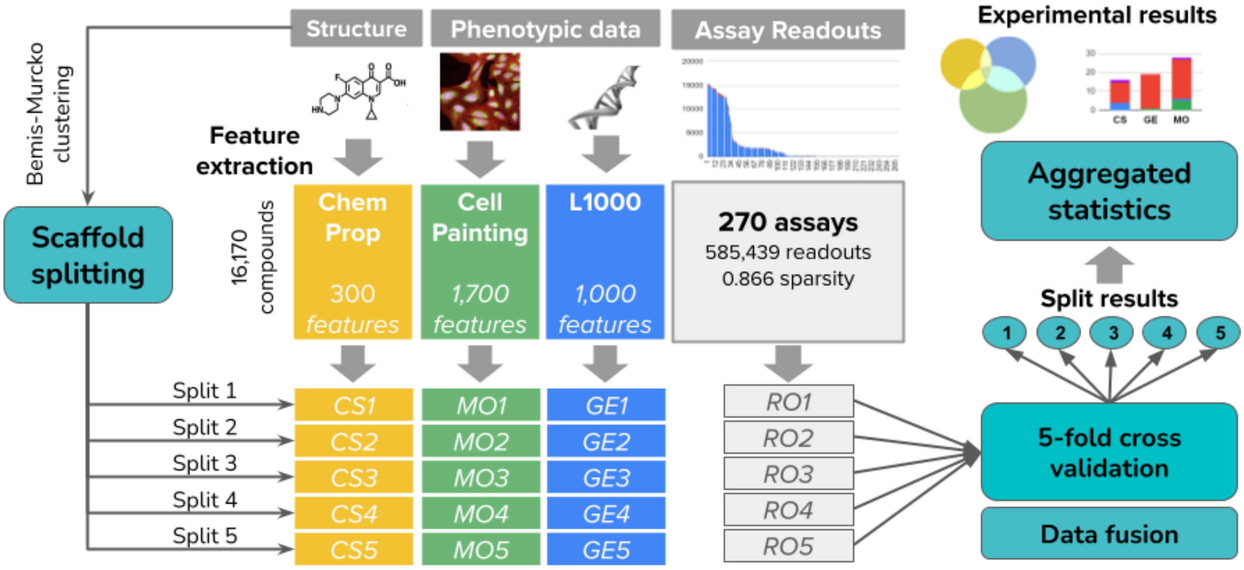
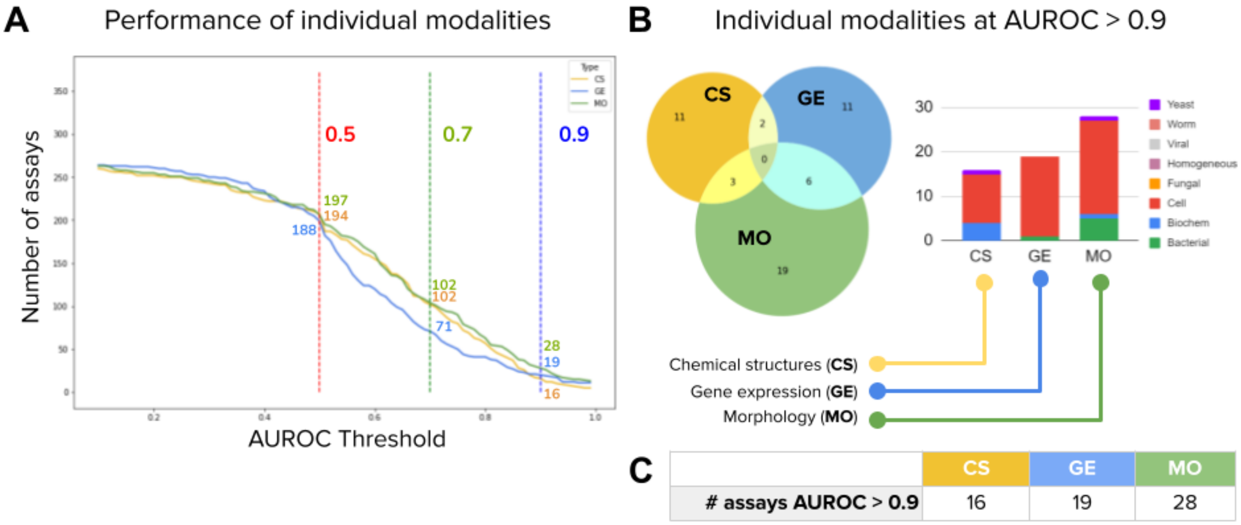


Figure 30: Illustration of experimental setup. The source of the figure [97].



	CS	GE	MO
# assays AUROC > 0.9	16	19	28

Figure 31: A. Performance of individual modalities measured as the number of assays (vertical axis) predicted with AUROC above a certain threshold (horizontal axis). B. The Venn diagrams show the number of accurate assays (median $AUROC > 0.9$ over cross-validation splits) that are common or unique to each profiling technique. The bar plot shows the distribution of assay types correctly predicted by single profiling modalities. C. Number of well predicted (median $AUROC > 0.9$ over cross-validation splits) assays by each modality. The source of the figure [97].

Not only one modality can be used for predicting the assay-compound interaction. To combine modalities into a single predictor, two approaches were used: a) ***Early fusion*** - the feature vectors are concatenated into a single vector and used as an input for the neural network. b) ***Late fusion*** - for each modality the separate model is trained and then the prediction scores are aggregated, using the maximum probability among predictions for each compound-assay pair.

According to our experiments (Table 2), early data fusion did not provide any additional performance, in fact, it did hurt the performance. Our results for individual modalities did show that they do not share many well-predicted assays in common (Figure 31), and when the feature vectors are combined, additional noise to the assays is introduced, as assays can be well predicted by one modality but cannot be predicted by another. Late fusion works better in practice, though according to the results, the performance gain is minor at best (31 well-predicted assays with CS+MO combination vs 28 with MO only). The fusion approaches in the demonstrated tests are quite simple and more investigation for more effective fusion techniques is needed. As an additional metric, retrospective performance was measured. It is a simulation of the best possible data fusion. In this analysis, know the predictions are known in advance. Usage of fused with individual modalities can give 7-17% of performance boost (Figure 32).

	Scaffold-based splits — Real world setting					
Avg. assays tested: 233.2	MO	MO-BC	GE	GE-S	CS-GC	CS-MF
Mean AUPRC	**0.261**	0.252	0.234	0.231	0.232	0.223
Mean AUROC	**0.657**	0.637	0.592	0.587	0.630	0.610
$AUC > 0.5$	**160.0**	151.4	139.2	138.8	150.2	146.8
$AUC > 0.7$	**91.2**	83.2	57.2	59.4	88.4	81.6
$AUC > 0.9$	27.0	**28.0**	21.8	18.4	21.6	21.0
	Gene expression splits (simulation)					
Avg. assays tested: 232.0	MO	MO-BC	GE	GE-S	CS-GC	CS-MF
Mean AUPRC	**0.263**	0.248	0.222	0.201	0.246	0.244
Mean AUROC	**0.664**	0.642	0.577	0.561	0.647	0.658
$AUC > 0.5$	155.6	150.2	127.6	127.2	153.2	**157.4**
$AUC > 0.7$	94.4	86.2	45.4	46.6	94.2	**99**
$AUC > 0.9$	**27.4**	23.6	14.2	12.6	22.6	22.4
	Morphology(bc)-based splits (simulation)					
Avg. assays tested: 179.8	MO	MO-BC	GE	GE-S	CS-GC	CS-MF
Mean AUPRC	0.224	0.207	0.199	0.198	0.225	**0.245**
Mean AUROC	0.634	0.600	0.562	0.564	0.631	**0.652**
$AUC > 0.5$	142	128.6	125.4	126.2	140.8	**143.6**
$AUC > 0.7$	**72.8**	63.0	49.2	49.2	81.0	82.6
$AUC > 0.9$	**21.6**	17.0	14.4	13.6	19.4	22.6
	Random splits (simulation)					
Avg. assays tested: 232.4	MO	MO-BC	GE	GE-S	CS-GC	CS-MF
Mean AUPRC	**0.259**	0.247	0.234	0.228	0.244	0.242
Mean AUROC	**0.670**	0.643	0.601	0.595	0.659	0.651
$AUC > 0.5$	**163.6**	154.2	145.6	144.0	157.6	157.8
$AUC > 0.7$	**97.2**	88.4	61.8	66.0	94.8	94.0
$AUC > 0.9$	**26.2**	22.0	20.4	17.4	25.8	23.4

Table 1: Results of 5-fold cross-validation experiments. The tables present the mean results of 5-fold cross-validation experiments according to different data partition approaches. The metrics are: Mean AUPRC for 5 splits, Mean AUROC for 5 splits, mean counts of the predicted assays thresholded by AUROC ($AUC > 0.5$, $AUC > 0.7$, $AUC > 0.9$) for 5 splits. Sources of data used: MO: morphological features without batch-correction. MO-BC: morphological features with batch-correction. GE: Gene expression features. CS-GC: graph convolutional (GC) features. CS-MF: Morgan fingerprints. An average number of assays in the test set differs between modalities, as it is impossible to evaluate an assay without hits in the test set (which are different as different train-test split approaches were used). The source of the table [97].

Baseline: independent modalities (scaffold-based partitions)

	MO		GE		CS	
	Mean	Std	Mean	Std	Mean	Std
Mean AUPRC	0.252	0.021	0.234	0.038	0.232	0.036
Mean AUROC	0.637	0.021	0.592	0.034	0.630	0.018
$AUC > 0.5$	151.4	13.502	139.2	13.773	150.2	13.255
$AUC > 0.7$	83.2	11.100	57.2	16.316	88.4	6.066
$AUC > 0.9$	28.0	4.848	21.8	8.198	21.6	6.229

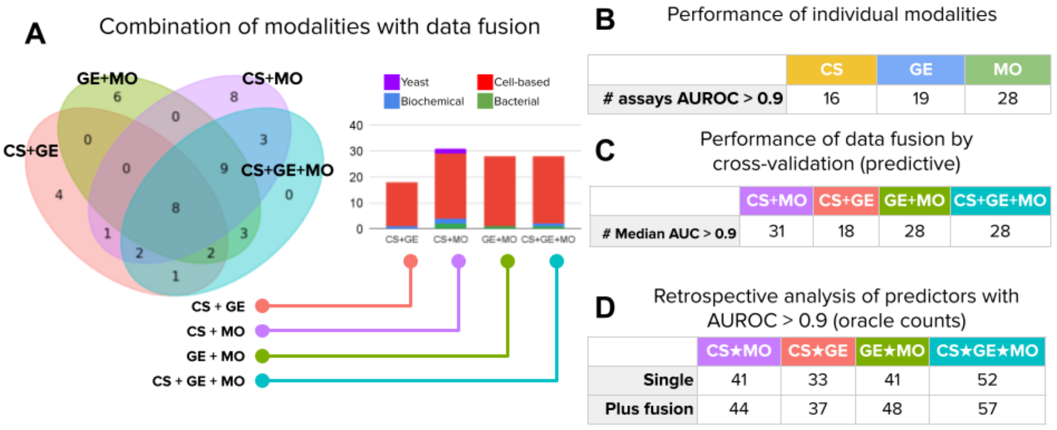
Early fusion — concatenation (scaffold-based partitions)

	GE-MO		MO-CS		GE-CS		GE-MO-CS	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Mean AUPRC	0.214	0.045	0.251	0.021	0.219	0.028	0.221	0.021
Mean AUROC	0.586	0.038	0.632	0.031	0.577	0.061	0.582	0.038
$AUC > 0.5$	138.8	18.377	151.8	19.905	138.6	26.773	137.2	22.928
$AUC > 0.7$	59.2	12.215	87.8	15.531	63.4	21.663	59.8	14.516
$AUC > 0.9$	16.0	4.743	23.6	4.159	17.0	2.292	20.4	4.278

Late fusion — max pooling (scaffold-based partitions)

	GE-MO		MO-CS		GE-CS		GE-MO-CS	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Mean AUPRC	0.261	0.026	0.267	0.034	0.251	0.039	0.265	0.032
Mean AUROC	0.652	0.028	0.661	0.027	0.645	0.026	0.665	0.031
$AUC > 0.5$	157.4	11.845	157.8	13.773	155.6	16.637	159.0	15.017
$AUC > 0.7$	86.0	9.670	98.8	7.430	87.0	9.566	96.4	10.877
$AUC > 0.9$	29.4	6.618	29.4	5.128	23.8	8.843	28.0	5.148

Table 2: Performance of individual and combined modalities for models trained with scaffold-based splits. The metrics are: Mean AUPRC for 5 splits, Mean AUROC for 5 splits, mean counts of the predicted assays thresholded by AUROC ($AUC > 0.5$, $AUC > 0.7$, $AUC > 0.9$) for 5 splits. Standard deviations are in a separate column. The source of the table [97].



Performance of individual modalities

	CS	GE	MO
# assays AUROC > 0.9	16	19	28

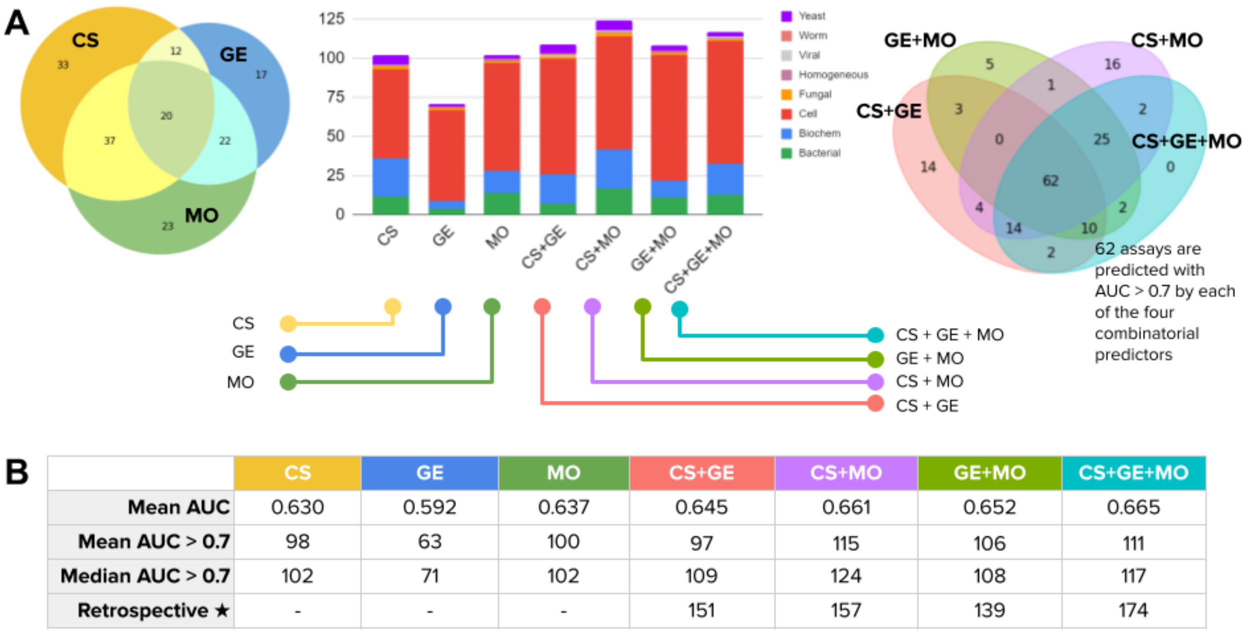
Performance of data fusion by cross-validation (predictive)

	CS+MO	CS+GE	GE+MO	CS+GE+MO
# Median AUC > 0.9	31	18	28	28

Retrospective analysis of predictors with AUROC > 0.9 (oracle counts)

	CS★MO	CS★GE	GE★MO	CS★GE★MO
Single	41	33	41	52
Plus fusion	44	37	48	57

Figure 32: Accurately predicted assays (median AUROC over splits is higher than 0.9). A. Venn diagram of accurately predicted assays using late fusion (left), bar plots show the distribution of accurately predicted assay types with late fusion (right). B. Number of accurately predicted assays per individual modality. C. Number of accurately predicted assays for combined modalities with the use of late fusion. Counts for median and mean AUROC over splits. D. Number of accurately predicted assays for retrospective analysis. “Single” is a simple union of the accurately predicted assays with individual modalities. “Plus fusion” is a union of accurately predicted assays with individual modalities plus the combined late fusion predictor. The source of the figure [97].



62 assays are predicted with AUC > 0.7 by each of the four combinatorial predictors

	CS	GE	MO	CS+GE	CS+MO	GE+MO	CS+GE+MO
Mean AUC	0.630	0.592	0.637	0.645	0.661	0.652	0.665
Mean AUC > 0.7	98	63	100	97	115	106	111
Median AUC > 0.7	102	71	102	109	124	108	117
Retrospective ★	-	-	-	151	157	139	174

Figure 33: Predicted assays with moderate accuracy (median AUROC over splits is higher than 0.7). A. Venn diagram of predicted assays with individual modalities (left), bar plot of predicted assay types by individual modalities and late fusion (center), Venn diagram of predicted assays with late fusion (right). B. Performance of individual modalities and late fusion. The metrics are: Mean AUC for 5 splits, mean counts of the predicted assays thresholded by AUROC ($AUC > 0.7$) for 5 splits. The source of the figure [97].

	CS	GE	MO	CS+GE	CS+MO	GE+MO	CS+GE+MO	Evaluated assays
Cell-based	7.05%	11.54%	13.46%	10.90%	16.03%	17.31%	16.67%	156
Biochemical	6.78%	0.00%	1.69%	1.69%	3.39%	0.00%	1.69%	59
Bacterial	0.00%	3.33%	16.67%	0.00%	6.67%	3.33%	3.33%	30
Yeast	5.56%	0.00%	5.56%	0.00%	11.11%	0.00%	0.00%	18
Fungal	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	3
Viral	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	2
Worm	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	1
Homogeneous	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	1

Table 3: Predicted assays by type at the 0.9 threshold, median AUROC over scaffold-based splits was used. The source of the table [97].

	CS	GE	MO	CS+GE	CS+MO	GE+MO	CS+GE+MO	Evaluated assays
Cell-based	36.54%	37.18%	44.23%	47.44%	46.15%	51.28%	50.00%	156
Biochemical	40.68%	8.47%	23.73%	32.20%	42.37%	18.64%	33.90%	59
Bacterial	40.00%	13.33%	46.67%	23.33%	56.67%	36.67%	43.33%	30
Yeast	33.33%	11.11%	11.11%	33.33%	33.33%	16.67%	16.67%	18
Fungal	66.67%	33.33%	33.33%	33.33%	66.67%	33.33%	33.33%	3
Viral	50.00%	0.00%	0.00%	50.00%	50.00%	0.00%	50.00%	2
Worm	0.00%	100.00%	100.00%	100.00%	0.00%	100.00%	0.00%	1
Homogeneous	0.00%	0.00%	100.00%	0.00%	100.00%	100.00%	100.00%	1

Table 4: Predicted assays by type at the 0.7 threshold, median AUROC over scaffold-based splits was used. The source of the table [97].

4 Conclusions

The progress in computation and high-throughput biology methods is a mutual exchange: the rise of computational power paved the way for high-throughput methods. This, in turn, engages the computational powers by producing new piles of data, which have to be analyzed. As a part of those processes, new scientific sub-fields and computational analysis methods emerged. Imaging waited its turn, strengthening its methods from the wet lab and computational sides for a little while, even though the first image analysis attempts were successful and founded a new field [9] [19].

Biological image analysis skyrocketed in the middle of the 2010s when the shift from classical image analysis to deep-learning-based image analysis started and GPU-computation has become affordable. By this time, the wet-lab protocols for imaging were mostly established, and new specific protocols [10] and techniques (i.e. super-resolution) [107] appeared. The methods for image classification, detection and segmentation were swiftly adopted by the community of computational biologists for the specific tasks [7][108].

The sub-field of cell (nucleus) segmentation has matured in the last few years, besides the new methods (including attempts to build a general cell segmentation method) and additional post-processing methods, also new large-scale datasets and annotation tools were published [7]. Currently, new methods, usually specific for a particular domain of data are developed, but the community strives for general segmentation models and 3D segmentation [7].

As a part of the renaissance of phenotypic drug discovery [46], one of the biological imaging analysis sub-fields of particular interest with wide applicability of deep-learning methods [48] [51] is image-based profiling [10]. It is expected to advance in near future from both biological and computational sides [52]. From the computational side, all eyes are on unsupervised deep-learning methods. The hope is, that those will be more capable of capturing biologically relevant features of single-cells [93], rather than their supervised and weakly-supervised counterparts.

This thesis is focused on the usage of deep learning-based methods for single-cell segmentation and phenotypic profiling. From the segmentation side, the thesis presents the review of the nucleus segmentation sub-field, an annotation tool to create cell(nucleus) segmentation datasets and an evaluation of a post-processing method for nucleus segmentation. From the phenotyping side, the thesis presents weakly-supervised learning for large-scale image-based profiling and an evaluation of the predictive power of different cellular data modalities.

1. In a review paper, descriptions of the deep learning-based segmentation methods for 2D and 3D data, descriptions of the datasets and annotation tools. Several important points regarding the current state of the field of nuclei segmentation were expressed with the hope that the community will take those into account. The decision support helper tool for segmentation method selection was developed.

2. AnnotatorJ, the plugin for the popular imaging software ImageJ/Fiji, which utilizes

pre-trained models based on U-Net to ease the annotations of nuclei images. The experiments with expert annotators showed that AnnotatorJ reduces the time needed for the annotation and improves the accuracy of the produced annotations.

3. The test-time augmentation approach was experimentally evaluated for two popular deep learning frameworks: U-Net and Mask R-CNN. According to the observed results, it is possible to obtain additional segmentation accuracy with TTA on average, though in individual cases it is not guaranteed. Besides, in cases with underfit models, the usage of TTA marginally hurts the average segmentation performance. Visual observation of the images also showed, that TTA mostly modifies the output segmentations in the objects' borders, though in rare cases, especially in the case of Mask R-CNN, as it is instance segmentation-based the segmentations of the whole objects (improving segmentation by removing false positives or adding true positives). The recommendation would be to use TTA for the analysis of uncertain regions in segmentation. Besides, the computational cost of predictions increases with the use of TTA, but it is a concern only at a very large scale or if the inference is running on a CPU.

4. CNNs trained with a weakly-supervised learning approach were benchmarked in three large-scale profiling datasets versus classical features and pre-trained CNN baselines. The main finding is that by maximizing technical and phenotypic variation, WSL improves in capturing the biologically relevant representations. Batch-correction turned out to be a crucial element in capturing phenotypic variation. During this project, the combined Cell Painting dataset was gathered and a software tool DeepProfiler for deep learning-based image profiling was developed. As a result of experiments, a trained model for feature extraction from Cell Painting data was obtained.

5. The predictive power of different data modalities was evaluated: morphology, transcriptional profiles and chemical structures for the prediction of assay readouts. The results show that those three modalities individually can predict 6-10% of assays with high accuracy. According to experiments, those modalities turned out to be complementary combined and can provide up to 21% of assays that can be predicted with high accuracy or up to 64% if lower accuracy is acceptable.

List of Figures

List of Tables

References

[1] Peter Horvath, Nathalie Aulner, Marc Bickle, Anthony M Davies, Elaine Del Nery, Daniel Ebner, Maria C Montoya, Päivi Östling, Vilja Pietiäinen, Leo S Price, Spencer L Shorte, Gerardo Turcatti, Carina von Schantz, and Neil O Carragher. Screening out irrelevant cell-based models of disease. *Nat. Rev. Drug Discov.*, 15(11):751–769, November 2016.

[2] Ehud Shapiro, Tamir Biezuner, and Sten Linnarsson. Single-cell sequencing-based technologies will revolutionize whole-organism science. *Nat. Rev. Genet.*, 14(9):618–630, September 2013.

[3] Lukas Badertscher, Thomas Wild, Christian Montellese, Leila T Alexander, Lukas Bammert, Marie Sarazova, Michael Stebler, Gabor Csucs, Thomas U Mayer, Nicola Zamboni, Ivo Zemp, Peter Horvath, and Ulrike Kutay. Genome-wide RNAi screening identifies protein modules required for 40S subunit synthesis in human cells. *Cell Rep.*, 13(12):2879–2891, December 2015.

[4] Olaf Wolkenhauer, Peter Wellstead, Kwang-Hyun Cho, Dhanya Mullassery, Caroline A Horton, Christopher D Wood, and Michael R H White. Single live-cell imaging for systems biology 9. *Essays Biochem.*, 45:121–134, September 2008.

[5] Jeffrey M Levsky, Shailesh M Shenoy, Rossanna C Pezo, and Robert H Singer. Single-cell gene expression profiling. *Science*, 297(5582):836–840, August 2002.

[6] Nikolai Slavov. Single-cell protein analysis by mass spectrometry. *Curr. Opin. Chem. Biol.*, 60:1–9, February 2021.

[7] Reka Hollandi, Nikita Moshkov, Lassi Paavolainen, Ervin Tasnadi, Filippo Piccinini, and Peter Horvath. Nucleus segmentation: towards automated solutions. *Trends Cell Biol.*, January 2022.

[8] Ben T Grys, Dara S Lo, Nil Sahin, Oren Z Kraus, Quaid Morris, Charles Boone, and Brenda J Andrews. Machine learning and computer vision approaches for phenotypic profiling. *J. Cell Biol.*, 216(1):65–71, January 2017.

[9] Zachary E. Perlman, Michael D. Slack, Yan Feng, Timothy J. Mitchison, Lani F. Wu, and Steven J. Altschuler. Multidimensional drug profiling by automated microscopy. *Science*, 306(5699):1194–1198, 2004.

[10] Mark-Anthony Bray, Shantanu Singh, Han Han, Chadwick T Davis, Blake Borgeson, Cathy Hartland, Maria Kost-Alimova, Sigrun M Gustafsdottir, Christopher C Gibson, and Anne E Carpenter. Cell painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes. January 2016.

[11] Juan C Caicedo, Allen Goodman, Kyle W Karhohs, Beth A Cimini, Jeanelle Ackerman, Marzieh Haghighi, Cherkeng Heng, Tim Becker, Minh Doan, Claire McQuin, Mohammad Rohban, Shantanu Singh, and Anne E Carpenter. Nucleus segmentation across imaging experiments: the 2018 data science bowl. *Nat. Methods*, 16(12):1247–1253, December 2019.

[12] Réka Hollandi, Ákos Diósdi, Gábor Hollandi, Nikita Moshkov, and Péter Horváth. AnnotatorJ: an ImageJ plugin to ease hand annotation of cellular compartments. *Mol. Biol. Cell*, 31(20):2179–2186, September 2020.

[13] Johannes Schindelin, Ignacio Arganda-Carreras, Erwin Frise, Verena Kaynig, Mark Longair, Tobias Pietzsch, Stephan Preibisch, Curtis Rueden, Stephan Saalfeld, Benjamin Schmid, Jean-Yves Tinevez, Daniel James White, Volker Hartenstein, Kevin Eliceiri, Pavel Tomancak, and Albert Cardona. Fiji: an open-source platform for biological-image analysis. *Nat. Methods*, 9(7):676–682, June 2012.

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for Large-Scale image recognition. September 2014.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.

[16] Mohammad H Rohban, Hamdah S Abbasi, Shantanu Singh, and Anne E Carpenter. Capturing single-cell heterogeneity via data fusion improves image-based profiling. *Nat. Commun.*, 10(1):2082, May 2019.

[17] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Philipp Eiden, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian Kelley, Miriam Mathea, Andrew Palmer, Volker Settels, Tommi Jaakkola, Klavs Jensen, and Regina Barzilay. Analyzing learned molecular representations for property prediction. *J. Chem. Inf. Model.*, 59(8):3370–3388, August 2019.

[18] Aravind Subramanian, Rajiv Narayan, Steven M Corsello, David D Peck, Ted E Natoli, Xiaodong Lu, Joshua Gould, John F Davis, Andrew A Tubelli, Jacob K Asiedu, David L Lahr, Jodi E Hirschman, Zihan Liu, Melanie Donahue, Bina Julian, Mariya Khan, David Wadden, Ian C Smith, Daniel Lam, Arthur Liberzon, Courtney Toder, Mukta Bagul, Marek Orzechowski, Oana M Enache, Federica Piccioni, Sarah A Johnson, Nicholas J Lyons, Alice H Berger, Alykhan F Shamji, Angela N Brooks, Anita Vrcic, Corey Flynn, Jacqueline Rosains, David Y Takeda, Roger Hu, Desiree Davison, Justin Lamb, Kristin Ardlie, Larson Hogstrom, Peyton Greenside, Nathanael S Gray, Paul A Clemons, Serena Silver, Xiaoyun Wu, Wen-Ning Zhao, Willis Read-Button, Xiaohua Wu, Stephen J Haggarty, Lucienne V Ronco, Jesse S Boehm, Stuart L Schreiber, John G Doench, Joshua A Bittker, David E Root, Bang Wong, and Todd R Golub.

A next generation connectivity map: L1000 platform and the first 1,000,000 profiles. *Cell*, 171(6):1437–1452.e17, November 2017.

[19] Anne E Carpenter, Thouis R Jones, Michael R Lamprecht, Colin Clarke, In Han Kang, Ola Friman, David A Guertin, Joo Han Chang, Robert A Lindquist, Jason Moffat, Polina Golland, and David M Sabatini. CellProfiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biol.*, 7(10):R100, October 2006.

[20] Nikita Moshkov, Botond Mathe, Attila Kertesz-Farkas, Reka Hollandi, and Peter Horvath. Test-time augmentation for deep learning-based cell segmentation on microscopy images. *Sci. Rep.*, 10(1):5068, March 2020.

[21] Aditya Pratapa, Michael Doron, and Juan C Caicedo. Image-based cell phenotyping with deep learning. *Curr. Opin. Chem. Biol.*, 65:9–17, December 2021.

[22] Erik Meijering. Cell segmentation: 50 years down the road [life sciences]. *IEEE Signal Process. Mag.*, 29(5):140–145, September 2012.

[23] Christoph Sommer, Christoph Straehle, Ullrich Köthe, and Fred A Hamprecht. Ilastik: Interactive learning and segmentation toolkit. In *2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 230–233, March 2011.

[24] Pascal Bamford and Brian Lovell. Unsupervised cell nucleus segmentation with active contours. *Signal Processing*, 71(2):203–213, December 1998.

[25] Jozsef Moinar, Adam Istvan Szucs, Csaba Molnar, and Peter Horvath. Active contours for selective object segmentation. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9, March 2016.

[26] Csaba Molnar, Ian H Jermyn, Zoltan Kato, Vesa Rahkama, Päivi Östling, Piia Mikkonen, Vilja Pietiäinen, and Peter Horvath. Accurate morphology preserving segmentation of overlapping cells based on active contours. *Sci. Rep.*, 6:32412, August 2016.

[27] Adrien Hallou, Hannah G Yevick, Bianca Dumitrascu, and Virginie Uhlmann. Deep learning for bioimage analysis in developmental biology. *Development*, 148(18), September 2021.

[28] Srinivas Niranj Chandrasekaran, Hugo Ceulemans, Justin D Boyd, and Anne E Carpenter. Image-based profiling for drug discovery: due for a machine-learning upgrade? *Nat. Rev. Drug Discov.*, 20(2):145–159, February 2021.

[29] Riddhiman Dhar, Alsu M Missarova, Ben Lehner, and Lucas B Carey. Single cell functional genomics reveals the importance of mitochondria in cell-to-cell phenotypic variation. *Elife*, 8, January 2019.

[30] Tomohiro Hayakawa, V B Surya Prasath, Hiroharu Kawanaka, Bruce J Aronow, and Shinji Tsuruoka. Computational nuclei segmentation methods in digital pathology: A survey. *Arch. Comput. Methods Eng.*, 28(1):1–13, January 2021.

[31] Nobuyuki Otsu. A threshold selection method from Gray-Level histograms. *IEEE Trans. Syst. Man Cybern.*, 9(1):62–66, January 1979.

[32] Stéfan van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and scikit-image contributors. scikit-image: image processing in python. *PeerJ*, 2:e453, June 2014.

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science*, Lecture notes in computer science, pages 234–241. Springer International Publishing, Cham, 2015.

[34] Carsen Stringer, Tim Wang, Michalis Michaelos, and Marius Pachitariu. Cellpose: a generalist algorithm for cellular segmentation. *Nat. Methods*, 18(1):100–106, January 2021.

[35] Uwe Schmidt, Martin Weigert, Coleman Broaddus, and Gene Myers. Cell detection with star-convex polygons. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*, Lecture notes in computer science, pages 265–273. Springer International Publishing, Cham, 2018.

[36] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask R-CNN. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, October 2017.

[37] Reka Hollandi, Abel Szkalisity, Timea Toth, Ervin Tasnadi, Csaba Molnar, Botond Mathe, Istvan Grexa, Jozsef Molnar, Arpad Balind, Mate Gorbe, Maria Kovacs, Ede Migh, Allen Goodman, Tamas Balassa, Krisztian Koos, Wenyu Wang, Juan Carlos Caicedo, Norbert Bara, Ferenc Kovacs, Lassi Paavolainen, Tivadar Danka, Andras Kriston, Anne Elizabeth Carpenter, Kevin Smith, and Peter Horvath. NucleAIzer: A parameter-free deep learning framework for nucleus segmentation using image style transfer. *Cell Syst.*, 10(5):453–458.e6, May 2020.

[38] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017.

[39] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.

[40] Andrew Y Ng. Feature selection, L1 vs. L2 regularization, and rotational invariance. In *Twenty-first international conference on Machine learning - ICML '04*, New York, New York, USA, 2004. ACM Press.

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[42] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149, June 2017.

[43] Ellen L Berg. The future of phenotypic drug discovery. *Cell Chem Biol*, 28(3):424–430, March 2021.

[44] David C Swinney and Jonathan A Lee. Recent advances in phenotypic drug discovery. *F1000Res.*, 9, August 2020.

[45] Jörg Eder, Richard Sedrani, and Christian Wiesmann. The discovery of first-in-class drugs: origins and evolution. *Nat. Rev. Drug Discov.*, 13(8):577–587, August 2014.

[46] Fabien Vincent, Arsenio Nueda, Jonathan Lee, Monica Schenone, Marco Prunotto, and Mark Mercola. Phenotypic drug discovery: recent successes, lessons learned and new directions. *Nature Reviews Drug Discovery*, may 2022.

[47] Sigrun M Gustafsdottir, Vebjorn Ljosa, Katherine L Sokolnicki, J Anthony Wilson, Deepika Walpita, Melissa M Kemp, Kathleen Petri Seiler, Hyman A Carrel, Todd R Golub, Stuart L Schreiber, Paul A Clemons, Anne E Carpenter, and Alykhan F Shamji. Multiplex cytological profiling assay to measure diverse cellular states. *PLoS One*, 8(12):e80999, December 2013.

[48] Juan C Caicedo, Sam Cooper, Florian Heigwer, Scott Warchal, Peng Qiu, Csaba Molnar, Aliaksei S Vasilevich, Joseph D Barry, Harmanjit Singh Bansal, Oren Kraus, Mathias Wawer, Lassi Paavolainen, Markus D Herrmann, Mohammad Rohban, Jane Hung, Holger Hennig, John Concannon, Ian Smith, Paul A Clemons, Shantanu Singh, Paul Rees, Peter Horvath, Roger G Linington, and Anne E Carpenter. Data-analysis strategies for image-based cell profiling. *Nat. Methods*, 14(9):849–863, August 2017.

[49] Joseph Boyd. *Deep learning for computational phenotyping in cell-based assays.* PhD thesis, Université Paris sciences et lettres, June 2020.

[50] Juan C Caicedo, Claire McQuin, Allen Goodman, Shantanu Singh, and Anne E Carpenter. Weakly supervised learning of Single-Cell feature embeddings. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2018:9309–9318, June 2018.

[51] Nick Pawlowski, Juan C Caicedo, Shantanu Singh, Anne E Carpenter, and Amos Storkey. Automating morphological profiling with generic deep convolutional networks. November 2016.

[52] Srinivas Niranj Chandrasekaran, Hugo Ceulemans, Justin D. Boyd, and Anne E. Carpenter. Image-based profiling for drug discovery: due for a machine-learning upgrade? *Nature Reviews Drug Discovery*, 20(2):145–159, dec 2020.

[53] Mingyue Zheng, Jihui Zhao, Chen Cui, Zunyun Fu, Xutong Li, Xiaohong Liu, Xiaoyu Ding, Xiaoqin Tan, Fei Li, Xiaomin Luo, Kaixian Chen, and Hualiang Jiang. Computational chemical biology and drug design: Facilitating protein structure, function, and modulation studies. *Med. Res. Rev.*, 38(3):914–950, May 2018.

[54] Douglas B Kell, Soumitra Samanta, and Neil Swainston. Deep learning and generative methods in cheminformatics and chemical biology: navigating small molecule space intelligently. *Biochem. J*, 477(23):4559–4580, December 2020.

[55] David Weininger. SMILES, a chemical language and information system. 1. introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.*, 28(1):31–36, February 1988.

[56] B Zagidullin, Z Wang, Y Guan, E Pitkänen, and J Tang. Comparative analysis of molecular fingerprints in prediction of drug combination effects. *Brief. Bioinform.*, 22(6), November 2021.

[57] H L Morgan. The generation of a unique machine description for chemical Structures-A technique developed at chemical abstracts service. *J. Chem. Doc.*, 5(2):107–113, May 1965.

[58] G W Bemis and M A Murcko. The properties of known drugs. 1. molecular frameworks. *J. Med. Chem.*, 39(15):2887–2893, July 1996.

[59] Karren Yang, Samuel Goldman, Wengong Jin, Alex Lu, Regina Barzilay, Tommi Jaakkola, and Caroline Uhler. Improved conditional flow models for molecule to image synthesis. June 2020.

[60] Jonathan M Stokes, Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M Donghia, Craig R MacNair, Shawn French, Lindsey A Carfrae, Zohar Bloom-Ackermann, Victoria M Tran, Anush Chiappino-Pepe, Ahmed H Badran, Ian W Andrews, Emma J Chory, George M Church, Eric D Brown, Tommi S Jaakkola, Regina Barzilay, and James J Collins. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702.e13, February 2020.

[61] Kai Yao, Kaizhu Huang, Jie Sun, Linzhi Jing, Dejian Huang, and Curran Jude. Scaffold-A549: A benchmark 3D fluorescence image dataset for unsupervised nuclei segmentation. *Cognit. Comput.*, November 2021.

[62] Christoffer Edlund, Timothy R Jackson, Nabeel Khalid, Nicola Bevan, Timothy Dale, Andreas Dengel, Sheraz Ahmed, Johan Trygg, and Rickard Sjögren. LIVECell-A

large-scale dataset for label-free live cell segmentation. *Nat. Methods*, 18(9):1038–1045, September 2021.

[63] Florin C Walter, Sebastian Damrich, and Fred A Hamprecht. Multistar: Instance segmentation of overlapping objects with star-convex polygons. In *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. IEEE, April 2021.

[64] Soham Mandal and Virginie Uhlmann. Splinedist: Automated cell segmentation with spline curves. In *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pages 1082–1086, April 2021.

[65] Linfeng Yang, Rajarshi P Ghosh, J Matthew Franklin, Simon Chen, Chenyu You, Raja R Narayan, Marc L Melcher, and Jan T Liphardt. NuSeT: A deep learning tool for reliably separating and analyzing crowded cells. *PLoS Comput. Biol.*, 16(9):e1008193, September 2020.

[66] Ulysse Rubens, Romain Mormont, Lassi Paavolainen, Volker Bäcker, Benjamin Pavie, Leandro A Scholz, Gino Michiels, Martin Maška, Devrim Ünay, Graeme Ball, Renaud Hoyoux, Rémy Vandaele, Ofra Golani, Stefan G Stanciu, Natasa Sladoje, Perrine Paul-Gilloteaux, Raphaël Marée, and Sébastien Tosi. BIAFLOWS: A collaborative framework to reproducibly deploy and benchmark bioimage analysis workflows. *Patterns (N Y)*, 1(3):100040, June 2020.

[67] Ruchika Verma, Neeraj Kumar, Abhijeet Patil, Nikhil Cherian Kurian, Swapnil Rane, Simon Graham, Quoc Dang Vu, Mieke Zwager, Shan E Ahmed Raza, Nasir Rajpoot, Xiyi Wu, Huai Chen, Yijie Huang, Lisheng Wang, Hyun Jung, G Thomas Brown, Yanling Liu, Shuolin Liu, Seyed Alireza Fatemi Jahromi, Ali Asghar Khani, Ehsan Montahaei, Mahdieh Soleymani Baghshah, Hamid Behroozi, Pavel Semkin, Alexandr Rassadin, Prasad Dutande, Romil Lodaya, Ujjwal Baid, Bhakti Baheti, Sanjay Talbar, Amirreza Mahbod, Rupert Ecker, Isabella Ellinger, Zhipeng Luo, Bin Dong, Zhengyu Xu, Yuehan Yao, Shuai Lv, Ming Feng, Kele Xu, Hasib Zunair, Abdessamad Ben Hamza, Steven Smiley, Tang-Kai Yin, Qi-Rui Fang, Shikhar Srivastava, Dwarikanath Mahapatra, Lubomira Trnavska, Hanyun Zhang, Priya Lakshmi Narayanan, Justin Law, Yinyin Yuan, Abhiroop Tejomay, Aditya Mitkari, Dinesh Koka, Vikas Ramachandra, Lata Kini, and Amit Sethi. MoNuSAC2020: A multi-organ nuclei segmentation and classification challenge. *IEEE Trans. Med. Imaging*, pages 1–1, 2021.

[68] Kazuhisa Matsunaga, Akira Hamada, Akane Minagawa, and Hiroshi Koga. Image classification of melanoma, nevus and seborrheic keratosis by deep neural network ensemble. March 2017.

[69] Murat Seckin Ayhan and Philipp Berens. Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks. April 2018.

[70] Guotai Wang, Wenqi Li, Michael Aertsen, Jan Deprest, Sébastien Ourselin, and Tom Vercauteren. Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks, 2019.

[71] Csilla Brasko, Kevin Smith, Csaba Molnar, Nora Farago, Lili Hegedus, Arpad Balind, Tamas Balassa, Abel Szkalisity, Farkas Sukosd, Katalin Kocsis, Balazs Balint, Lassi Paavolainen, Marton Z Enyedi, Istvan Nagy, Laszlo G Puskas, Lajos Haracska, Gabor Tamas, and Peter Horvath. Intelligent image-based in situ single-cell isolation, 2018.

[72] Peter D Caie, Rebecca E Walls, Alexandra Ingleston-Orme, Sandeep Daya, Tom Houslay, Rob Eagle, Mark E Roberts, and Neil O Carragher. High-content phenotypic profiling of drug response signatures across distinct cancer cells. *Mol. Cancer Ther.*, 9(6):1913–1926, June 2010.

[73] Luis Pedro Coelho, Aabid Shariff, and Robert F Murphy. Nuclear segmentation in microscope cell images: A hand-segmented dataset and comparison of algorithms, 2009.

[74] Kevin Smith, Yunpeng Li, Filippo Piccinini, Gabor Csucs, Csaba Balazs, Alessandro Bevilacqua, and Peter Horvath. CIDRE: an illumination-correction method for optical microscopy, 2015.

[75] Peter Naylor, Marick Lae, Fabien Reyal, and Thomas Walter. Segmentation of nuclei in histopathology images by deep regression of the distance map, 2019.

[76] Neeraj Kumar, Ruchika Verma, Sanuj Sharma, Surabhi Bhargava, Abhishek Vahadane, and Amit Sethi. A dataset and a technique for generalized nuclear segmentation for computational pathology. *IEEE Trans. Med. Imaging*, 36(7):1550–1560, July 2017.

[77] Juan C. Caicedo, Jonathan Roth, Allen Goodman, Tim Becker, Kyle W. Karhohs, Matthieu Broisin, Csaba Molnar, Claire McQuin, Shantanu Singh, Fabian J. Theis, and Anne E. Carpenter. Evaluation of deep learning strategies for nucleus segmentation in fluorescence images. *Cytometry Part A*, 95(9):952–965, 2019.

[78] zhixuhao. zhixuhao/unet. `https://github.com/zhixuhao/unet`. Accessed: 2019-10-7.

[79] Tensorflow Developers. TensorFlow, 2021.

[80] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. December 2014.

[81] matterport. matterport/Mask_RCNN. `https://github.com/matterport/Mask_RCNN`. Accessed: 2019-10-7.

[82] Nikita Moshkov, Michael Bornholdt, Santiago Benoit, Claire McQuin, Matthew Smith, Allen Goodman, Rebecca Senft, Yu Han, Mehrtash Babadi, Peter Horvath, Beth A. Cimini, Anne E. Carpenter, Shantanu Singh, and Juan C Caicedo. Learning representations for image-based profiling of perturbations. *bioRxiv*, 2022.

[83] Donald B Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *J. Educ. Psychol.*, 66(5):688–701, October 1974.

[84] Fredrik Johansson, Uri Shalit, and David Sontag. Learning representations for counterfactual inference. In Maria Florina Balcan and Kilian Q Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 3020–3029, New York, New York, USA, 2016. PMLR.

[85] Mohammad Hossein Rohban, Shantanu Singh, Xiaoyun Wu, Julia B Berthet, Mark-Anthony Bray, Yashaswi Shrestha, Xaralabos Varelas, Jesse S Boehm, and Anne E Carpenter. Systematic morphological profiling of human gene and allele function via cell painting. *Elife*, 6, March 2017.

[86] Mark-Anthony Bray, Sigrun M Gustafsdottir, Mohammad H Rohban, Shantanu Singh, Vebjorn Ljosa, Katherine L Sokolnicki, Joshua A Bittker, Nicole E Bodycombe, Vlado Dancík, Thomas P Hasaka, Cindy S Hon, Melissa M Kemp, Kejie Li, Deepika Walpita, Mathias J Wawer, Todd R Golub, Stuart L Schreiber, Paul A Clemons, Alykhan F Shamji, and Anne E Carpenter. A dataset of images and morphological profiles of 30 000 small-molecule treatments using the cell painting assay. *Gigascience*, 6(12):1–5, December 2017.

[87] J C Caicedo, J Arevalo, F Piccioni, and others. Cell painting predicts impact of lung cancer variants. *Mol. Biol. Cell*, 2022.

[88] Gregory P Way, Ted Natoli, Adeniyi Adeboye, Lev Litichevskiy, Andrew Yang, Xiaodong Lu, Juan C Caicedo, Beth A Cimini, Kyle Karhohs, David J Logan, Mohammad Rohban, Maria Kost-Alimova, Kate Hartland, Michael Bornholdt, Niranj Chandrasekaran, Marzieh Haghighi, Shantanu Singh, Aravind Subramanian, and Anne E Carpenter. Morphology and gene expression profiling provide complementary information for mapping cell state. October 2021.

[89] Mingxing Tan and Quoc V Le. EfficientNet: Rethinking model scaling for convolutional neural networks. May 2019.

[90] Stanley Bryan Z Hua, Alex X Lu, and Alan M Moses. CytoImageNet: A large-scale pretraining dataset for bioimage transfer learning. November 2021.

[91] Vebjorn Ljosa, Peter D Caie, Rob Ter Horst, Katherine L Sokolnicki, Emma L Jenkins, Sandeep Daya, Mark E Roberts, Thouis R Jones, Shantanu Singh, Auguste Genovesio,

Paul A Clemons, Neil O Carragher, and Anne E Carpenter. Comparison of methods for image-based profiling of cellular morphological responses to small-molecule treatment. *J. Biomol. Screen.*, 18(10):1321–1329, December 2013.

[92] D Michael Ando, Cory Y McLean, and Marc Berndl. Improving phenotypic measurements in High-Content imaging screens. July 2017.

[93] Alexis Perakis, Ali Gorji, Samriddhi Jain, Krishna Chaitanya, Simone Rizza, and Ender Konukoglu. Contrastive learning of Single-Cell phenotypic representations for treatment classification. In *Machine Learning in Medical Imaging*, pages 565–575. Springer International Publishing, 2021.

[94] Agnan Kessy, Alex Lewin, and Korbinian Strimmer. Optimal whitening and decorrelation. *Am. Stat.*, 72(4):309–314, October 2018.

[95] Christopher D Manning. *Introduction to information retrieval.* Syngress Publishing,, 2008.

[96] Etienne Becht, Leland McInnes, John Healy, Charles-Antoine Dutertre, Immanuel W H Kwok, Lai Guan Ng, Florent Ginhoux, and Evan W Newell. Dimensionality reduction for visualizing single-cell data using UMAP. *Nat. Biotechnol.*, December 2018.

[97] Nikita Moshkov, Tim Becker, Kevin Yang, Peter Horvath, Vlado C Dancik, Bridget K Wagner, Paul C Clemons, Shantanu Singh, Anne E Carpenter, and Juan C Caicedo. Predicting compound activity from phenotypic profiles and chemical structures. December 2020.

[98] M Hofmarcher, E Rumetshofer, D A Clevert, and others. Accurate prediction of biological assays with high-throughput microscopy images and convolutional networks. *Journal of chemical*, 2019.

[99] Gregory P Way, Maria Kost-Alimova, Tsukasa Shibue, William F Harrington, Stanley Gill, Federica Piccioni, Tim Becker, William C Hahn, Anne E Carpenter, Francisca Vazquez, and Shantanu Singh. Predicting cell health phenotypes using image-based morphology profiling. July 2020.

[100] Jonathan B Baell and Georgina A Holloway. New substructure filters for removal of pan assay interference compounds (PAINS) from screening libraries and for their exclusion in bioassays. *J. Med. Chem.*, 53(7):2719–2740, April 2010.

[101] Mathias J Wawer, David E Jaramillo, Vlado Dančík, Daniel M Fass, Stephen J Haggarty, Alykhan F Shamji, Bridget K Wagner, Stuart L Schreiber, and Paul A Clemons. Automated Structure-Activity relationship mining: Connecting chemical structure to biological profiles. *J. Biomol. Screen.*, 19(5):738–748, June 2014.

[102] Mathias J Wawer, Kejie Li, Sigrun M Gustafsdottir, Vebjorn Ljosa, Nicole E Bodycombe, Melissa A Marton, Katherine L Sokolnicki, Mark-Anthony Bray, Melissa M Kemp, Ellen Winchester, Bradley Taylor, George B Grant, C Suk-Yee Hon, Jeremy R Duvall, J Anthony Wilson, Joshua A Bittker, Vlado Dančík, Rajiv Narayan, Aravind Subramanian, Wendy Winckler, Todd R Golub, Anne E Carpenter, Alykhan F Shamji, Stuart L Schreiber, and Paul A Clemons. Toward performance-diverse small-molecule libraries for cell-based phenotypic screening using multiplexed high-dimensional profiling. *Proceedings of the National Academy of Sciences*, 111(30):10911–10916, July 2014.

[103] Sebastian G Rohrer and Knut Baumann. Maximum unbiased validation (MUV) data sets for virtual screening based on PubChem bioactivity data. *J. Chem. Inf. Model.*, 49(2):169–184, February 2009.

[104] jingw. GitHub - jingw2/size_constrained_clustering: Implementation of size constrained clustering algorithm. `https://github.com/jingw2/size_constrained_clustering`. Accessed: 2022-4-3.

[105] Jaak Simm, Günter Klambauer, Adam Arany, Marvin Steijaert, Jörg Kurt Wegner, Emmanuel Gustin, Vladimir Chupakhin, Yolanda T Chong, Jorge Vialard, Peter Buijnsters, Ingrid Velter, Alexander Vapirev, Shantanu Singh, Anne E Carpenter, Roel Wuyts, Sepp Hochreiter, Yves Moreau, and Hugo Ceulemans. Repurposing High-Throughput image assays enables biological activity prediction for drug discovery. *Cell Chem Biol*, 25(5):611–618.e3, May 2018.

[106] Maris Lapins and Ola Spjuth. Evaluation of gene expression and phenotypic profiling data as quantitative descriptors for predicting drug targets and mechanisms of action. July 2019.

[107] Malte Renz. Fluorescence microscopy—a historical and technical perspective. *Cytometry Part A*, 83(9):767–779, 2013.

[108] Erick Moen, Dylan Bannon, Takamasa Kudo, William Graf, Markus Covert, and David Van Valen. Deep learning for cellular image analysis. *Nature Methods*, 16(12):1233–1246, may 2019.

A Article ‘Nucleus segmentation: towards automated solutions‘

This article [7] is published in ”Trends of Cell Biology”, but unfortunately it is not Open Access. So instead of the journal PDF, the final submitted version is attached. The journal version of the artical can be accessed on ”Trends of Cell Biology” website.

TITLE:

Nucleus segmentation: towards automated solutions

AUTHORS:

Reka Hollandi[1,#], Nikita Moshkov[1,2,3,#], Lassi Paavolainen[4], Ervin Tasnadi[1,5], Filippo Piccinini[6], Peter Horvath[1,4,7,*]

AFFILIATIONS:

[1] Synthetic and Systems Biology Unit, Biological Research Centre (BRC), H-6726 Szeged, Hungary.

[2] Doctoral School of Interdisciplinary Medicine, University of Szeged, Szeged, Hungary.

[3] Laboratory on AI for Computational Biology, Faculty of Computer Science, HSE University, Moscow, Russia.

[4] Institute for Molecular Medicine Finland (FIMM), HiLIFE, University of Helsinki, FI-00014 Helsinki, Finland.

[5] Doctoral School of Computer Science, University of Szeged, Szeged, Hungary.

[6] IRCCS Istituto Romagnolo per lo Studio dei Tumori (IRST) ‘‘Dino Amadori”, I-47014 Meldola (FC), Italy.

[7] Single-Cell Technologies Ltd., H-6726 Szeged, Hungary.

[*] Correspondence: horvath.peter@brc.hu (Peter Horvath).

[#] These authors contributed equally.

KEYWORDS:

Nucleus Segmentation

Image Processing

Deep Learning

Microscopy

Oncology

Single-cell Analysis

GLOSSARY:

2D: **Two Dimensional**: the term typically used to indicate the standard format in which images are acquired by a standard camera

3D: **Three Dimensional**: the term typically used to indicate a z-stack of 2D images referring to different optical sections

API: Application Programming Interface: a set of functions and procedures allowing the development of applications that access the features or data of an operating system, application, or other service

BBBC: **Broad Bioimage Benchmark Collection**: an open microscopy image collection for scientific purposes

CNN: **Convolutional Neural Network**: a class of deep neural networks including convolutional layers based on blocks responsible for appropriate image feature retrieval (via convolutions) and scaling (with pooling blocks)

DAPI: a widely used fluorescent stain that binds to adenine–thymine-rich regions of the DNA, thus labels the nucleus

DIC: Differential Interference Contrast: a microscopy technique that introduces contrast to images of specimen with little or no contrast upon brightfield microscopy

DNN: Deep Neural Network is an artificial neural network machine learning architecture that includes several hidden layers, and can be trained to solve more complex tasks on more complex data compared to shallow neural networks

DSB2018: **Data Science Bowl 2018**: the data science competition held in 2018 with a task to segment nuclei in microscopy images. The official, open dataset of the competition is also referred to as such, and is often used to benchmark nucleus segmentation methods

GPL: **General Public License**: a series of widely used open-source licenses that guarantee end users the freedom to run, study, share, and modify the software

GPU: **Graphics Processing Unit**: a specialized electronic circuit designed to rapidly manipulate memory to accelerate computations related primarily to graphics

H&E: **Hematoxylin and Eosin**: a combination of two histological stains: hematoxylin and eosin. Hematoxylin stains cell nuclei to purplish blue, and eosin stains the extracellular matrix and cytoplasm to pink

IF: Immunofluorescence: a staining which utilizes fluorescent-labelled antibodies to detect specific target antigens

ISBI: **International Symposium on Biomedical Imaging**: a scientific conference series dedicated to mathematical, algorithmic, and computational aspects of biological and biomedical imaging

IT: **Iterative Thresholding**: an algorithm used to define the background and foreground in an image

LoS: **line-of-sight**: the straight line between the object and the target

mAP: **mean Average Precision**: a popular metric related to measuring the accuracy of object detectors

MITK: **Medical Imaging Interaction Toolkit**: a software suite designed for medical image analysis

NEUBIAS: Network of European BioImage Analysts, a network of experts in life sciences for image data analysis.

PC: Phase Contrast: an optical microscopy technique that converts phase shifts in light passing through a transparent specimen to brightness changes in the image

RPN: **Region Proposal Network**: a fully convolutional network that simultaneously predicts object bounds and objectness scores at each position

siRNA: **short-interfering RNA**: a class of double-stranded, non-coding RNA molecules, similar to miRNA, operating within the RNA interference (RNAi) pathway

SNR: **Signal-to-Noise Ratio**: a measure used to compare the level of a desired signal relative to the level of background noise

TCGA: The Cancer Genome Atlas: a huge cancer genomic program which covers many cancer types with a patient-based, open dataset including genomic, proteomic, imaging etc. data

TIFF: **Tagged Image File Format**: one of the most common image file formats

TTA: Test-time augmentation: the aggregation of predictions across transformed versions of a test input

WSI: **Whole Slide Image**: scanned image of an entire histopathology tissue section, usually of gigapixel size, resulting in file size of gigabytes, which is difficult to handle by an image processing software

ABSTRACT

Single nucleus segmentation is a frequent challenge of microscopy image processing, since it is the first step of many quantitative data analysis pipelines. The quality of tracking single cells, extracting features or classifying cellular phenotypes strongly depends on segmentation accuracy. Worldwide competitions have been held, aiming to improve segmentation, and recent years have definitely brought significant improvements: large annotated datasets are now freely available, several 2D segmentation strategies have been extended to 3D, and deep learning approaches have increased accuracy. However, even today, no generally accepted solution and benchmarking platform exist. We review the most recent single-cell segmentation tools, and provide an interactive method browser to select the most appropriate solution.

TOWARDS ROBUST AND AUTOMATED METHODS FOR NUCLEUS SEGMENTATION

The history [1] of detecting and segmenting single cells goes along with the first digitized microscopy images. Many research fields utilizing microscopy, such as developmental biology [2], drug discovery [3], functional genomics [4] and pathology [5] are dependent on accurate cell and nucleus segmentation as a vital part of image analysis workflows. Since image analysis has moved from a methodological research area towards data science as a result of the recent machine learning revolution, annotated datasets have become essential regarding the performance of nuclear segmentation methods. Especially, modality-independent, generalizable, and robust machine learning-based nucleus segmentation models need heterogeneous and large collections of expert-annotated images [6,7].

The level of difficulty of single-cell detection in an image, let alone precise outlining, widely varies (see **Figure 1**). In most simple cases nuclei have high contrast and are separated by proper experimental conditions (referred to as *easy* cases), hence their segmentation is not difficult (e.g. large **siRNA** - see **Glossary)** [8]. In other cases segmentation is highly *challenging*, for instance in **3D**, label-free or thick tissue sections where cells touch, overlap or have non-conventional morphology, intensity, or patterns. International competitions [6][9] have promoted the potential to overcome these issues, yet a genuinely general solution is still awaited. However, due to major advancements in this field in recent years, our community has reached an unprecedented improvement in detecting single nuclei [10]. Easy cases of segmentation, especially in 2D are not problematic anymore [11,12], while accuracy has also improved in challenging cases [6]. In addition, 3D data analysis methods have progressed with extended 2D segmentation solutions [13] or with native 3D ones [14][15]. The community has accumulated large amounts of annotated data (either by experts [6] or crowdsourcing [16]) for training machine learning segmentation models, and to evaluate the methods in public benchmarking platforms [6][17]. This review describes the specific techniques biologists can exploit for single-cell analysis. However, we emphasise that no standardized approach has been developed to date to properly compare different solutions before deciding which tool to use for a specific application.

First, the variety and extent of datasets currently available to test and train methods are presented. Next, a selection of annotation tools available for creation of training datasets for machine learning methods is introduced. Then the issues related to pre- and post-processing of images to reduce challenges inherent to complex data are briefly discussed (see **Suppl. Mat. 1** for details on different techniques), followed by insights into 2D nuclear segmentation methods. Classical approaches that provide task-specific and general solutions for a wide variety of acquisition techniques are presented. However, most recent methods usually rely on **DNNs**, and since the target objective is related to image processing, **CNNs** are most commonly applied to segment nuclei (see in **Glossary**). As processing 3D data is one of the major challenges in single nucleus segmentation, a set of promising and successful methods appropriate to solve specific 3D segmentation tasks is discussed.

Figure 2 supports a better understanding of the definitions of detection and segmentation tasks. When identifying single cells (objects) in microscopy images automatically, i.e. using computer algorithms, the results may be either (*1*) *detections* corresponding to the localization of the objects or (*2*) *segmentations* which separate independent image regions. The former is typically represented as bounding boxes, whereas the latter may be realized by either assigning a binary label to each pixel while dividing the image into not necessarily connected regions (*semantic segmentation*) or by separating individual objects (*instance segmentation*). One may choose from a plethora of methods and software tools to perform nuclear segmentation (see section "*Segmentation methods and toolkits*"). Major challenges are discussed in the **Outstanding Questions Box**. As many of the segmentation methods considered in this review utilize *deep learning* approaches, this subset of machine learning is also introduced. Deep learning involves the training of **DNNs** for complex yet arbitrary tasks, such as detection, segmentation and classification (not strictly in our domain of cellular image analysis, but also in natural image-, video- or audio-processing). Deep learning based approaches are proven to perform excellently on the trained domain, with the potential of extension to unseen domains [18]. Their translation is limited by the lack of publicly available datasets related to less common modalities (see section "*Annotated nucleus datasets*").

A portal[I] was developed to offer a graphical aid to select the most appropriate method for non-image analysis experts (see **Figure 2**). This portal has several advantages: (*1*) the imaging community can select the methods applicable/appropriate for their images of interest, and (*2*) developers can submit the description and best practices for their methods. Currently, the most common optical microscopy categories are considered. Notably, the web portal is declared to be maintained by the authors, yet the community is encouraged to actively contribute and eventually propose extensions to it. For benchmarking the segmentation methods, users can exploit BIAFLOWS[II] [17], a freely available web-based platform developed by **NEUBIAS**. The assessment of new proposals has been commonly performed with limited datasets and arbitrary metrics (see **Suppl. Mat. 2**); in contrast, NEUBIAS is a step forward to prevent such a biased evaluation. However, an unbiased quantitative method comparison is still impossible due to the lack of a comprehensive annotated dataset for training and testing the methods using a globally accepted benchmark platform and unified metrics (see section "*Segmentation methods and toolkits*").

ANNOTATED NUCLEUS DATASETS

Annotated datasets are used in computer science to validate the accuracy of developed algorithms. In addition, nowadays annotated datasets are also used for training machine learning models for various tasks. One of the key factors influencing the performance of segmentation models is the composition of annotated data. Ideally, a trainable model yields optimal results on a test set sampled from the same domain as training data are collected from, hence domain-specific annotated datasets serve as a valuable asset, especially when they are expert-curated. Highly specific domain datasets are usually complemented with proper metadata [19][20], such as the experimental setup, sample preparation or microscope device, and are expert-curated when it comes to annotations. However, they typically cover a narrow diversity, and are small in size. Open datasets may contain a varying number of images (see **Table 1**). An important aspect for the user to consider when either training a new model or evaluating segmentation performance on publicly available datasets is that the corresponding annotations occasionally contain such segmentations yielded by automatic methods [19] that might not be refined by an expert, thus the results might be biased. Annotated nucleus datasets displayed in **Table 1** show diversity in size (not only regarding the number of images, but also that of the objects too) and content, focusing on those widely used as benchmarks or training data. Annotations may be realized as objects (instance aware) or binary masks (semantic), are primarily 2D, and the two most common imaging modalities cover fluorescence stained cell cultures and **H&E** stained tissue sections.

Table 1. Open datasets of annotated nucleus for single-cell analysis purposes

Name	Individual objects (O) or binary masks (BM)	2D/3D	Microscopy	Staining	Sample	# images	# objects	Ref.
BBBC032	O	3D	confocal	fluo	mouse embryo blastocyst cells	1 (172*)	1220	[21]
BBBC033	O	3D	confocal	fluo	mouse trophoblast stem cells	1 (32*)	585	[21]
BBBC034	O	3D	brightfield/fluorescent	x/3 fluo**	hiPSC 3D	1 (52*)	790	[19]
Scaffold-A549 Dataset	O	3D	fluorescent	Hoechst +DiL	lung cancer tissue	21	800 (+10000 w/o labels)	[22]
BBBC039	O	2D	fluorescent	Hoechst	U2OS cells	200	23.615	[10]
CoNSeP	O	2D	brightfield	H&E	colon tissue	41	24.319	[23]
CryoNuSeg (TCGA[IV])	O	2D	brightfield	H&E	various tissues	30	7.596	[24]
DSB2018	O	2D	various	various	various tissues and cells	841	37.530	[6]
Janowczyk et al.[V]	BM	2D	brightfield	H&E	breast tissue	141	~12.000	[25]
LIVECell	O	2D	phase-contrast	label-free	cell cultures	5 239	1.686.352	[26]

Lizard	O	2D	brightfield	H&E	colon tissue	291	495.179	[27]
MoNuSeg2018	O	2D	brightfield	H&E	various tissues	44	28.846	[28]
NuCLS	O***	2D	brightfield	H&E	breast tissue	N/A	222.396	[16]
NucMM	O	3D	electron microscopy/ micro-CT	label-free	brain tissue	2	~170.000 + ~7.000	[29]
PanNuke	O	2D	brightfield	H&E	various tissues	481	205.343	[30]
S-BSST265	O	2D	fluorescent/ confocal	IF/ DAPI	various tissues and cells	79	7.813	[20]
TCGA[IV] images processed by Irshad et at.	N/A	2D	brightfield	H&E	kidney clear cell renal carcinoma tissue	63	N/A	[31]
TCGA[IV] images processed by Kumar et at.	O	2D	brightfield	H&E	various tissues	30	21.623	[28]
TNBC	O	2D	brightfield	H&E	breast tissue	50	4.022	[32]
Wienert et al.	O	2D	brightfield	H&E	various tissues	36	7.931	[33]
TissueNet Version v1.0	O	2D	fluorescent	various stainings	various tissues	6990	~1.200.000	[7]

* the number of slices is presented; 1 image is available

** CellMask Deep Red plasma membrane, EGFP beta-actin, Hoechst DNA

*** object contours or bounding boxes with class label

International challenges, such as the annual **ISBI** or competitions hosted by e.g. Kaggle with industry partners, inspire those in the field of research and development to propose new technologies and methods or combine existing ones for a new purpose. One of the most successful and widely used segmentation methods, *U-Net [34]* (see in **Supplementary Material 4**) arose from the 2015 ISBI Cell Tracking Challenge [35], and has been the basis for several novel CNN architectures ever since (see section "*Segmentation Methods and Toolkits***"**). Similar competitions contribute to the development of this field with invaluable collections of microscopy images, on which developers may benchmark their novel approaches according to standard evaluation metrics (typically **mAP**) in a fairly comparable way. In recent years the **DSB2018** [6] dataset has been applied as such, since its image set comprises various types of microscopy modalities, magnifications, labels, sources etc. This might also provide insight into the expected model performance. Generally, datasets originating from challenges are carefully validated by field-expert annotators [6][15] (usually biologists and pathologists), promoting their further applicability to train new models. Notably, annotations of the training set are usually released instantly, while test set annotations may remain private even after the challenge is concluded [35]. Dataset size strongly depends on the task, e.g. a competition in 2D instance segmentation (like DSB2018) generally has a larger number of annotated images than a tracking [35] or a 3D segmentation task [19][21].

Conclusively, a key contribution of the bioimage analysis community to this field is the release of open datasets of annotated images, in as many varying imaging modalities as possible. Data sharing is highly encouraged, especially in case of intrinsically challenging microscopy types, such as label-free imaging (notably, LIVECell [26] is a promising step in this direction) or generally in 3D. Provided in an open way, these annotated datasets could inspire method developers to increase their focus on less frequent modalities, and release pre-trained models for those as well. Also, they enable users to benchmark (evaluate the performance of) available methods on this data. Additionally, experiment-specific unlabelled image sets (e.g. TCGA[IV]) may also promote progress in case an annotated subset is shared later independently [24,28][31]. Finally, as annotated datasets require an appropriate software tool that the experts (or generally, annotators) can use to create the labels, various annotation software solutions are collected in the following section.

TOOLS FOR ANNOTATION

Countless software tools are available to create annotations for single-cell segmentation training or validation, with a widely varying spectrum of functionality. These tools are designed either for specialists, such as biologists and pathologists, or for method developers. Options for annotation typically include freehand drawing, point, ellipse or polygon labelling, all of which may be exported to formats suitable for different applications. The finer the representation (annotation) of the object is, the more information it provides for a model when used as training data. While object location marked simply by a centre point or bounding box coordinates is sufficient for detection or even classification training, contours (boundaries) labelled either semantically (binary) or in an instance-aware way are usually used to train segmentation. Equivalently, the same types of annotated data may be utilized to assess the accuracy of different methods.

Even though labelling several images tends to be time-consuming for a single expert, even students [16] can learn how to create accurate annotations when curated by experts, yielding large annotated datasets via joint and shared efforts. Semi-automatic annotation achieved by initial segmentation methods offers a convenient solution to speed up the annotation process for experts, and is often preferred by the community. Such annotation methods also help to increase [36][37] the agreement between experts, which is a common problem source in annotation. Alternatively, a consensus of multiple annotators may be used [31][6] at the object- or pixel level; crowdsourced annotations [16] are easier to combine this way. Commercial solutions and free-to-use software, including but not limited to those applied in cell biology, are described in detail in **Suppl. Mat. 3** and **Suppl. Table 2**.

Plugins or extensions to existing open-source software, such as *ImageJ/Fiji [38,39]* or *MITK [40]* are popular choices preferred by bioimage analysts already experienced with the given software. The *Fiji* plugins *Trainable Weka Segmentation* [41] and *LabKit*[III] use machine learning to train pixel classification similarly to *ilastik* [42] (see **Suppl. Mat. 3-4**), while *AnnotatorJ* [36] applies a *U-Net* to assist contour annotation. Assistance in the *MITK* plugin *3D-Cell-Annotator* [43] exploits active surfaces with shape descriptors in 3D, while *NuClick* [18] uses its own CNN for histopathology images.

Larger image analysis projects not primarily intended for annotation, but for a rather more comprehensive evaluation of the sample images (*Cytomine* [44][45], *ilastik [42]*, *DeepCell* [46], *QuPath* [47]), including e.g. the segmentation or classification of cells, may also provide convenient solutions for annotation. Still, each has its target application: e.g. *QuPath* is a desktop tool suitable for **WSI** analysis, while *Cytomine* processes WSIs online in a collaborative way, and DeepCell improves its segmentation DNN with annotation collaboration.

Standalone software packages (*Diffgram*, *LabelImg, Segmentor* [37]) offer a lightweight, specific solution for annotation: *Segmentor* [37]is intended for 3D annotation, *Make Sense* and *Diffgram* have additional online interfaces, and the latter also supports deep learning. Online tools (*VGG Image Annotator*, *Kaibu*, *supervise.ly, Piximi annotator*) require no installation and have no specific hardware requirements. However, it is worth noting that online service-based platforms (*Lionbridge.AI* or *Hive*) require that raw data are sent out of the laboratory, which might be undesirable in case of sensitive (e.g. patient-related) images.

Nonetheless, genuinely general-purpose image editing applications, such as *GIMP* (**GPL** licence, free) or *Photoshop* (Adobe, commercial) may also be used to create annotations at the expense of more cumbersome export, e.g. in the case of instance annotation labels.

Conclusively, several options are available, depending on the specific requirements of a project or experiment. Tools that provide multiple implementations (e.g. both local and online) might be ideal for more users.

SEGMENTATION METHODS AND TOOLKITS

Single nucleus segmentation methods may work with raw images, but in more challenging cases (e.g. **Figure 1. j-v**) the quality of the analysis (and specifically that of single nucleus segmentation) benefits from additional pre- and post-processing steps (e.g. illumination correction [48][49] or denoising [50] prior to the analysis, mask refinement or test time augmentation (**TTA**) [51] applied as post-processing). Application of these methods depends on the task and the desired quality of the result; some of the most commonly used processing steps are described in **Suppl. Mat. 1**.

Nucleus segmentation is traditionally performed using a data-specific workflow that contains various filtering and thresholding methods, followed by morphological operations and processing steps (*ImageJ/Fiji* [38,39], *QuPath* [47], *CellProfiler* [52]). Segmentation using pixel classification, based on classical machine learning methods has been used for challenging data for a decade, with early versions of tools including e.g. *DeepMIB [53]* and *ilastik* [42]. The fundamental difference between classical image processing-based nucleus segmentation and that with classical machine learning is the input required from the user: in the former case, manual parameter setting and fine-tuning is expected in different processing modules in the pipeline, which is still capable of yielding very high accuracy at the expense of time-consuming re-parameterization for each new experiment. The latter enables users to rely on automated feature extraction and learning by still providing examples manually, which most likely also need to be repeated in experiments. Notably, appropriate pre-processing of input images (e.g. intensity scaling) can help to unify the range of optimal parameters in both cases. The nuclear segmentation task has moved towards robust and automated approaches with *U-Net [34]* (see in **Suppl. Mat. 4**), which was a breakthrough for deep learning-based nucleus segmentation (and in the field of deep learning-based segmentation in general). In contrast to image processing and classical machine learning, deep learning-based methods require fewer input parameters from the user, and are generally more straightforward to apply between experiments than in the case of classical approaches. Nonetheless, pre-processing also increases the accuracy of CNNs in most cases. *U-Net* still serves as a baseline for semantic segmentation tasks, and is (*1*) used as part of recent general nucleus/cell segmentation pipelines, such as *Cellpose* [12] and *StarDist* [54], and (*2*) utilized or further developed in *nnU-Ne*t [55] and *UNet*++ [56]. Even though *U-Net* is a semantic segmentation framework, it can be extended to instance segmentation with post-processing. One typical solution is to classify pixels into three classes where one class represents nuclear edges, and as such, it can aid instance segmentation [10]. Computationally *U-Net* is relatively simple, thus it is possible to train a basic *U-Net* on workstations or even laptops with a **GPU**.

Another breakthrough in deep learning-based instance segmentation was *Mask R-CNN* [57]. This network was designed for the segmentation of natural images, however, it has been adapted for nucleus segmentation in methods such as *nucleAIzer [11]*. *Mask R-CNN* is built over a *CNN* feature extraction backbone and **RPN** *[58]* to suggest possible object regions. These proposals are classified and used for binary mask prediction. *Mask R-CNN* outputs a list of masks allowing overlaps, whereas the output of *U-Net* is an image with no overlaps. However, two recent extensions to *U-Net*-based *StarDist, MultiStar [59]* and *SplineDist* [60], enable segmentation of overlapping objects. *NuSeT* [61] combines RPN, *U-Net* and watershed post-processing to optimize segmentation of crowded cells. *Mask R-CNN* requires more computational resources than *U-Net*, still it can be trained on a modern workstation or laptop.

Even though many segmentation methods are not deep learning-based (*MINS [62,63]*, *XPIWIT [64]* etc.), the field has recently tended to shift towards approaches based on deep learning (e.g. *ilastik [42]* now offers **DNNs**). This includes bundles of specific deep learning methods for segmentation and pre-processing which could be used on Google Colab (*ZeroCostDL4Mic* [65], *Segmentation of stochastic optical reconstruction microscopy (STORM) images* [66]), or other client-server architecture (*ImJoy* [67], *DeepCell Kiosk [46][68]*, *HistomicsML2* [69]) with provided separate pre-trained models (*CDeep3M* [70], *nucleAIzer* [11], *Cellpose* [12]). *ImageJ* users can also utilize deep-learning based segmentation with plugins and pre-trained models (*DeepImageJ* [71]). The majority of the methods discussed here are deep learning-based (see **Table 2**), which require hardware resources due to the parallelizable and heavy computational costs of DNNs, hence **GPU** acceleration is advised, especially for training. Cloud-based solutions often meet this requirement.

Several methods mentioned above could be used for 3D datasets (see **Table 2**). Segmentation of 3D nuclear images with deep learning is not straightforward. The major limitation is that the annotated data in the field are less abundant compared to the planar case. There are several deep learning-based methods developed by the medical image analysis community facing a similar challenge. However, in the case of medical images, usually only one or a few objects need to be segmented. This task is different from and less difficult than nucleus segmentation, where hundreds of instances should be segmented even when they touch. For example, segmenting a medical image by combining the segmentations of 2D images may provide acceptable accuracy. In contrast, nucleus segmentation is an instance segmentation task where this approach alone is less likely to work in crowded parts of the image, but the connected components of the stacked 2D segmentations can be used as a seed image for the watershed transform to compute the final 3D instance segmentation [72]. Besides, 3D segmentation is more demanding in terms of computational resources (especially GPU memory and file sizes) when a dense 2D method is extended directly to process 3D images. Introduction of a further dimension may lead to substantially growing complexity (for example in case of differential geometry-based approaches) and more complex spatial dependencies in case of CNNs, however, this phenomenon termed ‘the curse of dimensionality’ is especially problematic, thus more training data and more computational resources are required. Still, several tools are specifically developed for the 3D segmentation task [73], and some deep learning based methods developed for 2D segmentation are also extended to 3D. The *IT3DImageJSuite* is an *ImageJ* (*Fiji*) [38,39] plugin that involves several algorithms (including iterative thresholding and watershed). *LoS* [74] approximates the convex decomposition of the objects with spectral clustering.

OpenSegSPIM [75] is a *MATLAB* application which performs instance segmentation by applying a pipeline of filters in a semi-automatic manner*.* *RACE* [76] and Ruszczycki *et al*. [15] first compute the 2D segmentation on the z-slices, and then combine them to 3D objects. Similarly to *BioImageXD* [77], *Fiji* [38,39] and *Icy* [78], *Vaa3D* [79] uses a pipeline consisting of Gaussian filtering, adaptive thresholding, distance transformation and 3D watershed [80], while the MITK plugin *3D-Cell-Annotator* [43] uses active contours for semi-automatic 3D segmentation. In contrast, most recent methods apply deep learning techniques to segment nuclei. These include *QCANet [81]*, developed to analyze mouse embryos in 3D, *3DeeCellTracker* [82]*,* intended for tracking after the segmentation of nucleus instances, and the algorithm proposed in Lapierre-Landry *et al*. [83] which performs watershed segmentation on the probability map, and supervoxel clustering to achieve the final instance segmentation.

Self-supervised and unsupervised learning approaches decrease or even eliminate the need of annotated training data. A few of such methods for nuclear segmentation have appeared recently [84,85][86]. These methods show competitive results, though their accuracy does not exceed that of the supervised state-of-the-art methods. Self-supervised segmentation for histopathology images [85] uses *ResUnet-101* and requires a minimum of annotated data for fine-tuning. Another approach [84] uses an attention mechanism, and does not require annotated data. *AD-GAN* [86]uses a sophisticated training approach based on GAN, does not require annotated data and also works for both 2D and 3D.

Table 2 and **Suppl. Table 3** report the list of tools mentioned above, whilst **Suppl. Mat. 4** includes their short descriptions.

Table 2. Relevant tools for nucleus segmentation

Algorithm: A complete method to segment nuclei. An algorithm can be shared as a source code for developers in e.g. a GitHub repository or can be implemented as a user-accessible method in a platform. *Pipeline*: A workflow of image processing algorithms to segment nuclei, allowing the user to set parameters for each step of the workflow or even change the included algorithms to optimize segmentation tailored to the specific data. *Platform*: A software package that includes multiple algorithms or pipelines for nucleus segmentation, and often has a defined API to include additional methods as well.

2D/3D/Both	Tool name	Pipeline/algorithm/platform	Code availability	Year	Reference	GUI/Tutorial/Biaflows/GPU/Cloud
2D	U-Net	Algorithm	Yes	2015	Ronneberger et al.[34]	N/N/Y/Y/N
2D	SegNet	Algorithm	Yes	2015	Badrinarayanan et al. [87]	N/N/?/Y/N
2D	Mask R-CNN	Algorithm	Yes	2017	He et al.[57]	N/Y/Y/Y/N
2D	QuPath	Platform	Yes	2017	Bankhead et al.[47]	Y/Y/N/Y/N
2D	UNet++	Algorithm	Yes	2018	Zhou et al.[56]	N/N/N/Y/N
2D	Segmentation of Nuclei in Histopathology Images by deep regression of the distance map	Algorithm	Yes	2018	Naylor et al.[88]	N/Y/N/Y/N

2D	Multi-scale Cell Instance Segmentation with Keypoint Graph based Bounding Boxes	Algorithm	Yes	2019	Yi et al. [89]	N/N/?/Y/N
2D	HoVer-Net	Algorithm	Yes	2019	Graham et al.[23]	N/Y/N/Y/N
2D	CIA-Net	Algorithm	No	2019	Zhou et al.[90]	N/N/N/Y/N
2D	Bend-Net	Algorithm	No	2020	Wang et al.[91]	N/N/N/Y/N
2D	nucleAIzer	Algorithm, Pipeline	Yes	2020	Hollandi et al.[11]	Y/Y/N/Y/Y
2D	MultiStar	Algorithm	Yes	2020	Walter et al.[59]	N/N/N/Y/N
2D	Instance-Aware Self-supervised Learning for Nuclei Segmentation	Algorithm	No	2020	Xie et al. [85]	N/N/N/Y/N
2D	Self-supervised Nuclei Segmentation in Histopathological Images Using Attention	Algorithm	Yes	2020	Sahasrabudhe et al. [84,85]	N/N/N/Y/N
2D	Triple U-Net	Algorithm	Yes	2020	Zhao et al.[92]	N/N/N/Y/N

2D	“High-resolution deep transferred ASPPU-Net for nuclei segmentation of histopathology images”	Algorithm	No	2021	Chanchal et al.[93]	N/N/N/Y/N
2D	NucleiSegNet	Algorithm	Yes	2021	Lal et al.[94]	N/Y/N/Y/N
2D	SplineDist	Algorithm	Yes	2021	Mandal et al.[60]	N/N/N/Y/N
2D	Contour Proposal Network	Algorithm	Yes	2021	Upschulte et al.[95]	N/N/N/Y/N
2D	HistomicsML2	Pipeline, Platform	Yes	2021	Lee et al.[69]	Y/Y/N/Y/Y
2D	STORM	Pipeline	Yes	2021	Mela et al.[66]	N/N/N/Y/Y
2D	MSRF-Net	Algorithm	Yes	2021	Srivastava et al. [96]	N/N/N/Y/N
3D	“3D cell nuclei segmentation based on gradient flow tracking”	Algorithm	No	2007	Li et al.[97]	N/N/N/N/N
3D	Vaa3D	Platform	Yes	2010	Peng et al.[79]	Y/Y/Y/Y/N
3D	IT3DImageJSuite	Platform	Yes	2013	Ollion et al.[98]	Y/Y/N/N/N
3D	LoS	Algorithm	Yes	2013	Asafi et al.[74]	N/Y/N/N/N

3D	“Automated cell segmentation with 3D fluorescence microscopy images”	Algorithm	No	2015	Kong et al.[99]	N/N/N/N/N
3D	OpenSegSPIM	Platform	Yes	2016	Gole et al.[75]	Y/Y/N/N/N
3D	RACE	Platform	Yes	2016	Stegmaier et al.[76]	Y/Y/N/Y/N
3D	U-Net (3D)	Algorithm	Yes	2016	Cicek et al.[100]	N/N/N/Y/N
3D	“Segmentation of fluorescence microscopy images using three dimensional active contours with inhomogeneity correction”	Algorithm	No	2017	Lee et al.[14]	N/N/N/N/N
3D	DeepSynth	Algorithm	No	2019	Dunn et al.[101]	N/N/N/Y/N
3D	“Three-Dimensional Segmentation and Reconstruction of Neuronal Nuclei in Confocal Microscopic Images“	Algorithm	Yes	2019	Ruszczycki et al.[15]	N/N/N/N/N

3D	“Semi supervised segmentation and graph-based tracking of 3D nuclei in time-lapse microscopy“	Algorithm	Yes	2020	Shailja et al.[102]	N/N/N/Y/N
3D	“A deep learning pipeline for nucleus segmentation”	Pipeline	No	2020	Zaki et al.[103]	N/N/N/Y/N
3D	“Combined detection and segmentation of cell nuclei in microscopy images using deep learning”	Algorithm	No	2020	Ram et al.[104]	N/N/N/Y/N
3D	QCANet	Algorithm	Yes	2020	Tokuoka et al.[81]	N/Y/N/Y/N
3D	Allen Cell and Structure Segmenter	Platform	Yes	2020	Chen et al.[105]	Y/Y/N/Y/N
3D	3D-Cell-Annotator	Platform	Yes	2020	Tasnadi et al.[43]	Y/Y/N/Y/N
3D	“Nuclei detection for 3D microscopy with a fully convolutional regression network”	Algorithm	No	2021	Lapierre-Landry et al.[83]	N/N/N/Y/N

3D	3DeeCellTracker	Platform	Yes	2021	Wen et al.[82]	N/Y/N/Y/N
Both	MINS	Platform	Yes	2014	Lou et al.[62,63]	Y/Y/N/N/N
Both	XPIWIT	Algorithm	Yes	2016	Bartschat et al.[64]	Y/Y/N/Y/N
Both	ilastik	Platform	Yes	2018	Berg et al.[42]	Y/Y/Y/Y/N
Both	DeepImageJ	Platform	Yes	2019	Gómez-de-Mariscal et al.[71]	Y/Y/N/Y/N
Both	ImJoy	Platform	Yes	2019	Ouyang et al.[67]	Y/Y/N/Y/Y
Both	"A coarse-to-fine data generation method for 2D and 3D cell nucleus segmentation"	Algorithm	No	2020	Zhao et al.[106]	N/N/N/Y/N
Both	Cellpose	Algorithm	Yes	2020	Stringer et al.[12]	Y/Y/Y/Y/Y
Both	CDeep3M	Platform	Yes	2020	Haberl et al.[70]	Y/Y/N/Y/Y
Both	StarDist	Algorithm	Yes	2020	Shmidt et al.[13] ; Weigert et al.[54]	N/Y/Y/Y/N
Both	NuSeT	Platform	Yes	2020	Yang et al.[61]	Y/Y/N/Y/N
Both	nnU-Net	Platform	Yes	2021	Isensee et al.[55]	N/Y/N/Y/N
Both	DeepMIB	Platform	Yes	2021	Belevich et al.[53]	Y/Y/N/Y/N

Both	InstantDL	Pipeline, Platform	Yes	2021	Waibel et al.[107]	N/Y/N/Y/N
Both	ZeroCostDL4Mic	Pipeline, Platform	Yes	2021	von Chamier et al.[65]	Y/Y/N/Y/Y
Both	DeepCell Kiosk	Pipeline, Platform	Yes	2021; 2016	Bannon et al. [46,68]]; Van Valen et al.[46,68]	Y/Y/Y/Y/Y
Both	AD-GAN	Algorithm	No	2021	Yao et al. [86]	N/N/N/Y/N
Both	Embedding-based Instance Segmentation in Microscopy	Algorithm	Yes	2021	Lalit et al. [108]	N/Y/?/Y/N

Most of the listed tools require some effort from the user to install, prepare the environment, do the pre-processing of the input if needed, and finally to run it. The amount of time and effort primarily depends on the computational background of the user, and on the tool itself. Cloud-based tools (usually supplied with web GUI) could be the primary starter choices for life scientists. However, there is a trade-off: cloud-based versions of tools have limited customizability, while local versions are more flexible, and the user does not need to share the data with third-party services. In the latter case the quality of the documentation also matters to assure proper setup. In **Table 2** we provide information on whether the tool is documented properly (only official documentation was taken into account). The algorithms quite often lack detailed official documentation, though provide the most flexibility (usually are parts of the complex pipelines), and for the most popular ones unofficial documentation or tutorials and third-party implementations exist too. The potential performance of a tool is obviously an important concern for the user, and it might be challenging to decide on choosing the appropriate tool. The user may decide based on the community's preferences. Alternatively, a reliable comparison of the performance of the different tools can support decision-making. However, apart from BIAFLOWS[II] and automatic challenge submission systems (e.g. Kaggle or ISBI[VI]), the microscopy image analyst community lacks (*1*) an evaluation platform for the objective comparison of nucleus segmentation methods, using (*2*) a standardized evaluation metric in a transparent way. Thus, a consensus on utilizing a single, standardized platform is eagerly awaited. Since challenge portals only provide this functionality for the datasets of given challenges, a more inclusive platform, such as BIAFLOWS is suggested. Even though the relevance of newly published methods is usually supported by some quantitative segmentation results, it has several shortcomings from the user's point of view as follows. (*1*) The test dataset might not suggest relevant performance when the dataset size is too small, or covers a single imaging modality only. On the other hand, approaches developed for specific microscopy images (such as H&E or fluorescence confocal images) or segmentation scenarios (e.g. crowded cell culture) are intended to work in their given domain of images, and should not be expected to perform just as well on more extensive or general datasets. (*2*) When comparison to prior methods is performed and reported, the number of tested methods is usually low, and (*3*) additional model- or data-specific modifications might have been applied to the compared methods (or the test images as pre-processing), thus merely literature-based comparisons of accuracy scores may confuse the user (see **Suppl. Mat. 2**).

CONCLUDING REMARKS

Recent years have brought significant improvements in nucleus segmentation, including large annotated datasets, new high-accuracy 2D/3D strategies, deep learning approaches, and segmentation benchmarking platforms, however, establishing a genuinely general solution for nucleus segmentation is still an unmet need. In this review and the accompanying web-based portal[l] we aimed to cover the missing link between recent advancements and users' needs by providing a detailed overview on the available means for nucleus segmentation. The concluding remarks below are focused on crucial limitations and future goals.

The ***first crucial point*** is to cover more modalities of microscopy data for both 2D, and especially 3D, with open datasets of annotated images. Current methods are expected to work when trained on additional microscopy data modalities [18,109]. Most datasets include H&E stained tissues or fluorescently labelled cell cultures (see **Table 1**) which are two of the most widely used modalities in practice. However, further microscopy types (e.g. DIC, light-sheet or phase contrast) lack such publicly available annotations, except a recently published, large, label-free dataset [26]. Even though researchers can train existing deep learning methods on their own nowadays, these models remain private (unless released on e.g. GitHub, zenodo, Kaggle or in a Napari [110] plugin; on the first three platforms datasets may also be deposited [32]) and the initial datasets are small, resulting in suboptimal model generalization. For a given modality of interest, generalization is also a crucial point for medical applications: the data should be as diverse as possible to promote robust models. Diversity from a computer vision point of view would include various regions of tissue with the distinct visual appearance of both the target objects and the surroundings, as well as covering several phenotypes of cells, different batches or slightly different experimental setups. An extensive annotated dataset including most (if not all) modalities occuring in single-cell analysis experiments with respect to the type of microscopy, sample and label could definitely improve existing trainable methods. Besides, it would offer the possibility of releasing genuinely general pre-trained models, and would also serve as a standard dataset, similarly to the widely used COCO dataset [111] in computer vision.

The ***second crucial point*** relates to solving common microscopy challenges for both 2D and 3D data, such as: touching, overlapping and irregularly shaped nuclei [54][59][60][61]. Either dataset design or model architecture can be beneficial for a solution. Current methods achieve various levels of success in overcoming these issues, thus further developments are needed.

The ***third crucial point*** is the lack of (*1*) a globally accepted benchmark platform for comparison, and (*2*) a unified metric for tool evaluation. BIAFLOWS and Kaggle are available solutions to overcome these issues. However, still most publications presenting novel methods or tools typically provide limited comparisons (either in terms of the data used in evaluation or the number of methods compared) and use not standardized metrics. Accordingly, the results published by different authors are often difficult to compare.

The ***ultimate goal*** is to develop an algorithm, and train it so that the resulting single model would be able to accurately segment nuclei in a variety of microscopy modalities. Some of the available algorithms and models are aimed to meet this requirement [11,12], and the field is moving towards a generally applicable solution. While a quantitative comparison of the methods available for each modality is beyond our intention, it is worth mentioning that deep learning tends to provide fine accuracy in segmenting nuclei in images obtained with different microscopy techniques, as shown at the DSB 2018 challenge [6][112].

ACKNOWLEDGEMENTS

We would like to thank Krisztián Koós and Ákos Diósdi (BIOMAG, Szeged, Hungary) for sharing images used in the Figures and their important opinions on the segmentation tools; Michael Reil (Kutay lab, Institute of Biochemistry, Department of Biology, ETH Zurich, Zurich, Switzerland), Marie Follo (Lighthouse Core Facility, Medical Center, University of Freiburg), microscope-microscope.org, Frits Thorsen (Molecular Imaging Center, Department of Biomedicine, University of Bergen, Bergen, Norway), Dr Markus Kerr (Institute for Molecular Bioscience, University of Queensland, Australia) for sharing images; Csaba Molnár (Broad Institute, Cambridge, MA, USA) for helping with the implementation of *unbias*. ET and NM acknowledge support from the Cooperative Doctoral Programme of the Ministry for Innovation and Technology (KDP-2020). FP acknowledges support from the Union for International Cancer Control (UICC) for a UICC Yamagiwa-Yoshida (YY) Memorial International Cancer Study Grant (ref: UICC-YY/678329). PH, ET, NM and

RH acknowledge support from the LENDULET-BIOMAG Grant (2018-342), from the European Regional Development Funds (GINOP-2.3.2-15-2016-00006, GINOP-2.3.2-15-2016-00026, GINOP-2.3.2-15-2016-00037), from COMPASS-ERAPerMed H2020, from CZI Deep Visual Proteomics, from H2020-DiscovAir, and from the ELKH-Excellence grant. LP acknowledges the FIMM High Content Imaging and Analysis unit (HiLIFE, University of Helsinki and Biocenter Finland) and the grants from Academy of Finland (340273, 337036). The authors thank Dora Bokor, PharmD (Szeged, Hungary), for proofreading the manuscript.

RESOURCES

I. https://biomag-lab.github.io/microscopy-tree/
II. https://biaflows.neubias.org
III. imagej.net/plugins/labkit
IV. http://cancergenome.nih.gov/
V. http://www.andrewjanowczyk.com/deep-learning/
VI. https://biomedicalimaging.org/

REFERENCES

1 Meijering, E. (2012) Cell Segmentation: 50 Years Down the Road [Life Sciences]. *IEEE Signal Process. Mag.* 29, 140–145

2 Hallou, A. *et al.* (2021) Deep learning for bioimage analysis in developmental biology. *Development* 148,

3 Chandrasekaran, S.N. *et al.* (2021) Image-based profiling for drug discovery: due for a machine-learning upgrade? *Nat. Rev. Drug Discov.* 20, 145–159

4 Dhar, R. *et al.* (2019) Single cell functional genomics reveals the importance of mitochondria in cell-to-cell phenotypic variation. *Elife* 8,

5 Hayakawa, T. *et al.* (2021) Computational nuclei segmentation methods in digital pathology: A survey. *Arch. Comput. Methods Eng.* 28, 1–13

6 Caicedo, J.C. *et al.* (2019) Nucleus segmentation across imaging experiments: the 2018 Data Science Bowl. *Nat. Methods* 16, 1247–1253

7 Greenwald, N.F. *et al.* (2021) Whole-cell segmentation of tissue images with

human-level performance using large-scale data annotation and deep learning. *Nat. Biotechnol.* DOI: 10.1038/s41587-021-01094-0

8 Pelkmans, L. *et al.* (2005) Genome-wide analysis of human kinases in clathrin- and caveolae/raft-mediated endocytosis. *Nature* 436, 78–86

9 Kumar, N. *et al.* (2020) A Multi-Organ Nucleus Segmentation Challenge. *IEEE Trans. Med. Imaging* 39, 1380–1391

10 Caicedo, J.C. *et al.* (2019) Evaluation of Deep Learning Strategies for Nucleus Segmentation in Fluorescence Images. *Cytometry A* 95, 952–965

11 Hollandi, R. *et al.* (2020) NucleAIzer: A parameter-free deep learning framework for nucleus segmentation using image style transfer. *Cell Syst.* 10, 453–458.e6

12 Stringer, C. *et al.* (2021) Cellpose: a generalist algorithm for cellular segmentation. *Nat. Methods* 18, 100–106

13 Weigert, M. *et al.* Star-convex Polyhedra for 3D Object Detection and Segmentation in Microscopy. , *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*. (2020)

14 Lee, S. *et al.* (2017) , Segmentation of fluorescence microscopy images using three dimensional active contours with inhomogeneity correction. , in *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*, pp. 709–713

15 Ruszczycki, B. *et al.* (2019) Three-Dimensional Segmentation and Reconstruction of Neuronal Nuclei in Confocal Microscopic Images. *Front. Neuroanat.* 13, 81

16 Amgad, M. *et al.* 18-Feb-(2021) , NuCLS: A scalable crowdsourcing, deep learning approach and dataset for nucleus classification, localization and segmentation. , *arXiv [cs.CV]*

17 Rubens, U. *et al.* (2020) BIAFLOWS: A Collaborative Framework to Reproducibly Deploy and Benchmark Bioimage Analysis Workflows. *Patterns (N Y)* 1, 100040

18 Alemi Koohbanani, N. *et al.* (2020) NuClick: A deep learning framework for interactive segmentation of microscopic images. *Med. Image Anal.* 65, 101771

19 Ljosa, V. *et al.* (2012) Annotated high-throughput microscopy image sets for validation. *Nat. Methods* 9, 637

20 Kromp, F. *et al.* (2020) An annotated fluorescence image dataset for training nuclear segmentation methods. *Sci Data* 7, 262

21 Rivron, N.C. *et al.* (2018) Blastocyst-like structures generated solely from stem cells. *Nature* 557, 106–111

22 Yao, K. *et al.* (2021) Scaffold-A549: A benchmark 3D fluorescence image dataset for unsupervised nuclei segmentation. *Cognit. Comput.* DOI: 10.1007/s12559-021-09944-4

23 Graham, S. *et al.* (2019) Hover-Net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images. *Med. Image Anal.* 58, 101563

24 Mahbod, A. *et al.* (2021) CryoNuSeg: A dataset for nuclei instance segmentation of cryosectioned H&E-stained histological images. *Comput. Biol. Med.* 132, 104349

25 Janowczyk, A. and Madabhushi, A. (2016) Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases. *J. Pathol. Inform.* 7, 29

26 Edlund, C. *et al.* (2021) LIVECell-A large-scale dataset for label-free live cell segmentation. *Nat. Methods* 18, 1038–1045

27 Graham, S. *et al.* 25-Aug-(2021) , Lizard: A Large-Scale Dataset for Colonic

Nuclear Instance Segmentation and Classification. , *arXiv [cs.CV]*

28 Kumar, N. *et al.* (2017) A Dataset and a Technique for Generalized Nuclear Segmentation for Computational Pathology. *IEEE Trans. Med. Imaging* 36, 1550–1560

29 Lin, Z. *et al.* (2021) , NucMM Dataset: 3D Neuronal Nuclei Instance Segmentation at Sub-Cubic Millimeter Scale. , in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*, pp. 164–174

30 Gamper, J. *et al.* PanNuke: An Open Pan-Cancer Histology Dataset for Nuclei Instance Segmentation and Classification. , *Digital Pathology*. (2019) , 11–19

31 Irshad, H. *et al.* (2015) Crowdsourcing image annotation for nucleus detection and segmentation in computational pathology: evaluating experts, automated methods, and the crowd. *Pac. Symp. Biocomput.*

32 Jack, N.P. *et al.* Segmentation of Nuclei in Histopathology Images by deep regression of the distance map. . (2018) , Zenodo

33 Wienert, S. *et al.* (2012) Detection and segmentation of cell nuclei in virtual microscopy images: a minimum-model approach. *Sci. Rep.* 2, 503

34 Ronneberger, O. *et al.* (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Lecture Notes in Computer Science* pp. 234–241, Springer International Publishing

35 Ulman, V. *et al.* (2017) An objective comparison of cell-tracking algorithms. *Nat. Methods* 14, 1141–1152

36 Hollandi, R. *et al.* (2020) AnnotatorJ: an ImageJ plugin to ease hand annotation of cellular compartments. *Mol. Biol. Cell* 31, 2179–2186

37 Borland, D. *et al.* (2021) Segmentor: a tool for manual refinement of 3D microscopy annotations. *BMC Bioinformatics* 22, 260

38 Schneider, C.A. *et al.* NIH Image to ImageJ: 25 years of image analysis. , *Nature Methods*, 9. (2012) , 671–675

39 Schindelin, J. *et al.* Fiji: an open-source platform for biological-image analysis. , *Nature Methods*, 9. (2012) , 676–682

40 Nolden, M. *et al.* (2013) The Medical Imaging Interaction Toolkit: challenges and advances : 10 years of open-source development. *Int. J. Comput. Assist. Radiol. Surg.* 8, 607–620

41 Arganda-Carreras, I. *et al.* (2017) Trainable Weka Segmentation: a machine learning tool for microscopy pixel classification. *Bioinformatics* 33, 2424–2426

42 Berg, S. *et al.* (2019) ilastik: interactive machine learning for (bio)image analysis. *Nat. Methods* 16, 1226–1232

43 Tasnadi, E.A. *et al.* (2020) 3D-Cell-Annotator: an open-source active surface tool for single-cell segmentation in 3D microscopy images. *Bioinformatics* 36, 2948–2949

44 Marée, R. *et al.* (2016) Collaborative analysis of multi-gigapixel imaging data using Cytomine. *Bioinformatics* 32, 1395–1401

45 Rubens, U. *et al.* (2019) Cytomine: Toward an Open and Collaborative Software Platform for Digital Pathology Bridged to Molecular Investigations. *Proteomics Clin. Appl.* 13, e1800057

46 Bannon, D. *et al.* (2021) DeepCell Kiosk: scaling deep learning-enabled cellular image analysis with Kubernetes. *Nat. Methods* 18, 43–45

47 Bankhead, P. *et al.* (2017) QuPath: Open source software for digital pathology image analysis. *Sci. Rep.* 7, 16878

48 Peng, T. *et al.* (2017) A BaSiC tool for background and shading correction of optical microscopy images. *Nat. Commun.* 8, 14836

49 Smith, K. *et al.* (2015) CIDRE: an illumination-correction method for optical microscopy. *Nat. Methods* 12, 404–406

50 Goyal, B. *et al.* (2020) Image denoising review: From classical to state-of-the-art approaches. *Inf. Fusion* 55, 220–244

51 Moshkov, N. *et al.* (2020) Test-time augmentation for deep learning-based cell segmentation on microscopy images. *Sci. Rep.* 10, 5068

52 McQuin, C. *et al.* (2018) CellProfiler 3.0: Next-generation image processing for biology. *PLoS Biol.* 16, e2005970

53 Belevich, I. and Jokitalo, E. (2021) DeepMIB: User-friendly and open-source software for training of deep learning network for biological image segmentation. *PLoS Comput. Biol.* 17, e1008374

54 Schmidt, U. *et al.* (2018) Cell detection with star-convex polygons. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018* pp. 265–273, Springer International Publishing

55 Isensee, F. *et al.* (2021) nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. *Nat. Methods* 18, 203–211

56 Zhou, Z. *et al.* (2018) UNet++: A Nested U-Net Architecture for Medical Image Segmentation. *Deep Learn Med Image Anal Multimodal Learn Clin Decis Support (2018)* 11045, 3–11

57 He, K. *et al.* (2017) , Mask R-CNN. , in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice

58 Ren, S. *et al.* (2017) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 1137–1149

59 Walter, F.C. *et al.* (2021) , Multistar: Instance segmentation of overlapping objects with star-convex polygons. , in *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, Nice, France

60 Mandal, S. and Uhlmann, V. (2021) , Splinedist: Automated Cell Segmentation With Spline Curves. , in *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pp. 1082–1086

61 Yang, L. *et al.* (2020) NuSeT: A deep learning tool for reliably separating and analyzing crowded cells. *PLoS Comput. Biol.* 16, e1008193

62 Lou, X. *et al.* (2014) A rapid and efficient 2D/3D nuclear segmentation method for analysis of early mouse embryo and stem cell image data. *Stem Cell Reports* 2, 382–397

63 Saiz, N. *et al.* (2016) Quantitative Analysis of Protein Expression to Study Lineage Specification in Mouse Preimplantation Embryos. *J. Vis. Exp.*

64 Bartschat, A. *et al.* (2016) XPIWIT--an XML pipeline wrapper for the Insight Toolkit. *Bioinformatics* 32, 315–317

65 von Chamier, L. *et al.* (2021) Democratising deep learning for microscopy with ZeroCostDL4Mic. *Nat. Commun.* 12, 2276

66 Mela, C.A. and Liu, Y. (2021) Application of convolutional neural networks towards nuclei segmentation in localization-based super-resolution fluorescence microscopy images. *BMC Bioinformatics* 22, 325

67 Ouyang, W. *et al.* (2019) ImJoy: an open-source computational platform for the deep learning era. *Nat. Methods* 16, 1199–1200

68 Van Valen, D.A. *et al.* (2016) Deep Learning Automates the Quantitative Analysis of Individual Cells in Live-Cell Imaging Experiments. *PLoS Comput. Biol.* 12, e1005177

69 Lee, S. *et al.* (2021) Interactive Classification of Whole-Slide Imaging Data for

Cancer Researchers. *Cancer Res.* 81, 1171–1177

70 Haberl, M.G. *et al.* (2018) CDeep3M-Plug-and-Play cloud-based deep learning for image segmentation. *Nat. Methods* 15, 677–680

71 Gómez-de-Mariscal, E. *et al.* 16-Oct-(2019) , DeepImageJ: A user-friendly environment to run deep learning models in ImageJ. , *bioRxiv*, bioRxiv

72 Kar, A. *et al.* 10-Jun-(2021) , Assessment of deep learning algorithms for 3D instance segmentation of confocal image datasets. , *bioRxiv*, 2021.06.09.447748

73 Piccinini, F. *et al.* (2020) Software tools for 3D nuclei segmentation and quantitative analysis in multicellular aggregates. *Comput. Struct. Biotechnol. J.* 18, 1287–1300

74 Asafi, S. *et al.* (2013) Weak convex decomposition by lines-of-sight. *Comput. Graph. Forum* 32, 23–31

75 Gole, L. *et al.* (2016) OpenSegSPIM: a user-friendly segmentation tool for SPIM data. *Bioinformatics* 32, 2075–2077

76 Stegmaier, J. *et al.* (2016) Real-Time Three-Dimensional Cell Segmentation in Large-Scale Microscopy Data of Developing Embryos. *Dev. Cell* 36, 225–240

77 Kankaanpää, P. *et al.* (2012) BioImageXD: an open, general-purpose and high-throughput image-processing platform. *Nat. Methods* 9, 683–689

78 de Chaumont, F. *et al.* (2012) Icy: an open bioimage informatics platform for extended reproducible research. *Nat. Methods* 9, 690–696

79 Peng, H. *et al.* (2010) V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets. *Nat. Biotechnol.* 28, 348–353

80 Long, F. *et al.* (2009) A 3D digital atlas of C. elegans and its application to single-cell analyses. *Nat. Methods* 6, 667–672

81 Tokuoka, Y. *et al.* (2020) 3D convolutional neural networks-based segmentation to acquire quantitative criteria of the nucleus during mouse embryogenesis. *NPJ Syst Biol Appl* 6, 32

82 Wen, C. *et al.* (2021) 3DeeCellTracker, a deep learning-based pipeline for segmenting and tracking cells in 3D time lapse images. *Elife* 10,

83 Lapierre-Landry, M. *et al.* (2021) Nuclei Detection for 3D Microscopy With a Fully Convolutional Regression Network. *IEEE Access* 9, 60396–60408

84 Sahasrabudhe, M. *et al.* (2020) Self-supervised nuclei segmentation in histopathological images using attention. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020* pp. 393–402, Springer International Publishing

85 Xie, X. *et al.* (2020) , Instance-Aware Self-supervised Learning for Nuclei Segmentation. , in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, pp. 341–350

86 Yao, K. *et al.* 23-Jul-(2021) , AD-GAN: End-to-end Unsupervised Nuclei Segmentation with Aligned Disentangling Training.

87 Badrinarayanan, V. *et al.* (2017) SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 2481–2495

88 Naylor, P. *et al.* (2019) Segmentation of Nuclei in Histopathology Images by Deep Regression of the Distance Map. *IEEE Trans. Med. Imaging* 38, 448–459

89 Yi, J. *et al.* (2019) Multi-scale cell instance segmentation with keypoint graph based bounding boxes. In *Lecture Notes in Computer Science* pp. 369–377, Springer International Publishing

90 Zhou, Y. *et al.* (2019) CIA-net: Robust nuclei instance segmentation with

contour-aware information aggregation. In *Lecture Notes in Computer Science* pp. 682–693, Springer International Publishing

91 Wang, H. *et al.* (2020) BENDING LOSS REGULARIZED NETWORK FOR NUCLEI SEGMENTATION IN HISTOPATHOLOGY IMAGES. *Proc. IEEE Int. Symp. Biomed. Imaging* 2020, 258–262

92 Zhao, B. *et al.* (2020) Triple U-net: Hematoxylin-aware nuclei segmentation with progressive dense feature aggregation. *Med. Image Anal.* 65, 101786

93 Chanchal, A.K. *et al.* (2021) High-resolution deep transferred ASPPU-Net for nuclei segmentation of histopathology images. *Int. J. Comput. Assist. Radiol. Surg.* DOI: 10.1007/s11548-021-02497-9

94 Lal, S. *et al.* (2021) NucleiSegNet: Robust deep learning architecture for the nuclei segmentation of liver cancer histopathology images. *Comput. Biol. Med.* 128, 104075

95 Upschulte, E. *et al.* 07-Apr-(2021) , Contour Proposal Networks for Biomedical Instance Segmentation. , *arXiv [cs.CV]*

96 Srivastava, A. *et al.* 16-May-(2021) , MSRF-Net: A Multi-Scale Residual Fusion Network for Biomedical Image Segmentation.

97 Li, G. *et al.* (2007) 3D cell nuclei segmentation based on gradient flow tracking. *BMC Cell Biol.* 8, 40

98 Ollion, J. *et al.* (2013) TANGO: a generic tool for high-throughput 3D image analysis for studying nuclear organization. *Bioinformatics* 29, 1840–1841

99 Kong, J. *et al.* (2015) AUTOMATED CELL SEGMENTATION WITH 3D FLUORESCENCE MICROSCOPY IMAGES. *Proc. IEEE Int. Symp. Biomed. Imaging* 2015, 1212–1215

100 Çiçek, Ö. *et al.* (2016) 3D U-net: Learning dense volumetric segmentation from sparse annotation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016* pp. 424–432, Springer International Publishing

101 Dunn, K.W. *et al.* (2019) DeepSynth: Three-dimensional nuclear segmentation of biological images using neural networks trained with synthetic data. *Sci. Rep.* 9, 18295

102 Shailja, S. *et al.* (2021) , Semi Supervised Segmentation And Graph-Based Tracking Of 3d Nuclei In Time-Lapse Microscopy. , in *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pp. 385–389

103 Zaki, G. *et al.* (2020) A Deep Learning Pipeline for Nucleus Segmentation. *Cytometry A* 97, 1248–1264

104 Ram, S. *et al.* (2020) , Combined Detection and Segmentation of Cell Nuclei in Microscopy Images Using Deep Learning. , in *2020 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, pp. 26–29

105 Chen, J. *et al.* 08-Dec-(2018) , The Allen Cell and Structure Segmenter: a new open source toolkit for segmenting 3D intracellular structures in fluorescence microscopy images. , *bioRxiv*, bioRxiv

106 Zhao, Z. *et al.* (2020) , A Coarse-to-Fine Data Generation Method for 2D and 3D Cell Nucleus Segmentation. , in *2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 41–46

107 Waibel, D.J.E. *et al.* (2021) InstantDL: an easy-to-use deep learning pipeline for image segmentation and classification. *BMC Bioinformatics* 22, 103

108 Lalit, M. *et al.* 25-Jan-(2021) , Embedding-based Instance Segmentation in Microscopy.

109 Fishman, D. *et al.* (2021) Practical segmentation of nuclei in brightfield cell images with neural networks trained on fluorescently labelled samples. *J.*

Microsc. 284, 12–24
110 Sofroniew, N. *et al.* (2021) *napari/napari: 0.4.11*, Zenodo.
111 Lin, T.-Y. *et al.* 01-May-(2014) , Microsoft COCO: Common Objects in Context.
112 Verma, R. *et al.* (2021) MoNuSAC2020: A Multi-organ Nuclei Segmentation and Classification Challenge. *IEEE Trans. Med. Imaging*

FIGURES

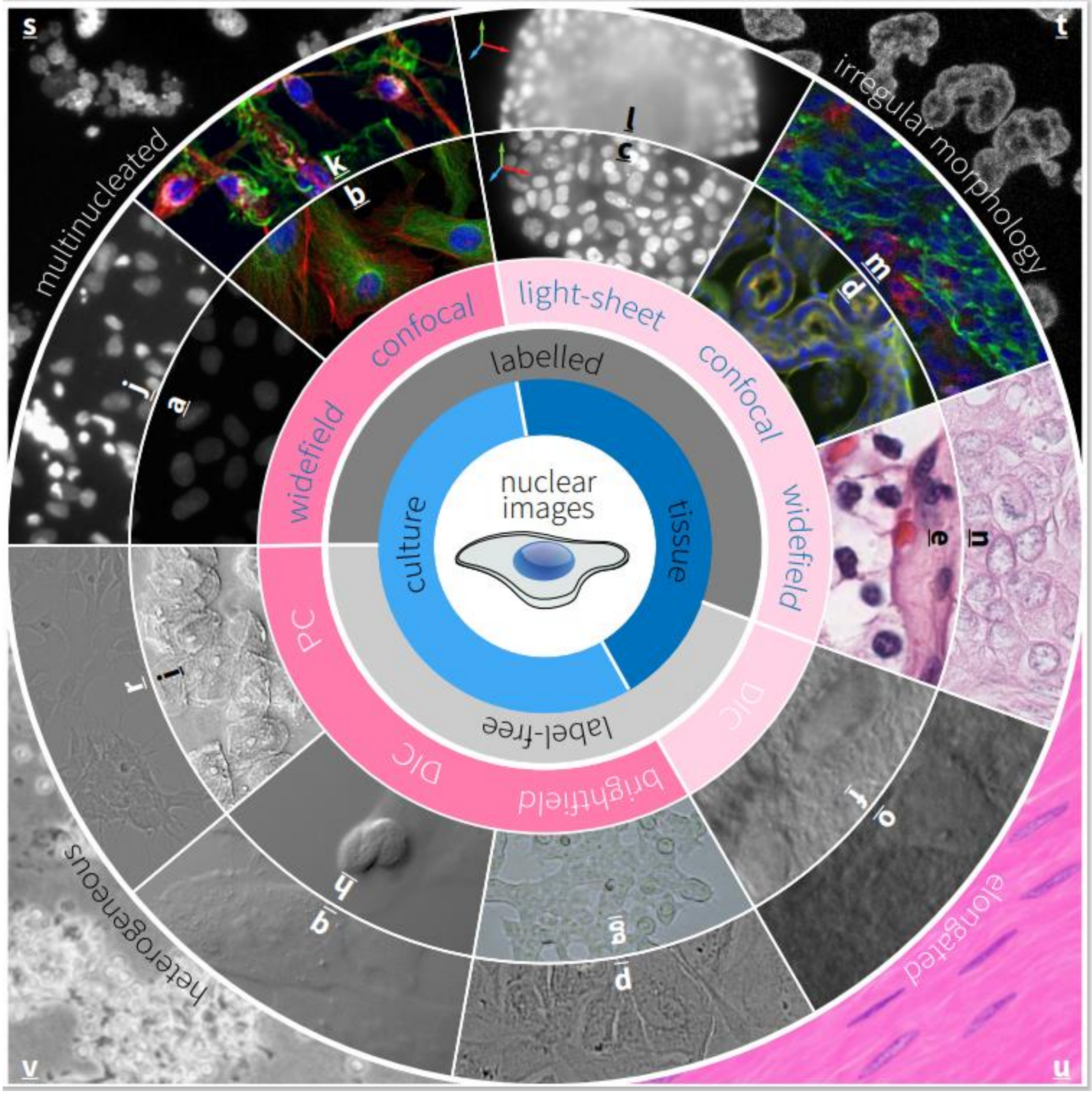
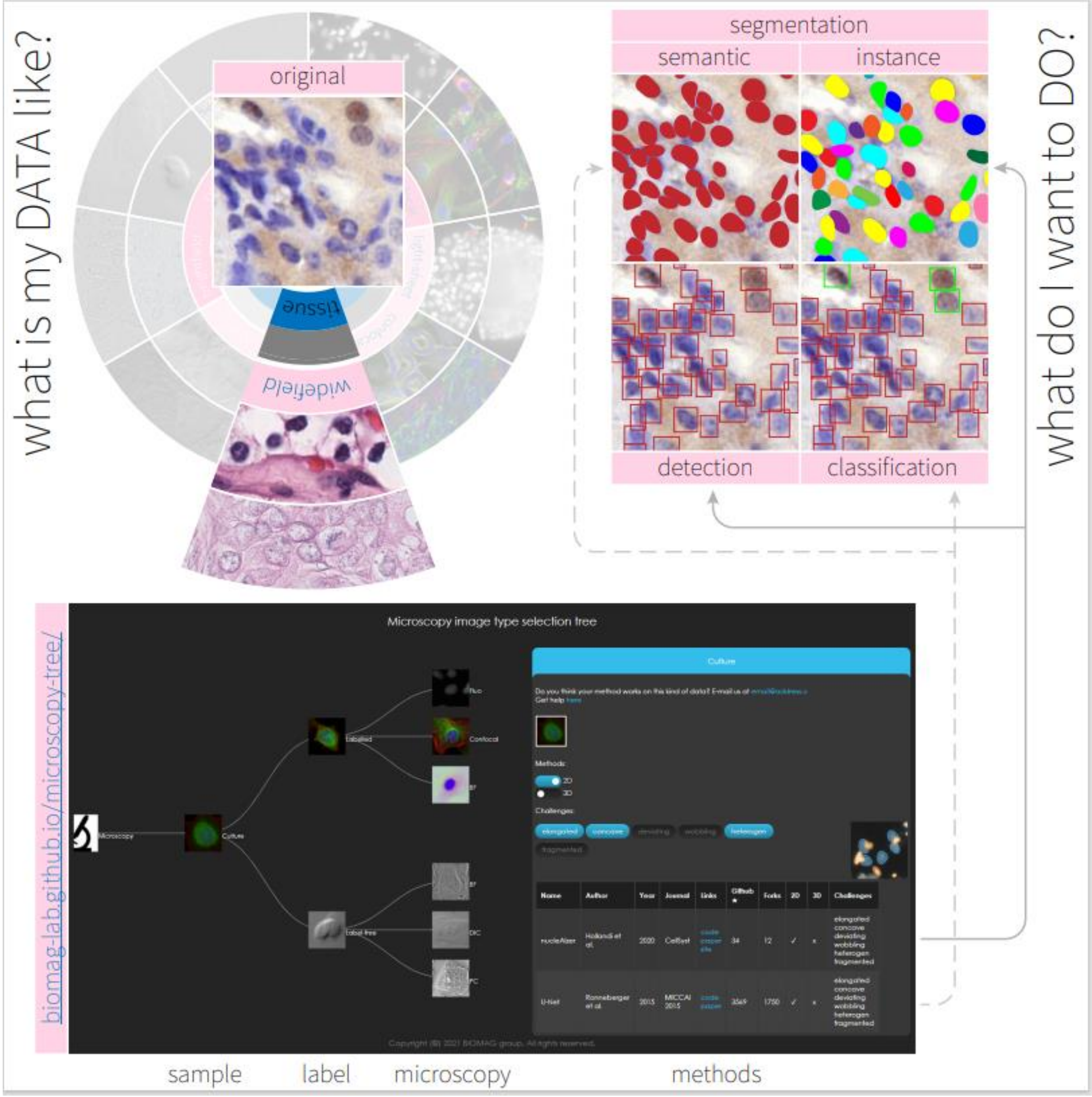


Figure 1. Diversity of optical microscopy images representing nuclei. The inner circle shows standard examples of each type (e.g. widefield, confocal, light-sheet, DIC, PC

images), while the outer circle presents more difficult cases. Finally, common challenging cases (e.g. multinucleated cells, irregular morphology, elongated shape, heterogeneous samples) regarding nucleus segmentation are reported in the corners. **c**,**d**, **l**, **f**, **g**, **o, t** are images from our laboratories/collaborators; **a**, **s, j** are from BBBC collection; **e**, **n** are from the TCGA collection; **r** is from the LIVECell dataset; the remaining ones are from the internet (see **Supplementary Table 1** for the sources).



Microscopy image type selection tree

Culture

Do you think your method works on this kind of data? E-mail us at [illegible]
Get help here

Methods:

● 2D
○ 3D

Challenges:

elongated concave deviating wobbling heterogen fragmented

Name	Author	Year	Journal	Links	Github ★	Forks	2D	3D	Challenges
nucleAlzer	Hollandi et al.	2020	CellSyst	code paper site	34	12	✓	x	elongated concave deviating wobbling heterogen fragmented
U-Net	Ronneberger et al.	2015	MICCAI 2015	code paper	3569	1750	✓	x	elongated concave deviating wobbling heterogen fragmented

Copyright (B) 2021 BIOMAG group. All rights reserved.

Figure 2. A sample-driven guide to select an appropriate method for single nucleus segmentation. Firstly, based on the images of the given experiment the user can determine the category (e.g. widefield, confocal, light-sheet, DIC, PC images) and select the corresponding node in the interactive online tool *unbias*^{I} according to the sample, label and microscopy type. Then, a list of segmentation methods is shown in the table on the right, including the method description and implementation if available alongside pre-trained models. The list may be filtered with the buttons above the table by dimension (2D/3D) and challenging segmentation issues (e.g. elongated nucleus in smooth muscle tissue). Finally, the goal of the experiment (e.g. object-aware segmentation or additional phenotyping i.e. classification) guides the user to select the appropriate segmentation method.

B Article ‘Test-time augmentation for deep learning-based cell segmentation on microscopy images‘

www.nature.com/scientificreports

SCIENTIFIC REPORTS

natureresearch

There are amendments to this paper

OPEN

Test-time augmentation for deep learning-based cell segmentation on microscopy images

Nikita Moshkov[1,2,3], Botond Mathe[1], Attila Kertesz-Farkas[3], Reka Hollandi[1] & Peter Horvath[1,4*]

Recent advancements in deep learning have revolutionized the way microscopy images of cells are processed. Deep learning network architectures have a large number of parameters, thus, in order to reach high accuracy, they require a massive amount of annotated data. A common way of improving accuracy builds on the artificial increase of the training set by using different augmentation techniques. A less common way relies on test-time augmentation (TTA) which yields transformed versions of the image for prediction and the results are merged. In this paper we describe how we have incorporated the test-time argumentation prediction method into two major segmentation approaches utilized in the single-cell analysis of microscopy images. These approaches are semantic segmentation based on the U-Net, and instance segmentation based on the Mask R-CNN models. Our findings show that even if only simple test-time augmentations (such as rotation or flipping and proper merging methods) are applied, TTA can significantly improve prediction accuracy. We have utilized images of tissue and cell cultures from the Data Science Bowl (DSB) 2018 nuclei segmentation competition and other sources. Additionally, boosting the highest-scoring method of the DSB with TTA, we could further improve prediction accuracy, and our method has reached an ever-best score at the DSB.

Identifying objects at the single-cell level is the starting point of most microscopy-based quantitative cellular image analysis tasks. Precise segmentation of the cell's nucleus is a major challenge here. Numerous approaches have been developed, including methods based on mathematical morphology[1] or differential geometry[2,3]. More recently, deep learning has yielded a never-seen improvement of accuracy and robustness[4–6]. Remarkably, Kaggle's Data Science Bowl 2018 (DSB)[7] was dedicated to nuclei segmentation, and gave a great momentum to this field. Deep learning-based approaches have proved their effectiveness: practically all the teams used some type of a deep architecture in the first few hundred leaderboard positions. The most popular architectures included U-Net[4], originally designed for medical image segmentation, and Mask R-CNN[8], used for instance segmentation of natural objects.

Deep learning approaches for object segmentation require a large, and often pixel-wise annotated dataset for training. This task relies on high-quality samples and domain experts to accurately annotate images. Besides, analysing biological images is challenging because of their heterogeneity and, sometimes, poorer quality compared to natural images. In addition, ground truth masks might be imperfect due to the annotator-related bias, which introduces further uncertainty. Consequently, a plethora of annotated samples is required, making object segmentation a laborious process. One of the techniques utilized to improve the model is data augmentation[9] of the training set. Conventionally, a transformation (i.e. rotation, flipping, noise addition, etc.) or a series of transformations are applied on the original images. Data augmentation has become the *de facto* technique in deep learning, especially in the case of heterogeneous or small datasets, to improve the accuracy of cell-based analysis.

Another option of improving performance relies on augmenting both the training and the test datasets, then performing the prediction both on the original and on the augmented versions of the image, followed by merging the predictions. This approach is called ***test-time augmentation*** (Fig. 1). This technique was successfully used in image classification tasks[10], for aleatoric uncertainty estimation[11], as well as for the segmentation of MRI slices/MRI volumes[12]. A theoretical formulation[12] of test-time augmentation has recently been described by Wang *et al*. Their experiments show that TTA helps to eliminate overconfident incorrect predictions. Additionally, a

[1]Biological Research Centre, Szeged, Hungary. [2]University of Szeged, Szeged, Hungary. [3]National Research University, Higher School of Economics, Moscow, Russia. [4]Institute for Molecular Medicine Finland, University of Helsinki, Helsinki, Finland. *email: horvath.peter@brc.hu

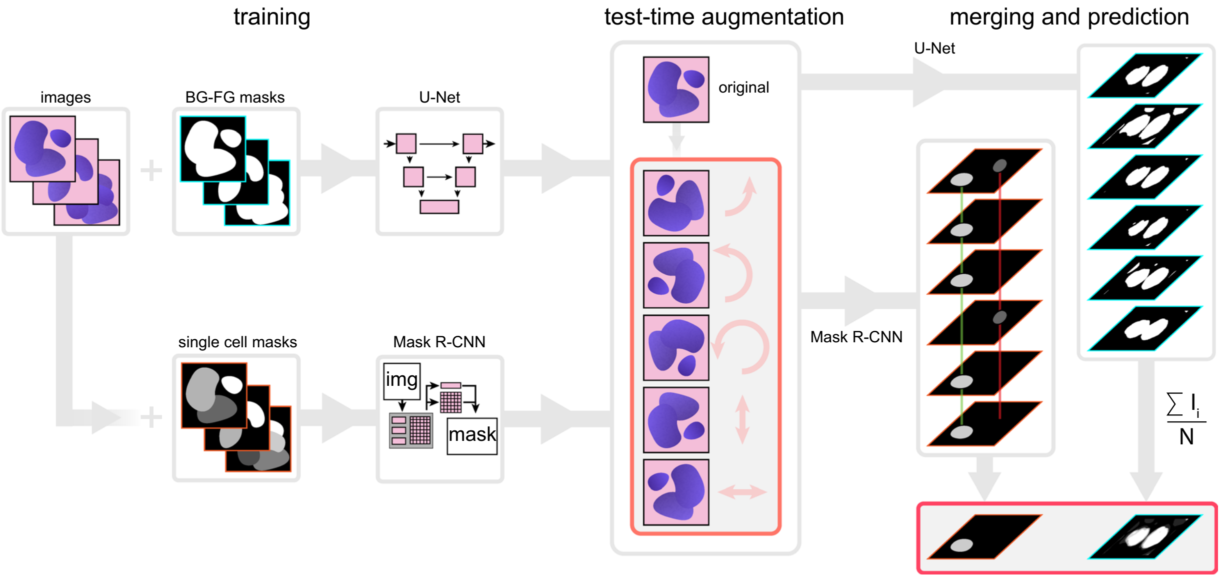


Figure 1. Principle of the proposed test-time augmentation techniques. Several augmented instances of the same test images are predicted, and the results are transformed back and merged. In the case of U-Net, pixel-wise majority voting was applied, while for Mask R-CNN a combination of object matching and majority voting was applied.

framework[13] has also been proposed for quantifying the uncertainty of the deep neural network (DNN) model for diagnosing diabetic retinopathy based on test-time data augmentation. Its disadvantage is increased prediction time, as it is run not only on the original image, but on all of its augmentations as well.

In the current paper we assess the impact and describe cases of utilizing test-time augmentation for deep-learning models trained on microscopy datasets. We have trained deep learning models for semantic segmentation (when the network only distinguishes the foreground from the background, using the U-Net architecture) and instance segmentation (when the network assigns labels to separate objects, using the Mask R-CNN architecture) (Fig. 1). Test-time augmentation has outperformed single instance predictions at each test case, and could further improve the best result of the DSB, as demonstrated by the improvement of the score, changing from 0.633 to 0.644.

Methods

Dataset acquisition and description. We have used two datasets: fluorescent microscopy images (further referred to as 'fluorescent' dataset) and histopathology images (further referred to as 'tissue' dataset). Most of the images have come from the stage 1 train/test data of Data Science Bowl 2018. We also used additional sources[14–20] and other data published in the discussion thread 'Official External Data Thread' (https://www.kaggle.com/c/data-science-bowl-2018/discussion/47572) related to DSB 2018. The images were labelled by experts using the annotation plugins of ImageJ/Fiji and Gimp. Both datasets were divided into three holdout train/test sets: approximately 5%, 15% (6 splits for each, cross-validation), and 30% (further referred to as '5', '15' and '30' in the dataset name, respectively) of uncropped images were held out as the test set. The test sets ('5', first cross-validation split of '15' and '30') did not intersect.

We used the same augmentations (horizontal and vertical flip, 90°, 180° and 270° rotations) for training both architectures. The images were cropped to the size of 512 × 512 pixels. Crops from the same image were used only in either the train or test set. Images with a resolution of less than 512 × 512 were resized to that particular size. Sample images are shown in Fig. 2.

Deep learning models and training. These augmented and cropped training data were used to train the models. For each dataset (5, 15 (6-fold cross validation) and 30 holdouts for both fluorescent and tissue images) separate models were trained. Additionally, we also trained U-Net without augmented data to analyse TTA performance on such a network as well (just 1 holdout 15 test set in that case).

Mask R-CNN (implementation[21]) is an extension of Faster R-CNN, the architecture for object detection. Solutions based on Mask R-CNN outperform the COCO 2016 challenge winners, and finished at the third place in Kaggle Data Science Bowl 2018[7]. The architecture of Mask R-CNN incorporates the following main stages: (1) Region proposal network (RPN) to propose candidate bounding boxes. It uses a backbone: a convolutional neural network which serves as a feature extractor. In this implementation it is possible to use *resnet50* or *resnet101* as a backbone, and we used *resnet101*. (2) Network head layers: they predict the class, box offset and an output binary mask for each region of interest (RoI). Masks are generated for each class without competition between the classes.

Following the strategy described by Hollandi *et al.*[5], the network was trained for 3 epochs for different layer groups: first, all network layers were trained at a learning rate of 10^{-3}, then training was restricted to ResNet stage

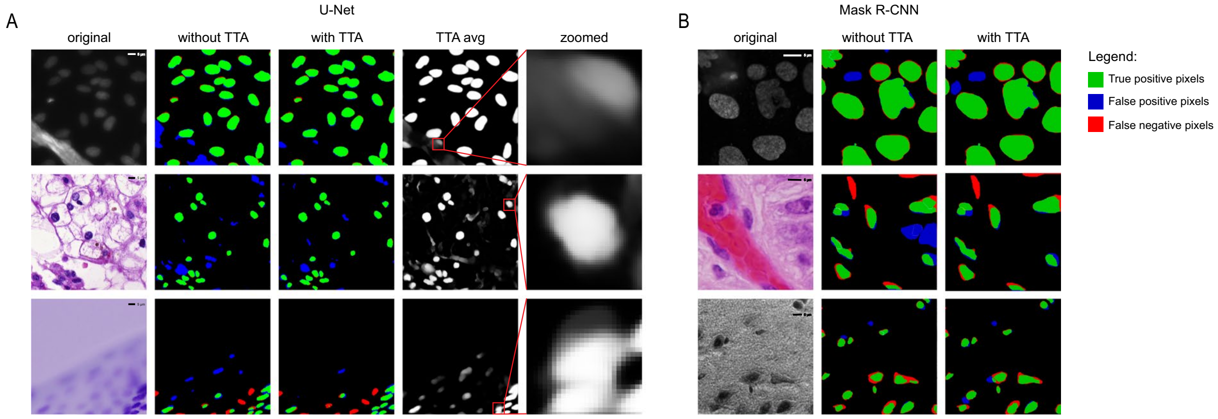


Figure 2. Examples of predictions. (**A**) U-Net predictions. First column - original image, second column - predictions without TTA compared to ground truth, third column - predictions with TTA compared to ground truth. Red indicates false negative pixels, green indicates true positive pixels and blue indicates false positive pixels. Dividing lines: yellow is false positive division of pixels into objects, and cyan is false negative division of pixels into objects. Fourth column - averaged TTA predictions before thresholding, fifth column - zoomed insets from the previous column. (**B**) Mask R-CNN predictions. Columns are the same as the first three columns in (**A**). Images in line 1 are examples of the fluorescent dataset, images in line 2 and 3 are examples of the tissue dataset.

5 (ResNet consists of 5 stages, each with convolution and identity blocks including 3 convolutional layers per block) and head layers at a learning rate of 5×10^{-4}, and finally only the head layers were trained at a learning rate of 10^{-4}. The model was initialized with pre-trained weights (https://github.com/matterport/Mask_RCNN/releases/download/v1.0/mask_rcnn_coco.h5) on the COCO dataset. The loss function of the architecture was binary cross-entropy with ADAM[22] (Adaptive Moment Estimation) solver, batch size 1, the number of iterations being equal to the train set size.

U-Net (implementation[23]) is an architecture originally designed to process biological images, which proved to be efficient, even when utilizing small training datasets. U-Net based solutions won the 2015 ISBI cell tracking challenge[4] and Kaggle Data Science Bowl 2018. Its architecture consists of two main parts: (1) a down-sampling convolution network or encoder by which we obtain the feature representation of the input image, and (2) an up-sampling convolution network or decoder, which produces the segmentation from a feature representation of the input image.

We trained U-Net for 200 epochs at a constant learning rate of 3×10^{-4}, and used a binary cross-entropy loss function with ADAM solver, batch size 1, the number of iterations being equal to the train set size.

Both U-Net and Mask R-CNN implementations are based on the deep learning framework Keras with Tensorflow backend. The training computations were conducted on a PC with NVIDIA Titan Xp GPU, 32 GB RAM and Core-i7 CPU.

Test-time augmentation.

Test-time augmentation includes four procedures: augmentation, prediction, dis-augmentation and merging. We first apply augmentations on the test image. These are the same as the augmentations previously applied on the training dataset. We predict on both the original and the augmented images, then we revert the transformation on the obtained predictions; this process is referred to as dis-augmentation. For example, when the prediction was performed on a flipped or rotated image, we restore the obtained prediction to its original orientation. The final merging step is not straightforward in case of Mask R-CNN, as the architecture is instance aware, thus the merging method has to handle instances. We have developed an extended merging method inspired by one of the DSB 2018 solutions[24] (Fig. 1, right). For each detected object from the original image, we find the same detected objects in the augmented images by calculating intersection over union (IoU) between the masks. The minimum IoU threshold used to decide whether the objects found are the same is 0.5. We iterate over all detected objects to find the best match. An object should be present in the majority of the images to be included as a final mask. Next, we check the first augmented image for any remaining unused objects (a possible scenario when an object is not detected in the original image but is detected in any of the augmented ones), and look for matching unassigned objects on other augmentations. Next, we check the second augmented image for detected objects, and perform the same operations. We repeat this process until the majority voting criterion can be theoretically satisfied (in half of the images at a maximum). An average binary object mask is created by majority pixel voting on paired objects.

For U-Net the merging process is straightforward as it is not instance aware, so we simply sum and average all the dis-augmented probability maps. It yields a floating point image that needs to be converted to a binary mask. A simple element-wise thresholding at the value of 0.5 converts the soft masks into binary masks (Fig. 1, right).

Test-time augmentation evaluation.

We have evaluated the test-time augmentation model on our test dataset predictions (see the previous section for details) compared to ground truth masks using the following evaluation strategies.

In case of Mask R-CNN we used the same metric as at the Data Science Bowl 2018. It calculates the mean average precision (mAP) at different intersection over union (IoU) thresholds. The thresholds (t) are in the range of [0.5, 0.95] with a step of 0.05. An object is considered true positive when the IoU with ground truth is greater than the threshold, false positive when the predicted object has no associated ground truth object or the overlap is smaller than the threshold, and false negative when the ground truth object has no associated predicted object.

$$IoU(A,\ B) = \frac{|A \cap B|}{|A \cup B|}$$

Thus, mAP for an image is calculated as follows:

$$\frac{1}{|thresholds|} = \sum_t \frac{TP(t)}{TP(t) + FP(t) + FN(t)}$$

Next, we calculate the average for all images in the test set. The final score is a value between 0 and 1.

U-Net predictions were evaluated using the intersection over union metric, executed at the pixel level. We summed up the prediction and ground truth binary masks, then we simply counted the pixels that are greater than one (i.e. the intersection), and divided the resulting values with the number of pixels greater than zero. The resulting value is a score ranging from 0 to 1.

As described above, we have evaluated the predictions with applying TTA (*merged*) and without applying TTA (*original*). Next, we have evaluated TTA's performance by calculating the difference as $delta = merged - original$.

Results

We have evaluated the performance of TTA on two datasets, named 'Fluorescent' (fluorescent microscopy images) and 'Tissue' (histopathology images) datasets, described in the "Dataset acquisition and description" section in detail. Each of them was split in 3 different ways to have approximately 5% (one holdout set), 15% (cross-validation, 6 splits for each) and 30% (one holdout set) as a test set. By using such versatile data collected from different sources and representing a wide variety of experimental conditions, as well as by the test set splits, we aimed to present the truly general performance of TTA, and demonstrate how robustly it works. Regarding that most of these images were used in a Data Science competition, and some additional images came from other sources, our final datasets are similar to real-world scenarios.

Our choice of the two popular deep learning architectures, Mask R-CNN (yielding instances) and U-Net (semantic segmentation) also served the purpose of testing robustness, as the tasks of semantic and instance segmentation are different, and require different approaches to apply the same method to them. For each dataset/split, we have trained separate U-Net and Mask R-CNN models. Then, we have evaluated the performance of TTA for each model's checkpoint (checkpoints were made for each epoch of training: in case of U-Net, a total of '15' sets, i.e. every 10th epoch was designated as a checkpoint for cross-validation splits 2–6) as described in the "Test-time augmentation evaluation" subsection. Next, we performed statistical tests to assess whether the improvement of the performance is significant.

In the case of Mask R-CNN, TTA on average has provided an improved performance for all dataset splits and for all model checkpoints. The average mAP score *delta* is about 0.01 for all "Fluorescent" and "Tissue_5" sets and 0.02 for the other sets. In all scenarios, TTA has improved the score for most of the images (see Fig. 3 and Supplementary Fig. 1 for cross-validation splits 2–6). Such a *delta* value usually corresponds for better segmentation borders and a reduced rate of false positive or/and false negative detections.

In the case of U-Net, we have evaluated the performance at each epoch during training. For the "Tissue" dataset TTA has demonstrated a performance gain for all epochs. In case of the "Fluorescent" dataset, a slight decline in the performance of TTA was observed during early (first 30–50) epochs, which has turned positive after further training (Fig. 4A,B). After about epoch 50, the performance without TTA was seen to fluctuate without a clear trend in all cases (Fig. 4C,D), while the performance with TTA tended to rise for almost all cases, except in the case of the "Tissue" dataset, where no augmentations were used for training (Fig. 4A). A slight decline or a slight improvement in the score is usually related to cell borders (as the most uncertain regions in the images). In some cases, TTA helps to eliminate artifacts and rarely occurring false positive/false negative objects.

For some images TTA has significantly improved the final prediction. Examples of such cases for both U-Net and Mask R-CNN are shown in Fig. 2.

We have performed Wilcoxon paired test for each dataset/split/checkpoint for the Mask R-CNN results. P-values in all cases have passed the threshold value of 0.05. For U-Net, the test was performed on the means of each 10th epoch (20 vs 20 data points) for each dataset/split. The P-values are shown in Supplementary Table 4.

Applying TTA on the DSB2018 (stage2) test set of images has improved performance significantly, surpassing the best performing method[5] by 0.011 in the DSB scoring metric, which is identical to the mAP used in this paper and the output of which was a set of instance segmented masks (Fig. 5). In the context of data science competitions, when the scores are rather dense, we consider this improvement as significant (difference between 2nd and 1st place on DSB 2018 was only 0.017).

The results without TTA and *delta* values for each set are available as Supplementary Materials (Supplementary Table 1. U-Net when augmentations during training were used, Supplementary Table 2. U-Net when augmentations during training were not used, Supplementary Table 3. Mask R-CNN when augmentations during training were used).

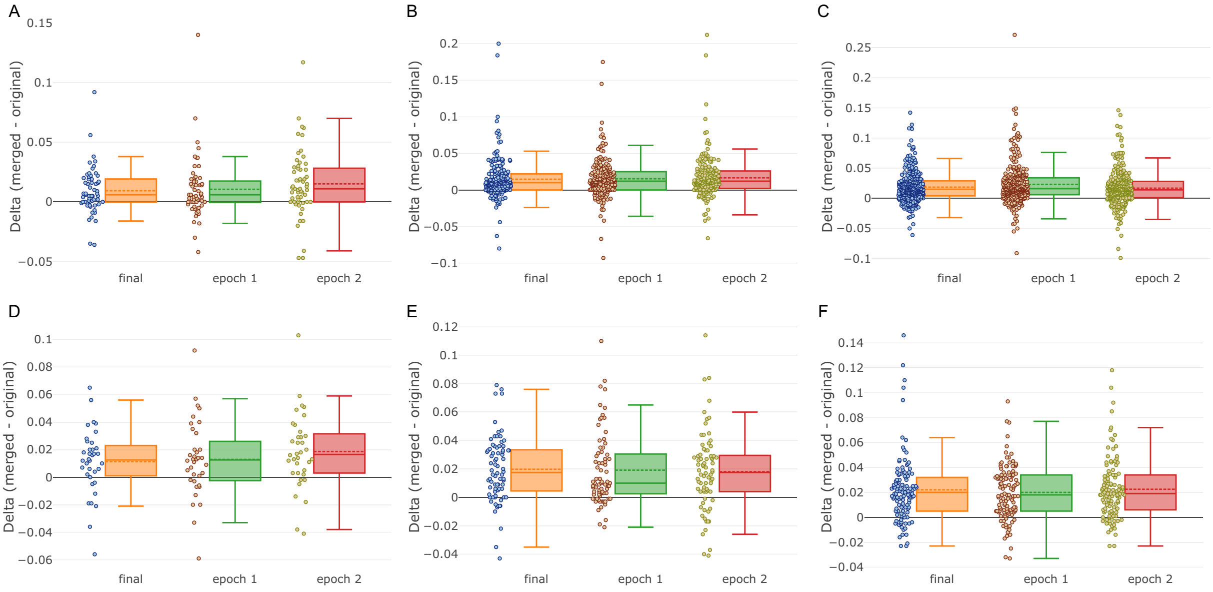


Figure 3. TTA performance for Mask R-CNN. TTA performance (*delta* = *merged* − *original*). Each point represents an image. Dashed line - mean, solid line - median. (**A**) Fluorescent set 5. (**B**) Fluorescent set 15 (cross-validation split 1). (**C**) Fluorescent set 30. (**D**) Tissue set 5. (**E**) Tissue set 15 (cross-validation split 1). (**F**) Tissue set 30. Orange boxplot - the final model (epoch 3), green boxplot - model trained for 1 epoch, red boxplot - model trained for 2 epochs.

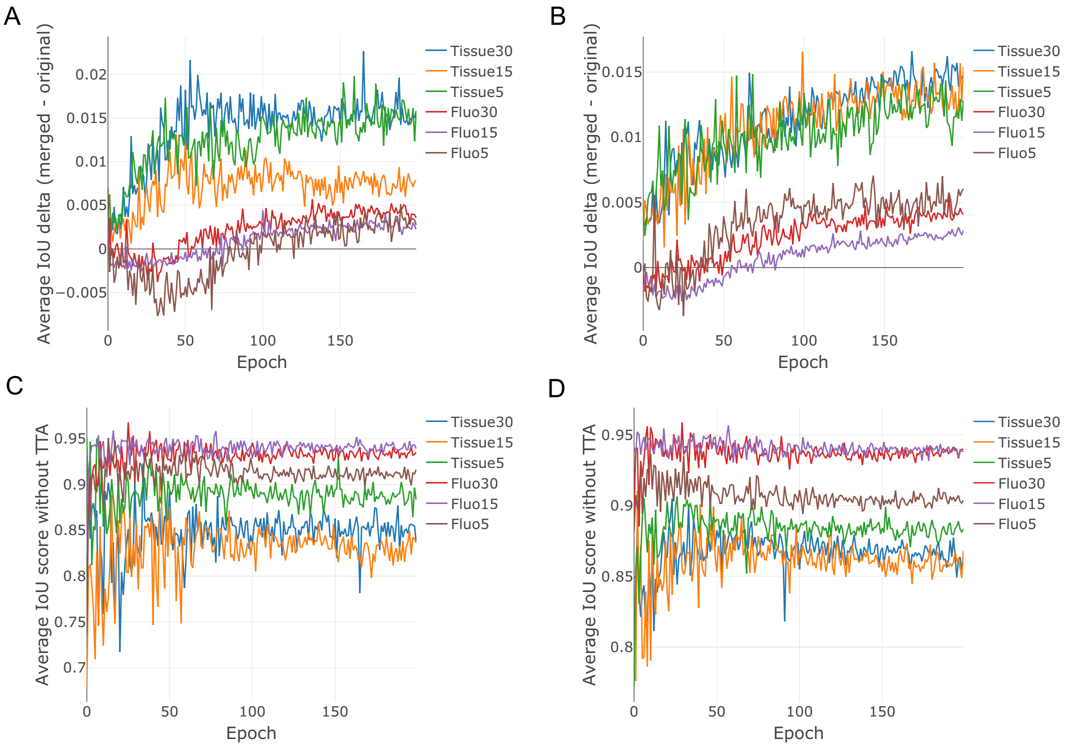


Figure 4. Average performance of TTA for U-Net with different training and test augmentations. (**A**) Average TTA performance trained without augmentations over epochs. (**B**) Average TTA performance trained with augmentations over epochs. (**C**) Average performance without TTA without augmentations during training. (**D**) Average performance without TTA with augmentations during training. Tissue15 and Fluorescent15 stand for the first cross-validation split.

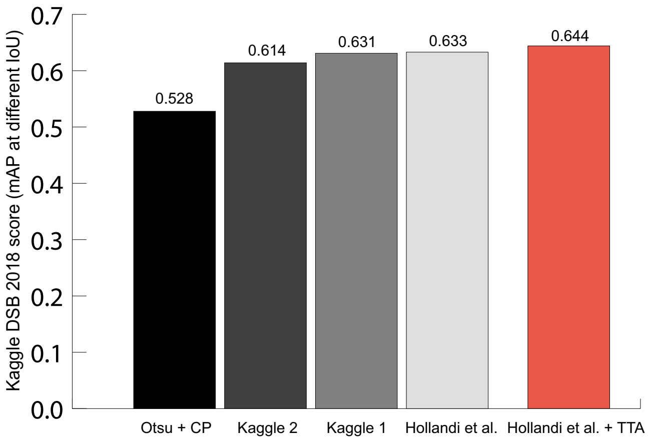


Figure 5. DSB Stage 2 scores for various methods (CellProfiler, Kaggle DSB 2018 2nd and 1st places, Hollandi *et al.*[5] method and the same method with TTA). The red bar shows the highest score.

Conclusions

We have performed experiments to estimate test-time augmentation's performance for two popular deep learning frameworks trained to segment nuclei in microscopy images. Our results indicate that on average TTA can provide higher segmentation accuracy compared to predicting based on the original images only, even though for some images the results might be marginally worse.

TTA mostly affects the objects' borders, but in uncertain cases it can help to fit whole objects (remove false positives or add true positives, especially in case of Mask R-CNN). Overall, in most cases, TTA improves segmentation accuracy. The main use case of TTA is the analysis of uncertain regions in segmentation. However, the high cost of TTA, related to the fact that multiple times more predictions are required for the same object, is also an issue to be considered. Therefore, TTA is mainly recommended for use when the basic cost of prediction is low.

Received: 31 October 2019; Accepted: 20 February 2020;
Published online: 19 March 2020

References

1. Carpenter, A. E. *et al.* CellProfiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biol.* **7**, R100 (2006).
2. Molnar, C. *et al.* Accurate Morphology Preserving Segmentation of Overlapping Cells based on Active Contours. *Sci. Rep.* **6**, 32412 (2016).
3. Molnar, J., Molnar, C. & Horvath, P. An Object Splitting Model Using Higher-Order Active Contours for Single-Cell Segmentation. *Advances in Visual Computing* 24–34, https://doi.org/10.1007/978-3-319-50835-1_3 (2016).
4. Ronneberger, O., Fischer, P. & Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *Lecture Notes in Computer Science* 234–241, https://doi.org/10.1007/978-3-319-24574-4_28 (2015).
5. Hollandi, R. *et al.* A deep learning framework for nucleus segmentation using image style transfer. *bioRxiv* 580605, https://doi.org/10.1101/580605 (2019).
6. Dobos, O., Horvath, P., Nagy, F., Danka, T. & Viczián, A. A deep learning-based approach for high-throughput hypocotyl phenotyping. *bioRxiv* 651729, https://doi.org/10.1101/651729 (2019).
7. Caicedo, J. *et al.* Nucleus segmentation across imaging experiments: the 2018 Data Science Bowl. *Nature Methods*, https://doi.org/10.1038/s41592-019-0612-7 (2019).
8. He, K., Gkioxari, G., Dollar, P. & Girshick, R. Mask R-CNN. *IEEE Trans. Pattern Anal. Mach. Intell.*, https://doi.org/10.1109/TPAMI.2018.2844175 (2018).
9. Krizhevsky, A., Sutskever, I. & Hinton, G. E. ImageNet classification with deep convolutional neural networks. *Communications of the ACM* **60**, 84–90 (2017).
10. Matsunaga, K., Hamada, A., Minagawa, A. & Koga, H. *Image Classification of Melanoma, Nevus and Seborrheic Keratosis by Deep Neural Network Ensemble*. (2017).
11. Ayhan, M. S. & Berens, P. *Test-time Data Augmentation for Estimation of Heteroscedastic Aleatoric Uncertainty in Deep Neural Networks*. (2018).
12. Wang, G. *et al.* Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks. *Neurocomputing* **338**, 34–45 (2019).
13. Ayhan, M. S. *et al.* Expert-validated estimation of diagnostic uncertainty for deep neural networks in diabetic retinopathy detection. *medRxiv* 19002154 (2019).
14. Brasko, C. *et al.* Intelligent image-based *in situ* single-cell isolation. *Nature Communications* **9** (2018).
15. Caicedo, J. C. *et al.* Evaluation of Deep Learning Strategies for Nucleus Segmentation in Fluorescence Images., https://doi.org/10.1101/335216.
16. Caie, P. D. *et al.* High-content phenotypic profiling of drug response signatures across distinct cancer cells. *Mol. Cancer Ther.* **9**, 1913–1926 (2010).
17. Coelho, L. P., Shariff, A. & Murphy, R. F. Nuclear segmentation in microscope cell images: A hand-segmented dataset and comparison of algorithms. *2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, https://doi.org/10.1109/isbi.2009.5193098 (2009).
18. Smith, K. *et al.* CIDRE: an illumination-correction method for optical microscopy. *Nature Methods* **12**, 404–406 (2015).
19. Naylor, P., Lae, M., Reyal, F. & Walter, T. Segmentation of Nuclei in Histopathology Images by Deep Regression of the Distance Map. *IEEE Transactions on Medical Imaging* **38**, 448–459 (2019).

20. Kumar, N. *et al.* A Dataset and a Technique for Generalized Nuclear Segmentation for Computational Pathology. *IEEE Trans. Med. Imaging* **36**, 1550–1560 (2017).
21. Matterport. matterport/Mask_RCNN. *GitHub* Available at, https://github.com/matterport/Mask_RCNN. (Accessed: 7th October 2019).
22. Kingma, D. P. & Ba, J. *Adam: A Method for Stochastic Optimization* (2014).
23. Zhixuhao. zhixuhao/unet. *GitHub* Available at, https://github.com/zhixuhao/unet. (Accessed: 7th October 2019).
24. Mirzaevinom. mirzaevinom/data_science_bowl_2018. *GitHub* Available at, https://github.com/mirzaevinom/data_science_bowl_2018. (Accessed: 7th October 2019).

Acknowledgements

N.M., R.H., B.M., and P.H. acknowledge support from the LENDULET-BIOMAG Grant (2018–342) and from the European Regional Development Funds (GINOP-2.3.2-15-2016-00006, GINOP-2.3.2-15-2016-00037), N.M. is supported by the Doctoral School of Interdisciplinary Medicine of the University of Szeged and the Doctoral School of Computer Science at the National Research University, Higher School of Economics. The authors acknowledge an NVIDIA grant for TitanXp GPUs. The authors thank Dora Bokor, PharmD, for proofreading the manuscript.

Author contributions

N.M., B.M., R.H., A.K.-F. and P.H. wrote the manuscript. N.M., B.M. and R.H. performed the experiments. R.H. collected the dataset. R.H. and N.M. prepared the figures for the paper.

Competing interests

The authors declare no competing interests.

Additional information

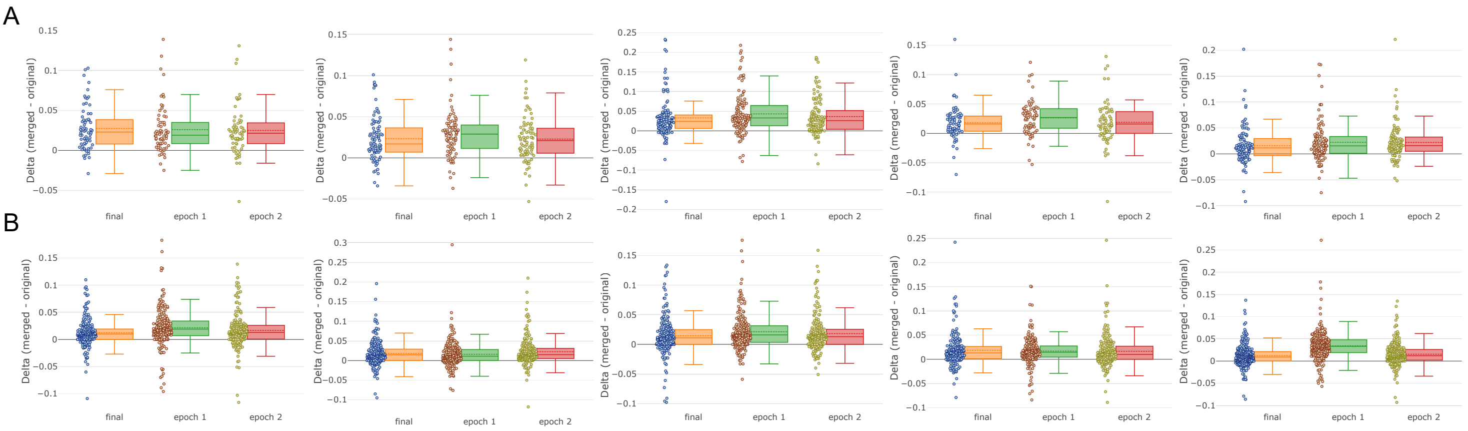
Supplementary information is available for this paper at https://doi.org/10.1038/s41598-020-61808-3.

Correspondence and requests for materials should be addressed to P.H.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Nikita Moshkov, Botond Mathe, Attila Kertesz-Farkas, Reka Hollandi, Peter Horvath
Test-time augmentation for deep learning-based cell segmentation on microscopy images



Supplementary Figure 1. TTA performance for Mask R-CNN
TTA performance (delta = merged-original). Each point represents an image. Dashed line - mean, solid line - median. A | Tissue 15 set cross-validation folds 2-6 (left to right) B | Fluorescent 15 set cross-validation folds 2-6 (left to right) Orange boxplot - the final model (epoch 3), green boxplot - model trained for 1 epoch, red boxplot - model trained for 2 epochs.

C Article ‘AnnotatorJ: an ImageJ plugin to ease hand annotation of cellular compartments‘

AnnotatorJ: an ImageJ plugin to ease hand annotation of cellular compartments

Réka Hollandi[a], Ákos Diósdi[a,b], Gábor Hollandi[a], Nikita Moshkov[a,c,d], and Péter Horváth[a,e,*]
[a]Synthetic and Systems Biology Unit, Biological Research Center, 6726 Szeged, Hungary; [b]Doctoral School of Biology, University of Szeged, 6726 Szeged, Hungary; [c]Doctoral School of Interdisciplinary Medicine, University of Szeged, Koranyi fasor 10, 6720 Szeged, Hungary; [d]National Research University Higher School of Economics, Faculty of Computer Science, 101000 Moscow, Russia; [e]Institute for Molecular Medicine Finland, University of Helsinki, 00014 Helsinki, Finland

ABSTRACT **AnnotatorJ combines single-cell identification with deep learning (DL) and manual annotation. Cellular analysis quality depends on accurate and reliable detection and segmentation of cells so that the subsequent steps of analyses, for example, expression measurements, may be carried out precisely and without bias. DL has recently become a popular way of segmenting cells, performing unimaginably better than conventional methods. However, such DL applications may be trained on a large amount of annotated data to be able to match the highest expectations. High-quality annotations are unfortunately expensive as they require field experts to create them, and often cannot be shared outside the lab due to medical regulations. We propose AnnotatorJ, an ImageJ plugin for the semiautomatic annotation of cells (or generally, objects of interest) on (not only) microscopy images in 2D that helps find the true contour of individual objects by applying U-Net–based presegmentation. The manual labor of hand annotating cells can be significantly accelerated by using our tool. Thus, it enables users to create such datasets that could potentially increase the accuracy of state-of-the-art solutions, DL or otherwise, when used as training data.**

Monitoring Editor
Jennifer Lippincott-Schwartz
Howard Hughes Medical Institute

Received: Feb 27, 2020
Revised: Jun 24, 2020
Accepted: Jul 17, 2020

INTRODUCTION

Single-cell analysis pipelines begin with an accurate detection of the cells. Even though microscopy analysis software tools aim to become more and more robust to various experimental setups and imaging conditions, most lack efficiency in complex scenarios such as label-free samples or unforeseen imaging conditions (e.g., higher signal-to-noise ratio, novel microscopy, or staining techniques), which opens up a new expectation of such software tools: adaptation ability (Hollandi *et al.*, 2020). Another crucial requirement is to maintain ease of usage and limit the number of parameters the users need to fine-tune to match their exact data domain.

Recently, deep learning (DL) methods have proven themselves worthy of consideration in microscopy image analysis tools as they have also been successfully applied in a wider range of applications including but not limited to face detection (Sun *et al.*, 2014; Taigman *et al.*, 2014; Schroff *et al.*, 2015), self-driving cars (Redmon *et al.*, 2016; Badrinarayanan *et al.*, 2017, Grigorescu *et al.*, 2019), and speech recognition (Hinton *et al.*, 2012). Caicedo *et al.* (Caicedo *et al.*, 2019) and others (Hollandi *et al.*, 2020; Moshkov *et al.*, 2020) proved that single-cell detection and segmentation accuracy can be significantly improved utilizing DL networks. The most popular and widely used deep convolutional neural networks (DCNNs) include Mask R-CNN (He *et al.*, 2017): an object detection and instance segmentation network; YOLO (Redmon *et al.*, 2016; Redmon and Farhadi, 2018): a fast object detector; and U-Net (Ronneberger *et al.*, 2015): a fully convolutional network specifically intended for bioimage analysis purposes and mostly used for pixel classification. StarDist (Schmidt *et al.*, 2018) is an instance segmentation DCNN optimal for convex or elliptical shapes (such as nuclei).

As robustly and accurately as they may perform, these networks rely on sufficient data, both in amount and quality, which tends to be the bottleneck of their applicability in certain cases such as single-cell detection. While in more industrial applications (see Grigorescu *et al.*, 2019 for an overview of autonomous driving) a large amount of training data can be collected relatively easily (see the cityscapes dataset [Cordts *et al.*, 2016; available at https://www.cityscapes-dataset.com/] of traffic video frames using a car and camera to

This article was published online ahead of print in MBoC in Press (http://www.molbiolcell.org/cgi/doi/10.1091/mbc.E20-02-0156). on July 22, 2020

*Address correspondence to: Péter Horváth (horvath.peter@brc.hu).

Abbreviations used: DL, deep learning; DL4J, Deeplearning4j; IoU, intersection over union; ROI, region of interest.

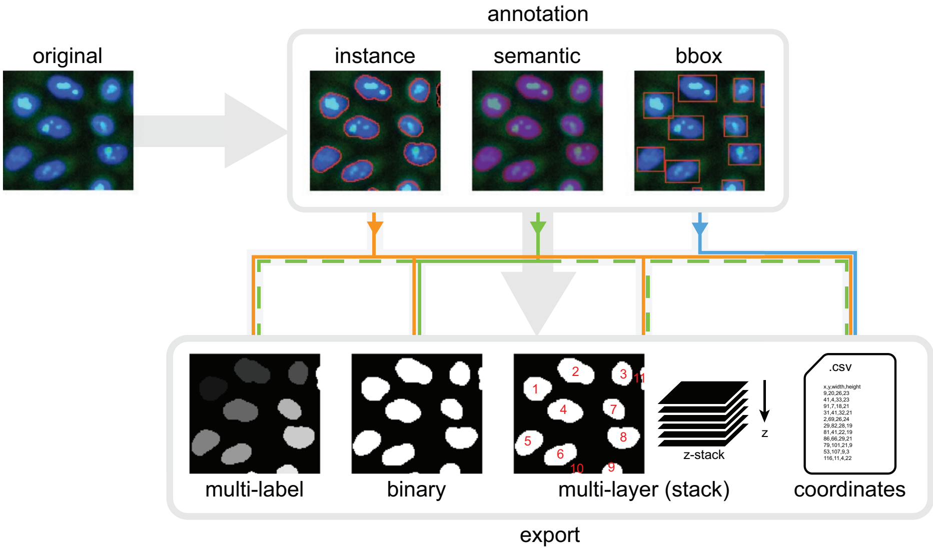


FIGURE 1: Annotation types. The top row displays our supported types of annotation: instance, semantic, and bounding box (noted as "bbox" in the figure) based on the same objects of interest, in this case nuclei, shown in red. Instances mark the object contours, semantic overlay shows the regions (area) covered, while bounding boxes are the smallest enclosing rectangles around the object borders. Export options are shown in the bottom row: multilabel, binary, multilayer images, and coordinates in a text file. Lines mark the supported export options for each annotation type by colors: orange for instance, green for semantic, and blue for bounding box. Dashed lines indicate additional export options for semantics that should be used carefully.

record and potentially nonexpert individuals to label the objects), clinical data is considerably more difficult, due to ethical constraints, and expensive to gather as expert annotation is required. Datasets available in the public domain such as BBBC (Ljosa *et al.*, 2012) at https://data.broadinstitute.org/bbbc/, TNBC (Naylor *et al.*, 2017, 2019) or TCGA (Cancer Genome Atlas Research Network, 2008; Kumar *et al.*, 2017), and detection challenges including ISBI (Coelho *et al.*, 2009), Kaggle (https://www.kaggle.com/, e.g., Data Science Bowl 2018; see at https://www.kaggle.com/c/data-science-bowl-2018), ImageNet (Russakovsky *et al.*, 2015), etc., contribute to the development of genuinely useful DL methods; however, most of them lack heterogeneity of the covered domains and are limited in data size. Even combining them one could not possibly prepare their network/method to generalize well (enough) on unseen domains that vastly differ from the pool they covered. On the contrary, such an adaptation ability can be achieved if the target domain is represented in the training data, as proposed in Hollandi *et al.*, 2020, where synthetic training examples are generated automatically in the target domain via image style transfer.

Eventually, similar DL approaches' performance can only be increased over a certain level if we provide more training examples. The proposed software tool was created for this purpose: the expert can more quickly and easily create a new annotated dataset in their desired domain and feed the examples to DL methods with ease. The user-friendly functions included in the plugin help organize data and support annotation, for example, multiple annotation types, editing, classes, etc. Additionally, a batch exporter is provided offering different export formats matching typical DL models'; supported annotation and export types are visualized in Figure 1; open-source code is available at https://github.com/spreka/annotatorj under GNU GPLv3 license.

We implemented the tool as an ImageJ (Abramoff *et al.*, 2004, Schneider *et al.*, 2012) plugin because ImageJ is frequently used by bioimage analysts, providing a familiar environment for users. While other software also provide a means to support annotation, for example, by machine learning–based pixel classification (see a detailed comparison in *Materials and Methods*), AnnotatorJ is a lightweight, free, open-source, cross-platform alternative. It can be easily installed via its ImageJ update site at https://sites.imagej.net/Spreka/ or run as a standalone ImageJ instance containing the plugin.

In AnnotatorJ we initialize annotations with DL presegmentation using U-Net to suggest contours from as little as a quickly drawn line over the object (see Supplemental Material and Figure 2). U-Net predicts pixels belonging to the target class with the highest probability within a small bounding box (a rectangle) around the initially drawn contour; then a fine approximation of the true object boundary is calculated from connected pixels; this is referred to as the suggested contour. The user then manually refines the contour to create a pixel-perfect annotation of the object.

RESULTS AND DISCUSSION

Performance evaluation

We quantitatively evaluated annotation performance and speed in AnnotatorJ (see Figures 3 and 4) with the help of three annotators who had experience in cellular compartment annotation. Both annotation accuracy and time were measured on the same two test sets: a nucleus and a cytoplasm image set (see also Supplemental Figure S1 and Supplemental Material). Both test sets contained images of various experimental conditions, including fluorescently labeled and brightfield-stained samples, tissue section, and cell culture images. We compared the effectiveness of our plugin using *Contour assist* mode to only allowing the use of *Automatic adding*. Even though the latter is also a functionality of AnnotatorJ, it ensured that the measured annotation times correspond to a single object each. Without this option the user must press the key "t" after every contour drawn to add it to the region of interest (ROI) list, which can be unintendedly missed, increasing its time as the same contour must be drawn again.

For the annotation time test presented in Figure 3 we measured the time passed between adding new objects to the annotated object set in ROI Manager for each object, then averaged the times for each image and each annotator, respectively. Time was measured in the Java implementation of the plugin in milliseconds. Figures 3 and 4 show SEM error bars for each mean measurement (see Supplemental Material for details).

In the case of annotating cell nuclei, results confirm that hand-annotation tasks can be significantly accelerated using our tool. Each of the three annotators were faster by using *Contour assist*; two of them nearly double their speed.

To ensure efficient usage of our plugin in annotation assistance, we also evaluated the accuracies achieved in each test case by calculating mean intersection over union (IoU) scores of the annotations as segmentations compared with ground truth masks previously created by different expert annotators. We used the mean IoU score defined in the Data Science Bowl 2018 competition (https://www.kaggle.com/c/data-science-bowl-2018/overview/evaluation) and in Hollandi *et al.*, 2020:

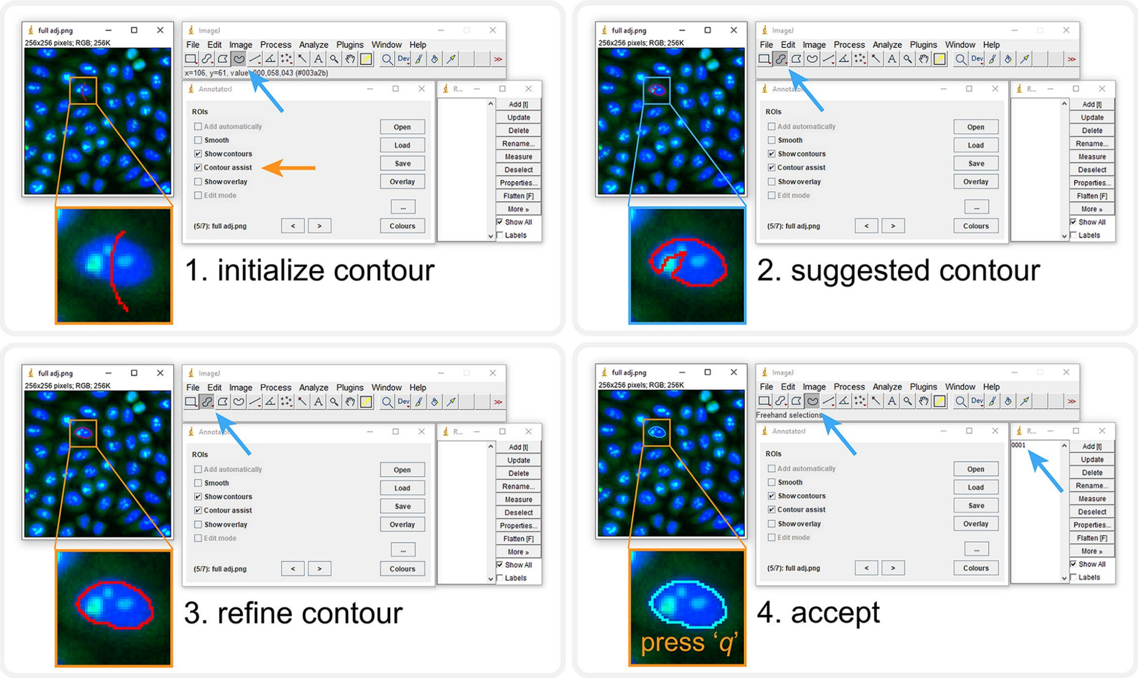


FIGURE 2: Contour assist mode of AnnotatorJ. The blocks show the order of steps; the given tool needed is automatically selected. User interactions are marked with orange arrows, and automatic steps with blue. 1) Initialize the contour with a lazily drawn line; 2) the suggested contour appears (a window is shown until processing completes), brush selection tool is selected automatically; 3) refine the contour as needed; 4) accept it by pressing the key "q" or reject with "Ctrl" + "delete." Accepting adds the ROI to ROI Manager with a numbered label. See also Supplemental Material for a demo video (figure2.mov).

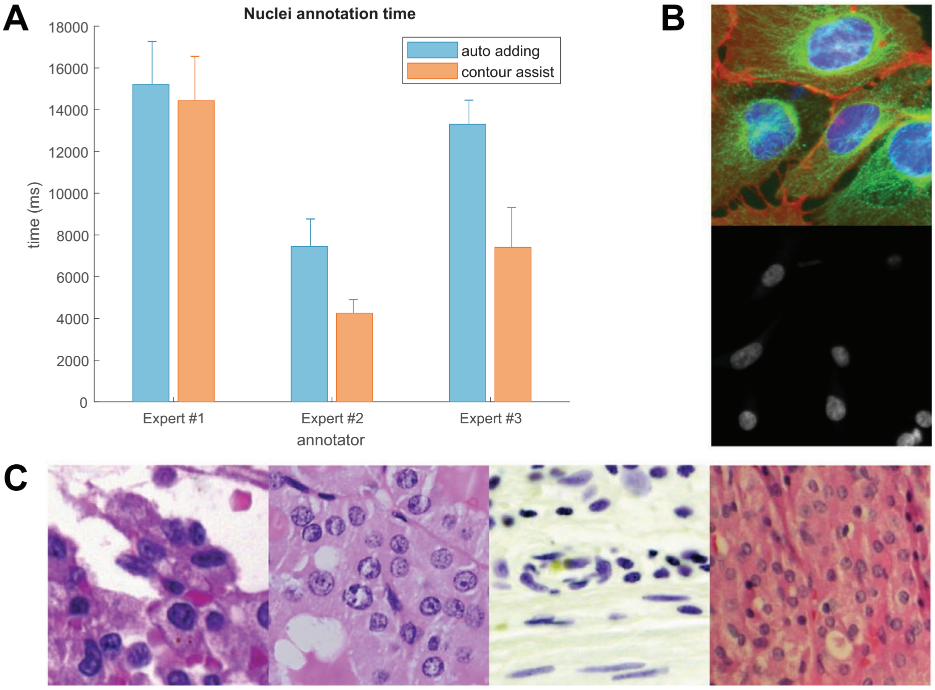


FIGURE 3: Annotation times on nucleus images. AnnotatorJ was tested on sample microscopy images (both fluorescent and brightfield, as well as cell culture and tissue section images); annotation time was measured on a per-object (nucleus) level. Bars represent the mean annotation times on the test image set; error bars show SEM. Orange corresponds to *Contour assist* mode and blue to only allowing the *Automatic adding* option. (A) Nucleus test set annotation times. (B) Example cell culture test images. (C) Example histopathology images. Images shown in B and C are 256 × 256 crops of original images. Some images are courtesy of Kerstin Elisabeth Dörner, Andreas Mund, Viktor Honti, and Hella Bolck.

$$\mathrm{IoU\,score}(t)=\frac{\mathrm{TP}(t)}{\mathrm{TP}(t)+\mathrm{FP}(t)+\mathrm{FN}(t)+\varepsilon} \qquad (1)$$

IoU determines the overlapping pixels of the segmented mask with the ground truth mask (intersection) compared with their union. The IoU score is calculated at 10 different thresholds from 0.5 to 0.95 with 0.05 steps; at each threshold true positive (TP), false positive (FP), and false negative (FN) objects are counted. An object is considered TP if its IoU is greater than the given threshold *t*. IoU scores calculated at all 10 thresholds were finally averaged to yield a single IoU score for a given image in the test set.

An arbitrarily small $\varepsilon = 10^{-40}$ value was added to the denominators for numerical stability. Equation 1 is a modified version of mean average precision (mAP) typically used to describe the accuracy of instance segmentation approaches. Precision is formulated as

$$\mathrm{precision}(t)=\frac{\mathrm{TP}(t)}{\mathrm{TP}(t)+\mathrm{FP}(t)+\varepsilon} \qquad (2)$$

Nucleus and cytoplasm image segmentation accuracies were averaged over the test sets, respectively. We compared our

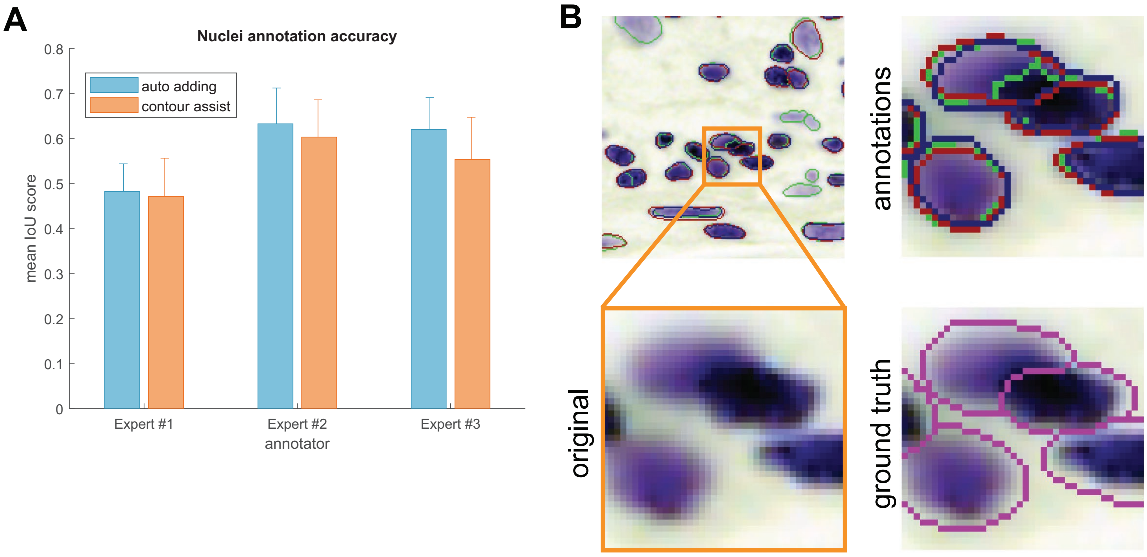


FIGURE 4: Annotation accuracies. Annotations created in the same test as the times measured in Figure 3 were evaluated using mean IoU scores for the nucleus test set. Error bars show SEM; colors are as in Figure 3. (A) Nucleus test set accuracies. (B) Example contours drawn by our expert annotators. Inset highlighted in orange is zoomed in showing the original image; annotations are marked in red, green, and blue corresponding to experts #1–3, respectively, and ground truth contours (in magenta) are overlayed on the original image for comparison.

annotators using and not using *Contour assist* mode (Figure 4). The results show greater interexpert than intraexpert differences, allowing us to conclude that the annotations created in AnnotatorJ are nearly as accurate as freehand annotations.

Export evaluation

As the training data annotation process for deep learning applications requires the annotated objects to be exported in a manner that DL models can load them, which typically covers the four types of export options offered in AnnotatorJExporter, it is also important to investigate the efficiency of export. We measured export times similarly to annotation times. For the baseline results, each object defined by their ROI was copied to a new empty image, then filled and saved to create a segmentation mask image file. Exportation from AnnotatorJExporter was significantly faster and only required a few clicks: it took four orders of magnitude less time to export the annotations (~60 ms). Export times reported correspond to a randomly selected expert so that computer hardware specifications remain the same.

Comparison to other tools and software packages

The desire to collect annotated datasets has arisen with the growing popularity and availability of application-specific DL methods. Object classification on natural images (photos) and face recognition are frequently used examples of such applications in computer vision. We discuss some of the available software tools created for image annotation tasks and compare their feature scope in the following table (Table 1; see also Supplemental Table S1) and in the Supplemental Material.

We collected our list of methods to compare following Morikawa, 2019 and "The best image annotation platforms", 2018. While there certainly is a considerable amount of annotation tools for object detection purposes, most of them are not open source. We included Lionbridge.AI (https://lionbridge.ai/services/image-annotation/) and Hive (https://thehive.ai/), two service-based solutions, because of their wide functionality and artificial intelligence support. Both of them work in a project-management way and outsource the annotation task to enable fast and accurate results. Their main application spectra cover more general object detection tasks like classification of traffic video frames. LabelImg (https://github.com/tzutalin/labelImg), on the other hand, as well as the following tools, is open source but offers a narrower range of annotation options and lacks machine learning support making it a lightweight but free alternative. VGG Image Annotator (Dutta and Zisserman, 2019) comes on a web-based platform, therefore making it very easy for the user to become familiarized with the software. It enables multiple types of annotation with class definition. Diffgram (https://diffgram.com/) is available both online and as a locally installable version (Python) and adds DL support which speeds up the annotation process significantly; that is, provided the intended object classes are already trained and the DL predictions only need minor edit. A similar, also web-based approach is provided by supervise.ly (https://supervise.ly/; see the Supplemental Material), which is free for research purposes. Even though web-hosted services offer a convenient solution for training new models (if supported), handling sensitive clinical data may be problematic. Hence, locally installable software is more desirable in biological and medical applications. A software closer to the bioimage analyst community is CytoMine (Marée *et al.*, 2016; Rubens *et al.*, 2019), a more general image processing tool with a lot of annotation options that also provides DL support and has a web interface. SlideRunner (Aubreville *et al.*, 2018) was created for large tissue section (slide) annotation specifically, but similar to others it does not integrate machine learning methods to help annotation and rather focuses on the classification task.

AnnotatorJ, on the other hand, as an ImageJ (Fiji) plugin should provide a familiar environment for bioimage annotators to work in. It offers all the functionality available in similar tools (such as different annotation options: bounding box, polygon, freehand drawing, semantic segmentation, and editing them) while it also incorporates support for a popular DL model, U-Net. Furthermore, any user-trained Keras model can be loaded into the plugin with ease because of the DL4J framework, extending its use cases to general object annotation tasks (see Supplemental Figure S2 and Supplemental Material). Due to its open-source implementation, the users can modify or extend the plugin to even better fit their needs.

TABLE 1: Comparison of annotation software tools.

Feature		Tool							
		LabelImg	Lionbridge.AI	Hive	VGG Image Annotator	Diffgram	CytoMine	SlideRunner	AnnotatorJ
Open source		✓	×	×	✓	✓	✓	✓	✓
Cross-platform		✓	service	service	✓	✓	Ubuntu	✓	✓
Implementation		Python	N/A	N/A	web	Python, web	Docker, web	Python	Java
Annotation	Bounding box	✓	✓	✓	✓	✓	✓	✓	✓
	Freehand ROI	×	×	N/A	×	×	✓	×	✓
	Polygonal region	×	✓	✓	✓	✓	✓	✓	✓ (ImageJ)
	Semantic	×	✓	✓	×	×	✓	×	✓
	Single click[a]	×	×	×	×	×	✓ (magic wand)	✓	✓ (drag)
	Edit selection	× (drag)	N/A	N/A	✓	✓	✓	N/A	✓
Class option		✓	✓	✓	✓	✓	✓	✓	✓
DL	Support	×	AI assist	N/A	×	✓	✓	×	✓
	Model import	×	N/A	N/A	×	Tensorflow	N/A	×	Keras, DL4J

[a] Bounding box drawing with a single click and drag is not considered a single click annotation.

Additionally, as an ImageJ plugin it requires no software installation, can be downloaded inside ImageJ/Fiji (via its update site, https://sites.imagej.net/Spreka/), or run as a standalone ImageJ instance with this plugin.

We also briefly discuss ilastik (Sommer *et al.*, 2011; Berg *et al.*, 2019) and Suite2p (Pachitariu *et al.*, 2017) in the Supplemental Material because they are not primarily intended for annotation purposes. Two ImageJ plugins that offer manual annotation and machine learning–generated outputs, Trainable Weka Segmentation (Arganda-Carreras *et al.*, 2017) and LabKit (Arzt, 2017), are also detailed in the Supplemental Material.

We presented an ImageJ plugin, AnnotatorJ, for convenient and fast annotation and labeling of objects on digital images. Multiple export options are also offered in the plugin.

We tested the efficiency of our plugin with three experts on two test sets comprising nucleus and cytoplasm images. We found that our plugin accelerates the hand-annotation process on average and offers up to four orders of magnitude faster export. By integrating the DL4J Java framework for U-Net contour suggestion in *Contour assist* mode any class of object can be annotated easily: the users can load their own custom models for the target class.

MATERIALS AND METHODS

Motivation

We propose AnnotatorJ, an ImageJ (Abramoff *et al.*, 2004, Schneider *et al.*, 2012) plugin for the annotation and export of cellular compartments that can be used to boost DL models' performance. The plugin is mainly intended for bioimage annotation but could possibly be used to annotate any type of object on images (see Supplemental Figure S2 for a general example). During development we kept in mind that the intended user should be able to get comfortable with the software very quickly and quicken the otherwise truly time-consuming and exhausting process of manually annotating single cells or their compartments (such as individual nucleoli, lipid droplets, nucleus, or cytoplasm).

The performance of DL segmentation methods is significantly influenced by both the training data size and its quality. Should we feed automatically segmented objects to the network, errors present in the original data will be propagated through the network during training and bias the performance, hence such training data should always be avoided. Hand-annotated and curated data, however, will minimize the initial error boosting the expected performance increase on the target domain to which the annotated data belongs. NucleAIzer (Hollandi *et al.*, 2020) showed an increase in nucleus segmentation accuracy when a DL model was trained on synthetic images generated from ground truth annotations instead of presegmented masks.

Features

AnnotatorJ helps organize the input and output files by automatically creating folders and matching file names to the selected type and class of annotation. Currently, the supported annotation types are 1) instance, 2) semantic, and 3) bounding box (see Figure 1). Each of these are typical inputs of DL networks; instance annotation provides individual objects separated by their boundaries (useful in the case of, e.g., clumped cells of cancerous regions) and can be used to provide training data for instance segmentation networks such as Mask R-CNN (He *et al.*, 2017). Semantic annotation means foreground–background separation of the image without distinguishing individual objects (foreground); a typical architecture using such segmentations is U-Net (Ronneberger *et al.*, 2015). And finally, bounding box annotation is done by identifying the object's bounding rectangle, and is generally used in object detection networks (like YOLO [Redmon *et al.*, 2016] or R-CNN [Girshick *et al.*, 2014]).

Semantic annotation is done by painting areas on the image overlay. All necessary tools to operate a given function of the plugin are selected automatically. Contour or overlay colors can be selected from the plugin window. For a detailed description and user guide please see the documentation of the tool (available at https://github.com/spreka/annotatorj repository).

Annotations can be saved to default or user-defined "classes" corresponding to biological phenotypes (e.g., normal or cancerous) or object classes—used as in DL terminology (such as person, chair, bicycle, etc.), and later exported in a batch by class. Phenotypic differentiation of objects can be supported by loading a previously annotated class's objects for comparison as overlay to the image and toggling their appearance by a checkbox.

We use the default ImageJ ROI Manager to handle instance annotations as individual objects. Annotated objects can be added to the ROI list automatically (without the bound keystroke "t" as defined by ROI Manager) when the user releases the mouse button used to draw the contour by checking its option in the main window of the plugin. This ensures that no annotation drawn is missing from the ROI list.

Contour editing is also possible in our plugin using "Edit mode" (by selecting its checkbox) in which the user can select any already annotated object on the image by clicking on it, then proceed to edit the contour and either apply modifications with the shortcut "Ctrl" + "q," discard them with "escape," or delete the contour with "Ctrl" + "delete." The given object selected for edit is highlighted in inverse contour color.

Object-based classification is also possible in "Class mode" (via its checkbox): similarly to "Edit mode," ROIs can be assigned to a class by clicking on them on the image which will also update the selected ROI's contour to the current class's color. New classes can be added and removed, and their class color can be changed. A default class can be set for all unassigned objects on the image. Upon export (using either the quick export button "[^]" in the main window or the exporter plugin) masks are saved by classes.

In the options (button "…" in the main window) the user can select to use either U-Net or a classical region-growing method to initialize the contour around the object marked. Currently only instance annotation can be assisted.

Contour suggestion using U-Net

Our annotation helper feature "*Contour assist*" (see Figure 2) allows the user to work on initialized object boundaries by roughly marking an object's location on the image which is converted to a well-defined object contour via weighted thresholding after a U-Net (Ronneberger *et al.*, 2015) model trained on nucleus or other compartment data predicts the region covered by the object. We refer to this as the *suggested contour* and expect the user to refine the boundaries to match the object border precisely. The suggested contour can be further optimized by applying *active contour* (AC; Kass *et al.*, 1988) to it. We aim to avoid fully automatic annotation (as previously argued) by only enabling one object suggestion at a time and requiring manual interaction to either refine, accept, or reject the suggested contour. These operations are bound to keyboard shortcuts for convenience (see Figure 2). When using the *Contour assist* function automatic adding of objects is not available to encourage the user to manually validate and correct the suggested contour as needed.

In Figure 2 we demonstrate *Contour assist* using a U-Net model trained on versatile microscopy images of nuclei in Keras and on a fluorescent microscopy image of a cell culture where the target objects, nuclei, are labeled with DAPI (in blue). This model is provided at https://github.com/spreka/annotatorj/releases/tag/v0.0.2-model in the open-source code repository of the plugin.

Contour suggestions can be efficiently used for proper initialization of object annotation, saving valuable time for the expert annotator by suggesting a nearly perfect object contour that only needs refinement (as shown in Figure 2). Using a U-Net model accurate enough for the target object class, the expert can focus on those image regions where the model is rather uncertain (e.g., around the edges of an object or the separating line between adjacent objects) and fine-tune the contour accurately while sparing considerable effort on more obvious regions (like an isolated object on simple background) by accepting the suggested contour after marginal correction.

The framework of the chosen U-Net implementation, DL4J (available at http://deeplearning4j.org/ or https://github.com/eclipse/deeplearning4j), supports Keras model import, hence custom, application-specific models can be loaded in the plugin easily by either training them in DL4J (Java) or Python (Keras) and saving the trained weights and model configuration in .h5 and .json files. This vastly extends the possible fields of application for the plugin to general object detection or segmentation tasks (see Supplemental Material and Supplemental Figures S2 and S3).

Exporter

The annotation tool is supplemented by an exporter, AnnotatorJ-Exporter plugin, also available in the package. It was optimized for the batch export of annotations created by our annotation tool. For consistency, one class of objects can be exported at a time. We offer four export options: 1) multilabeled, 2) multilayered, 3) semantic images, and 4) coordinates (see Figure 1). Instance annotations are typically expected to be exported as multilabeled (instance-aware) or multilayered (stack) grayscale images, the latter of which is useful for handling overlapping objects such as cytoplasms in cell culture images. Semantic images are binary foreground–background images of the target objects while coordinates (top-left corner [*x*,*y*] of the bounding rectangle appended by its width and height in pixels) can be useful training data for object detection applications including astrocyte localization (Suleymanova *et al.*, 2018) or in a broader aspect, face detection (Taigman *et al.*, 2014). All export options are supported for semantic annotation; however, we note that in instance-aware options (multilabeled or multilayered mask and coordinates) only such objects are distinguished whose contours do not touch on the annotation image.

OpSeF compatibility

OpSeF (Open Segmentation Framework; Rasse *et al.*, 2020) is an interactive python notebook-based framework (available at https://github.com/trasse/OpSeF-IV) that allows users to easily try different DL segmentation methods in customizable pipelines. We extended AnnotatorJ to support the data structure and format used in OpSeF to allow seamless integration in these pipelines, so users can manually modify, create, or classify objects found by OpSeF in AnnotatorJ, then export the results in a compatible format for further use in the former software. A user guide is provided in the documentation of https://github.com/trasse/OpSeF-IV.

ImageJ

ImageJ (or Fiji: Fiji is just ImageJ; Schindelin *et al.*, 2012) is an open-source, cross-platform image analysis software tool in Java that has been successfully applied in numerous bioimage analysis tasks (segmentation [Legland *et al.*, 2016; Arganda-Carreras *et al.*, 2017], particle analysis [Abramoff *et al.* 2004], etc.) and is supported by a broad range of community, comprising of biomage analyst end users and developers as well. It provides a convenient framework for new developers to create their custom plugins and share them with the community. Many typical image analysis pipelines have already been implemented as a plugin, for example, U-Net segmentation plugin (Falk *et al.*, 2019) or StarDist segmentation plugin (Schmidt *et al.*, 2018).

U-Net implementation

We used the DL4J (http://deeplearning4j.org/) implementation of U-Net in Java. DL4J enables building and training custom DL networks, preparing input data for efficient handling and supports both GPU and CPU computation throughout its ND4J library.

The architecture of U-Net was first developed by Ronneberger *et al.* (Ronneberger *et al.*, 2015) and was designed to learn medical image segmentation on a small training set when a limited amount of labeled data is available, which is often the case in biological contexts. To handle touching objects as often is the case in nuclei segmentation, it uses a weighted cross entropy loss function to enhance the object-separating background pixels.

Region growing

A classical image processing algorithm, region growing (Haralick and Shapiro, 1985; Adams and Bischof, 1994) starts from initial seed points or objects and expands the regions towards the object boundaries based on the intensity changes on the image and constraints on distance or shape. We used our own implementation of this algorithm.

ACKNOWLEDGMENTS

R.H., N.M., A.D., and P.H. acknowledge support from the LENDULET-BIOMAG grant (Grant no. 2018-342), from the European Regional Development Funds (GINOP-2.3.2-15-2016-00006, GINOP-2.3.2-15-2016-00026, and GINOP-2.3.2-15-2016-00037), from the H2020-discovAIR (874656), and from Chan Zuckerberg Initiative, Seed Networks for the HCA-DVP. We thank Krisztián Koós for the technical help, and Máté Görbe and Tamás Monostori for testing the plugin.

REFERENCES

The best image annotation platforms for computer vision (2018, October 30). + an honest review of each, https://hackernoon.com/the-best-image-annotation-platforms-for-computer-vision-an-honest-review-of-each-dac7f565fea.

Abramoff MD, Magalhaes PJ, Ram SJ (2004). Image processing with ImageJ. Biophoton Int, 11, 36–42.

Adams R, Bischof L (1994). Seeded region growing, IEEE Trans Pattern Anal Mach Intell 16, 641–647.

Arganda-Carreras I, Kaynig V, Rueden C, Eliceiri KW, Schindelin J, Cardona A, Sebastian Seung H (2017). Trainable Weka segmentation: a machine learning tool for microscopy pixel classification. Bioinformatics 33, 2424–2426.

Arganda-Carreras I, Turaga SC, Berger DR, Cireşan D, Giusti A, Gambardella LM, Schmidhuber J, Laptev D, Dwivedi S, Buhmann JM, *et al.* (2015). Crowdsourcing the creation of image segmentation algorithms for connectomics. Front Neuroanat 9, 142.

Arzt M (2017). https://imagej.net/Labkit.

Aubreville M, Bertram C, Klopfleisch R, Maier A (2018). SlideRunner. In: Bildverarbeitung für die Medizin 2018, Heidelberg, Berlin: Informatik aktuell, Springer Vieweg, pp. 309–314. https://doi.org/10.1007/978-3-662-56537-7_81.

Badrinarayanan V, Kendall A, Cipolla R (2017). SegNet: a deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans Pattern Anal Mach Intell 39, 2481–2495.

Berg S, Kutra D, Kroeger T, StraehleCN, KauslerBX, HauboldC, SchieggM, AlesJ, BeierT, RudyM, *et al.* (2019). ilastik: interactive machine learning for (bio)image analysis. Nat Methods 16, 1226–1232.

Caicedo JC, Roth J, Goodman A, Becker T, Karhohs KW, Broisin M, Molnar C,BeckerT, KarhohsKW, BroisinM, *et al.* (2019). Evaluation of deep learning strategies for nucleus segmentation in fluorescence images. Cytometry A 95, 952–965.

Cancer Genome Atlas Research Network (2008). Comprehensive genomic characterization defines human glioblastoma genes and core pathways. Nature 455, 1061–1068.

Coelho LP, Shariff A, Murphy RF (2009). Nuclear segmentation in microscope cell images: a hand-segmented dataset and comparison of algorithms. 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro. Available at https://doi.org/10.1109/isbi.2009.5193098.

Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B, *et al.* (2016). The cityscapes dataset for semantic urban scene understanding. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Available at https://doi.org/10.1109/cvpr.2016.350.

Dutta A, Zisserman A (2019). The VIA annotation software for images, audio and video. In Proceedings of the 27th ACM International Conference on Multimedia - MM '19. Available at https://doi.org/10.1145/3343031.3350535.

Eclipse Deeplearning4j Development Team. Deeplearning4j: open-source distributed deep learning for the JVM, Apache Software Foundation License 2.0. http://deeplearning4j.org

Falk T, Mai D, Bensch R, Çiçek Ö, Abdulkadir A, Marrakchi Y, Böhm A, Deubner J, Jäckel Z, Seiwald K, *et al.* (2019). U-Net: deep learning for cell counting, detection, and morphometry. Nat Methods 16, 67–70.

Frank E, Hall MA, Witten IH (2016). "The WEKA Workbench", online appendix for Data Mining: Practical Machine Learning Tools and Techniques, 4th ed., Morgan Kaufmann.

Girshick R, Donahue J, Darrell T, Malik J (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In 2014 IEEE Conference on Computer Vision and Pattern Recognition. Available at https://doi.org/10.1109/cvpr.2014.81.

Grigorescu S, Trasnea B, Cocias T, Macesanu G (2019). A survey of deep learning techniques for autonomous driving. J Field Rob 37, 362–386.

Haralick RM, Shapiro LG (1985). Image segmentation techniques. Applications of Artificial Intelligence II. Available at https://doi.org/10.1117/12.948400.

He K, Gkioxari G, Dollar P, Girshick R (2017). Mask R-CNN. In 2017 IEEE International Conference on Computer Vision (ICCV). Available at https://doi.org/10.1109/iccv.2017.322.

Hinton G, Deng L, Yu D, Dahl G, Mohamed A-R, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath TN, *et al.* (2012). Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process Mag 29, 82–97.

Hollandi R, Szkalisity A, Toth T, Tasnadi E, Molnar C, Mathe B, Grexa I, Molnar J, Balind A, Gorbe M, *et al.* (2020). nucleAIzer: a parameter-free deep learning framework for nucleus segmentation using image style transfer. Cell Syst 10, 453–458.e6.

Kass M, Witkin A, Terzopoulos D (1988). Snakes: active contour models. Int J Comput Vis 1, 321–331.

Kumar N, Verma R, Sharma S, Bhargava S, Vahadane A, Sethi A (2017). A dataset and a technique for generalized nuclear segmentation for computational pathology. IEEE Trans Med Imaging 36, 1550–1560.

Legland D, Arganda-Carreras I, Andrey P (2016). MorphoLibJ: integrated library and plugins for mathematical morphology with ImageJ. Bioinformatics 32, 3532–3534.

Ljosa V, Sokolnicki KL, Carpenter AE (2012). Annotated high-throughput microscopy image sets for validation. Nat Methods 9, 637.

Marée R, Rollus L, Stévens B, Hoyoux R, Louppe G, Vandaele R, Begon J-M, Kainz P, Geurts P, Wehenkel L, *et al.* (2016). Collaborative analysis of multi-gigapixel imaging data using Cytomine. Bioinformatics 32, 1395–1401.

Morikawa R (July 18, 2019). 24 best image annotation tools for computer vision. Available at https://lionbridge.ai/articles/image-annotation-tools-for-computer-vision/.

Moshkov N, Mathe B, Kertesz-Farkas A, Hollandi R, Horvath P (2020). Test-time augmentation for deep learning-based cell segmentation on microscopy images. Sci Rep 10, 5068.

Naylor P, Lae M, Reyal F, Walter T (2017). Nuclei segmentation in histopathology images using deep neural networks. In 2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017). Available at https://doi.org/10.1109/isbi.2017.7950669.

Naylor P, Lae M, Reyal F, Walter T (2019). Segmentation of nuclei in histopathology images by deep regression of the distance map. IEEE Trans Med Imaging 38, 448–459.

Pachitariu M, Stringer C, Dipoppa M, Schröder S, Rossi LF, Dalgleish H, Carandini M, (2017). Suite2p: beyond 10,000 neurons with standard two-photon microscopy. BioRxiv. https://doi.org/10.1101/061507.

Rasse TM, Hollandi R, Horvath P (2020). OpSeF IV: open source Python framework for segmentation of biomedical images. BioRxiv. https://doi.org/10.1101/2020.04.29.068023.

Redmon J, Divvala S, Girshick R, Farhadi A (2016). You only look once: unified, real-time object detection. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Available at https://doi.org/10.1109/cvpr.2016.91.

Redmon J, Farhadi A (2018). YOLOv3: an incremental improvement. arXiv https://arxiv.org/abs/1804.02767.

Ronneberger O, Fischer P, Brox T (2015). U-Net: convolutional networks for biomedical image segmentation. In Lecture Notes in Computer Science, Springer, Vol. 9351, 234–241.

Rubens U, Hoyoux R, Vanosmael L, Ouras M, Tasset M, Hamilton C, Longuespée R, Marée R (2019). Cytomine: toward an open and collaborative software platform for digital pathology bridged to molecular investigations. Prot Clin Appl 13, 1800057.

Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, *et al.* (2015). ImageNet large scale visual recognition challenge. Int J Comput Vis, 115, 211–252.

Schindelin J, Arganda-Carreras I, Frise E, Kaynig V, Longair M, Pietzsch T, Preibisch S, Rueden C, Saalfeld S, Schmid B, *et al.* (2012). Fiji: an open-source platform for biological-image analysis. Nat Methods 9, 676–682.

Schmidt U, Weigert M, Broaddus C, Myers G (2018). Cell detection with star-convex polygons. In Lecture Notes in Computer Science, Springer, 265–273. Crossref. Web.

Schneider CA, Rasband WS, Eliceiri KW (2012). NIH image to ImageJ: 25 years of image analysis. Nat Methods 9, 671–675.

Schroff F, Kalenichenko D, Philbin J (2015). FaceNet: a unified embedding for face recognition and clustering. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 815–823. doi: 10.1109/CVPR.2015.7298682.

Sommer C, Straehle C, Kothe U, Hamprecht FA (2011). Ilastik: interactive learning and segmentation toolkit. In IEEE International Symposium on Biomedical Imaging: From Nano to Macro 2011, 230–233.

Suleymanova I, Balassa T, Tripathi S, Molnar C, Saarma M, Sidorova Y, Horvath P (2018). A deep convolutional neural network approach for astrocyte detection. Sci Rep 8, 12878.

Sun Y, Wang X, Tang X, (2014). Deep learning face representation from predicting 10,000 classes. In 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1891—1898. doi: 10.1109/CVPR.2014.244.

Taigman Y, Yang M, Ranzato M'A, Wolf L (2014). DeepFace: closing the gap to human-level performance in face verification. In 2014 IEEE Conference on Computer Vision and Pattern Recognition. Available at https://doi.org/10.1109/cvpr.2014.220.

D Article (pre-print) ‘Learning representations for image-based profiling of perturbations‘

Learning representations for image-based profiling of perturbations

Authors

Nikita Moshkov, Michael Bornholdt, Santiago Benoit, Matthew Smith, Claire McQuin, Allen Goodman, Rebecca Senft, Yu Han, Mehrtash Babadi, Peter Horvath, Beth A. Cimini, Anne E. Carpenter, Shantanu Singh, Juan C. Caicedo*

* Corresponding author

Abstract

Measuring the phenotypic effect of treatments on cells through imaging assays is an efficient and powerful way of studying cell biology, and requires computational methods for transforming images into quantitative data that highlights phenotypic outcomes. Here, we present an optimized strategy for learning representations of treatment effects from high-throughput imaging data, which follows a causal framework for interpreting results and guiding performance improvements. We use weakly supervised learning (WSL) for modeling associations between images and treatments, and show that it encodes both confounding factors and phenotypic features in the learned representation. To facilitate their separation, we constructed a large training dataset with Cell Painting images from five different sources to maximize experimental diversity, following insights from our causal analysis. Training a WSL model with this dataset successfully improves downstream performance, and produces a reusable convolutional network for image-based profiling, which we call *Cell Painting CNN*. We conducted a comprehensive evaluation of our strategy on three publicly available Cell Painting datasets, discovering that representations obtained by the Cell Painting CNN can improve performance in downstream analysis up to 25% with respect to classical features, while also being more computationally efficient.

Introduction

High-throughput imaging and automated image analysis are powerful tools for studying the inner workings of cells under experimental interventions. In particular, the Cell Painting assay [1,2] has been adopted both by academic and industrial laboratories to evaluate how perturbations alter overall cell biology. It has been successfully used for studying compound libraries [3–5], predicting phenotypic activity [6–9], and profiling human disease [10,11], among many others. To reveal the phenotypic outcome of treatments, image-based profiling transforms microscopy images into rich high-dimensional data using morphological feature extraction [12]. Cell Painting datasets with thousands of experimental interventions provide a unique opportunity to use machine learning for obtaining causal representations of the phenotypic outcomes of treatments.

Improved feature representations of cellular morphology have the potential to increase the sensitivity and robustness of image-based profiling to support a wide range of discovery applications [13,14]. Feature extraction has been traditionally approached with classical image processing [15,16], which is based on manually engineered features that may not capture all the relevant phenotypic variation. Several studies have used convolutional neural networks (CNNs) pre-trained on natural images [17–19], which are optimized to capture variation of macroscopic objects instead of images of cells. To recover causal representations of treatment effects, feature representations need to be sensitive to subtle changes in morphology. However, classical features and pre-trained networks may not have sufficient expressive power to realize that potential. Representation learning has been used as a tool to learn domain-specific features from cellular images in a data-driven way [4,20–23], but it also brings unique challenges to avoid being dominated by confounding factors [24,25].

In this paper, we investigate the problem of learning representations for image-based profiling with Cell Painting. Our goal is to identify an optimal strategy for learning cellular features, and then use it for training models that recover improved representations of the phenotypic outcomes of treatments. We use a causal framework to reason about the challenges of learning representations of cell morphology (e.g. confounding factors), which naturally fits in the context of perturbation experiments [26,27], and serves as a tool to optimize the workflow and yield better performance (Figure 1). In addition, we adopted a quantitative evaluation of the impact of feature representations in a biological downstream task, to guide the search for an optimized workflow. The evaluation is based on querying a reference collection of treatments to find biological matches in a perturbation experiment. In each evaluation, cell morphology features change to compare different strategies, while the rest of the image-based profiling workflow remains constant. Performance is measured using metrics for the quality of a ranked list of results for each query (Figure 1H). With this evaluation framework, we conduct an extensive analysis on three publicly available Cell Painting datasets.

Within the proposed causal framework, weakly supervised learning (WSL) [20] is a powerful approach to model the associations between images and treatments, and we found that it captures rich cellular features that simultaneously encode confounding factors and phenotypic outcomes as latent variables. Our analysis indicates that to disentangle them and recover the causal representations of the true outcome of perturbations, it is important to train models with highly diverse data for improving the ability of models to learn the difference between the two types of variation. Therefore, we constructed a new dataset that combines variation from five different sources to maximize the diversity of treatments and confounding factors (Figure 1F). As a result, we successfully trained a new, reusable model: the Cell Painting CNN (Figure 1G), which yields better performance in the evaluated benchmarks and displays sufficient generalization ability to profile other datasets effectively.

Results

Recovering features of the causal effects of treatments

We use a causal model as a conceptual framework to reason and analyze the results of representation learning strategies. Figure 1B presents the causal graph with four variables: interventions (treatments $\mathbf{T}$), observations (images $\mathbf{O}$), outcomes (phenotypes $\mathbf{Y}$) and confounders (e.g. batches $\mathbf{C}$). For simplicity, we assume that there is a single context ($\mathbf{X}$, not in the diagram) for experimental treatments, which are clonal cells of an isogenic cell line; perturbation experiments with multiple cell lines may need a different model. Some variables are observables (shaded circles), while others represent latent variables (empty circles). This graph is a model of the causal assumptions we make for representation learning and for interpreting the results.

We are interested in estimating a continuous, multidimensional representation of treatment outcomes ($\mathbf{Y}$), which can later be used in downstream analysis tasks. A major issue for learning a causal representation of the phenotypic outcome of treatments is technical variation ($\mathbf{C}$), which groups a set of factors introducing unwanted variation in images. Note that images, phenotypes and treatments respond to changes in technical variation. Images are impacted by artifacts in image acquisition, including microscope settings and assay preparation. Treatments are impacted by plate map designs that are not fully randomized and usually group treatments in specific plate positions. Confounders also impact phenotypes due to variations in cell density and other conditions that make cells grow and respond differently.

The main direct cause of phenotypic changes in cells in a perturbation experiment is the treatment. We observe treatment outcomes indirectly through imaging assays, and thus, we need image analysis to recover the phenotypic effect and to separate it from unwanted variation. A representation of the phenotypic effect can be obtained with the workflow depicted in Figures 1C-E, which illustrates three major steps: 1) modeling the correlations between images and treatments using a CNN trained with weakly supervised learning (WSL), 2) using batch correction to learn a transformation of the latent representation of images obtained from intermediate layers of the CNN, and 3) generating representations of treatment effects in cellular morphology for downstream analysis.

We use WSL [20] (Figure 1C) for obtaining representations of the phenotypic outcome of treatments by training a classifier to distinguish all treatments from each other. This approach models the relationship between observed images and treatments using an EfficientNet [28] with a classification loss (Methods), and makes the following assumptions: 1) if a treatment has an observable effect then it can be seen in images, and therefore, training a CNN helps identify visual features that make it detectably different from all other treatments. 2) Treatment labels in the classification task are weak labels because they are not the final outcome of interest, they do not reflect expert biological ground truth, and there is no certainty that all treatments produce a phenotypic outcome, nor that they produce a different phenotypic outcome from each other. 3) Intermediate layers of the CNN trained with treatment labels capture all visual variation of images as latent variables, including confounders and causal phenotypic features.

To recover the phenotypic features of treatments from the latent representations of the weakly supervised CNN, we employ a batch correction model inspired by the Typical Variation Normalization (TVN) technique [18]. This transform aims to reduce the variation associated with confounders and amplify features caused by phenotypic outcomes (Figure 1D). The main idea of this approach is to use negative control samples as a model of unwanted variation under the assumption that their phenotypic features should be neutral, and therefore differences in control images reflect mainly confounding factors. We follow this assumption and use a sphering transformation (Methods) to learn a function that projects latent features from the CNN to a corrected feature space that preserves the phenotypic features caused by treatments.

As a result of learning latent features with WSL and correcting for unwanted variation, the new feature space more cleanly highlights feature representations of treatment effects. These representations are continuous multidimensional estimators of the true phenotypic outcome of treatments in the perturbation experiment, and because of their unbiased design can be used to approach various questions and downstream analysis tasks, such as predicting compound bioactivity or inferring the impact of cancer variants (Figure 1E). In this paper, we conduct a quantitative evaluation of these representations in a biological matching task (genetic pathway or MoA) using a guilt by association approach, which is a meaningful way to validate the biological relevance of the resulting features.

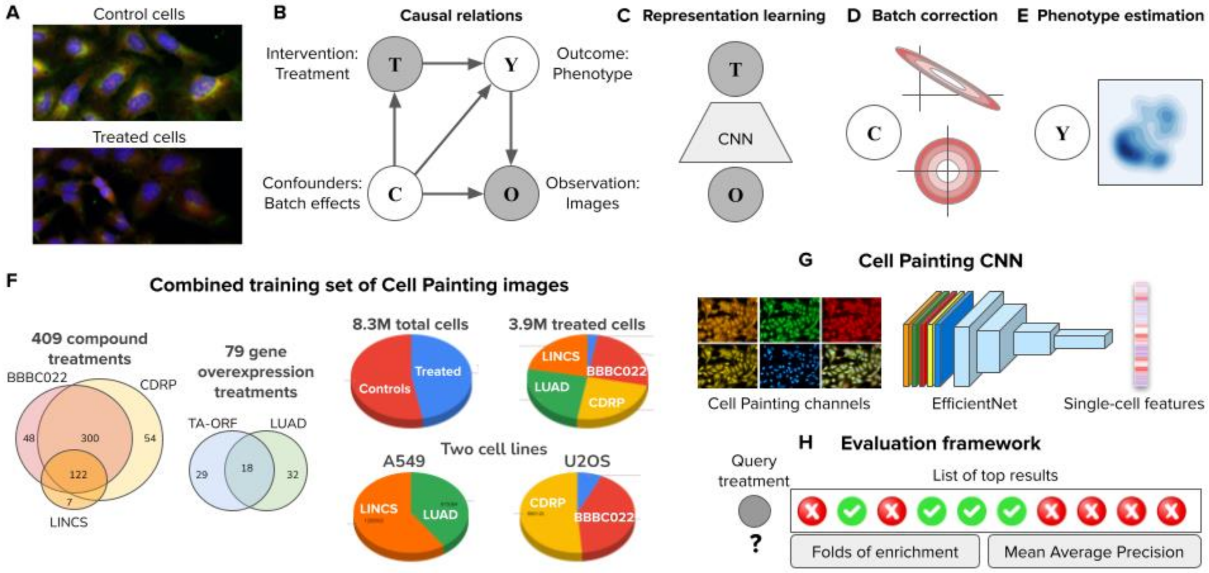


Figure 1. Framework for analyzing image-based profiling experiments. *A) Example Cell Painting images from the BBBC037 dataset of control cells (empty status) and one experimental intervention (JUN wild-type overexpression) in the U2OS cell line. B) Causal graph of a conventional high-throughput Cell Painting experiment with two observables in shaded circles (treatments and images) and two latent variables in empty circles (phenotypes and batch effects). The arrows indicate the direction of causation. C) Weakly supervised learning as a strategy to model associations between images (**O**) and treatments (**T**) using a convolutional neural network (CNN). The CNN captures information about the latent variables **C** and **Y** in the causal graph because both are intermediate nodes in the paths connecting images and treatments. D) Illustration of the sphering batch-correction method where control samples are a model of unwanted variation (top). After sphering, the biases of unwanted variation in control samples is reduced (bottom). E) The goal of image-based profiling is to recover the outcome of treatments by estimating a representation of the resulting phenotype, free from unwanted confounding effects. F) Statistics of the Combined Cell Painting dataset created to train a generalist model, which brings 488 treatments from 5 different publicly available sources (Methods): LINCS, LUAD, and the three datasets evaluated here; the Venn diagrams illustrate the common treatments among them. There are two types of treatments (compounds and gene overexpression), two types of controls (empty and DMSO), two cell lines (A549 and U2OS), for a total of 8.3 million single cells from 232 plates in the resulting training resource. G) Illustration of the Cell Painting CNN, an EfficientNet model trained on the dataset in F to extract features from single cells. H) The evaluation of performance is based on nearest neighbor queries performed in the space of phenotype representations to match treatments with the same phenotypic outcome. Performance is measured with two metrics: folds of enrichment and mean average precision (Methods).*

Representations learned by the Cell Painting CNN encode improved biological features

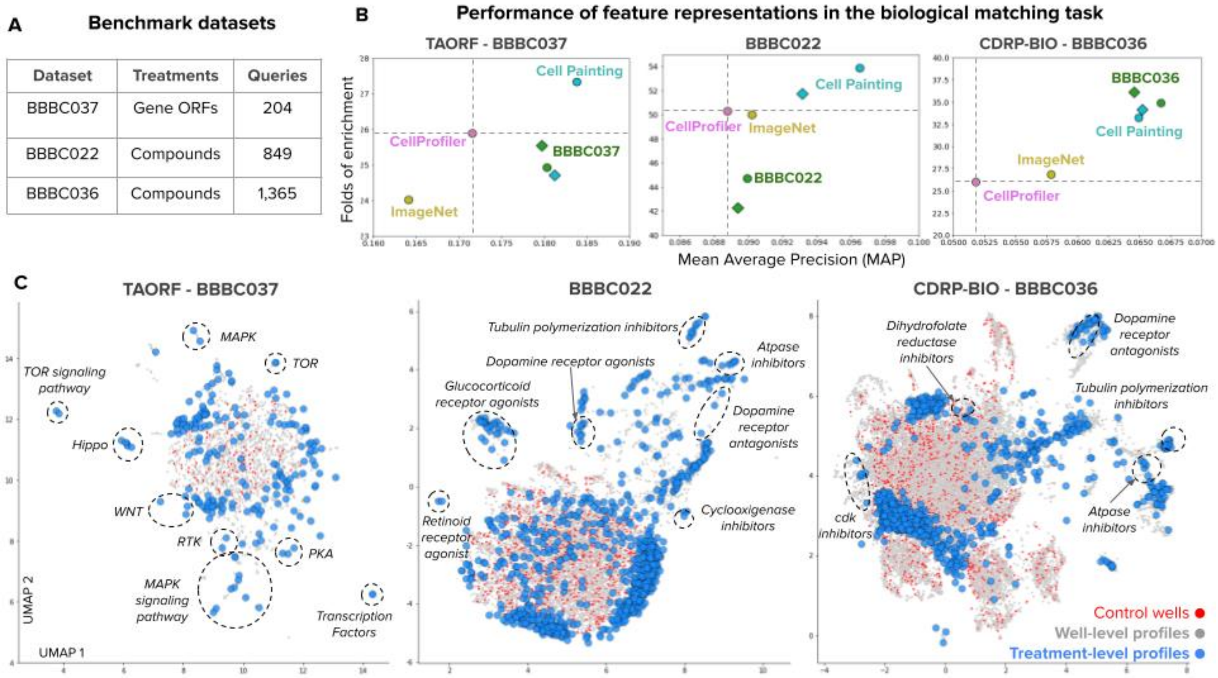
Figure 2C shows a UMAP projection [29] of the feature space obtained using our Cell Painting CNN for the three datasets evaluated in this study. From a qualitative perspective, the UMAP plot of the BBBC037 dataset (a gene overexpression screen) shows groups of treatments clustered according to their corresponding genetic pathway, and recapitulates previous observations of known biology [30]. The BBBC022 [31] and BBBC036 [32] datasets (compound screens) likewise show many treatments grouped together according to their mechanism of action (MoA).

For quantitative comparison of multiple feature extraction strategies, we simulate a user searching a reference library to find a “match” to their query treatment of interest. We used a leave-one-treatment-out strategy for all annotated treatments in the three benchmark datasets, following previous research in the field [18,33,34]. In all cases, queries and reference items are aggregated treatment-level profiles (Methods) matched using the cosine similarity (Methods). The result of searching the library with one treatment query is a ranked list of treatments in descending order of relevance. A result in the ranked list is considered a positive hit if it shares at least one biological annotation in common with the query; otherwise it is a negative result (Figure 1H). Biological annotations are genetic pathways for overexpression treatments in BBBC037 and MoAs for compound treatments in the BBBC022 and BBBC036 datasets. Then, we proceed to quantify performance using the folds of enrichment and mean average precision metrics (see Methods).

The generalized Cell Painting CNN model, trained with WSL on the combined set, performs better than baseline approaches in the task of biologically matching queries against a reference annotated library of treatments, across all three benchmarks (Figure 2B). We consider two baseline strategies in our evaluation: 1) creating image-based profiles with classical features obtained with custom CellProfiler pipelines (Methods), and 2) computing profiles with a CNN pre-trained on ImageNet, a dataset of natural images in RGB (Methods). Intuitively, we expect feature representations trained on Cell Painting images to perform better at the matching task than the baselines. In the case of CellProfiler, manually engineered features may not capture all the relevant phenotypic variation, and in the case of ImageNet pre-trained networks, they are optimized for macroscopic objects in 3-channel natural images instead of 5-channel fluorescence images of cells.

We found that ImageNet features showed variable performance compared to CellProfiler features (Figure 2B), sometimes yielding similar performance (BBBC022), sometimes lower performance (BBBC037) and sometimes slightly better performance (BBBC036). The three benchmarks used in this study are larger scale and more challenging than datasets used in previous studies [17,18] where it was observed that ImageNet features are typically more powerful than classical features. Our results indicate that, in large scale perturbation experiments with Cell Painting, ImageNet features do not conclusively capture more cellular-specific variation than manually engineered features using classical image processing.

Finally, we also assessed models trained on each dataset alone, rather than our generalized Cell Painting CNN model trained on various datasets. We found that narrowing the training to only the dataset at hand typically diminished performance as compared to the generalized model (green dots in Figure 2B), with one exception, BBBC036, where they had comparable performance. This brings us to the question of what is necessary to train a successful Cell Painting model, which we explore in the next sections.



Dataset	Treatments	Queries
BBBC037	Gene ORFs	204
BBBC022	Compounds	849
BBBC036	Compounds	1,365

Figure 2. Quantitative evaluation of feature representations of treatment effects. *The main evaluation task is biological matching, which simulates queries with a treatment and aims to find other treatments that match their phenotypic outcome (e.g. same mechanism of action or same genetic pathway). A) Table of the three benchmark datasets used in this study. The quantitative results for each dataset are the mean across the queries listed in this table. B) Performance of feature representations for the three evaluated datasets according to two metrics: Mean Average Precision (MAP) in the x axis and Folds of Enrichment in the y axis (see Methods). Each point indicates the mean of these metrics over all queries using the following feature representations: CellProfiler (pink), a CNN pre-trained on ImageNet (yellow), a CNN trained on the combined set of Cell Painting images (cyan, Figure 1F), and a CNN trained on Cell Painting images from the same dataset (green). CNNs trained on Cell Painting images have two results depending on the training-validation strategy (see Figure 3 for more details): leave plates out (diamonds) or leave cells out (circles). C) UMAP visualizations of the phenotypic profiles of treatments recovered with the Cell Painting CNN after batch correction for the three datasets evaluated in this work. The plot includes a projection of well-level profiles (gray points), control wells (red points), and aggregated treatment-level profiles of treatments (blue points). Dashed lines indicate clusters of treatment-level profiles where all or the majority of the points share the same biological annotation.*

Weakly supervised learning captures latent representations of confounders and phenotypic outcomes of treatments

Since we observed that models trained on images from each dataset alone do not succeed at improving performance with respect to baselines (green points in Figure 2B) we sought to determine whether WSL models are capturing more information from confounding factors than phenotypic effects (variables $\mathbf{C}$ and $\mathbf{Y}$ in the causal graph, respectively). WSL models are trained with a classification loss to detect the treatment in images of single cells (Figure 1C, Methods), which is a pretext task to learn representations. Given that we always know the treatment applied to cells in a well, we can quantify the success rate of such classifiers on this pretext task. We conducted single-cell classification experiments with WSL using two validation schemes, herein called leave-cells-out and leave-plates-out validation (Figure 3A), which aim to reveal how sensitive WSL is to biological and technical variation.

The leave-cells-out validation regime uses single cells from all plates and treatments in the experiment for training, and leaves a random fraction out for validation. By doing so, trained CNNs have the opportunity to observe the whole distribution of phenotypic features (all treatments) as well as the whole distribution of confounding factors (all batches or plates). In contrast, the leave-plates-out validation regime separates different technical replicates (plates) for training and validation, resulting in a model that still observes the whole distribution of treatments, but only partially sees the distribution of confounding factors.

The results in Figure 3B-D show a stark contrast in performance between the two validation strategies. When leaving cells out (results in blue), a CNN can accurately learn to classify single cells in the training and validation sets with only a minor difference in performance (Figure 3B); when leaving plates out (results in orange), the CNN learns to classify the training set well but fails to generalize correctly to the validation set, resulting in a major gap according to the learning curves (Figure 3B). The generalization ability of the two models is further highlighted in the validation results in Figures 3C,D, which present the distribution of precision, recall and F1-scores.

Importantly, while these WSL models trained on the same datasets alone exhibit a major difference in performance in the pretext classification task, their performance in the downstream analysis task is almost the same (green circles vs green diamonds in Figure 2B). In addition, WSL models trained on the full distribution of treatments usually fail to improve performance with respect to baseline approaches (green points in Figure 2B, except BBBC036). This dichotomy between performance in the pretext task and the downstream task reveals that WSL models leverage any way in which the relationship between images and treatments can be explained, including spurious correlations. In fact, the validation results of leaving-cells-out are an overly optimistic estimate of how well a CNN can recognize treatments in single cells, because the models leverage batch effects to make the correct connection. On the other hand, the results of leaving-plates-out are an overly pessimistic estimate because the CNN fails to generalize to unseen replicates with unknown confounding variation. The true estimate of performance in the pretext classification task is likely to be in the middle when accounting for confounding factors. This is indeed what we observe in the downstream analysis results: after batch correction, the representations of models trained with leave-cells-out and leave-plates-out come closer together in the biological matching task (green circles vs green diamonds in Figure 2B).

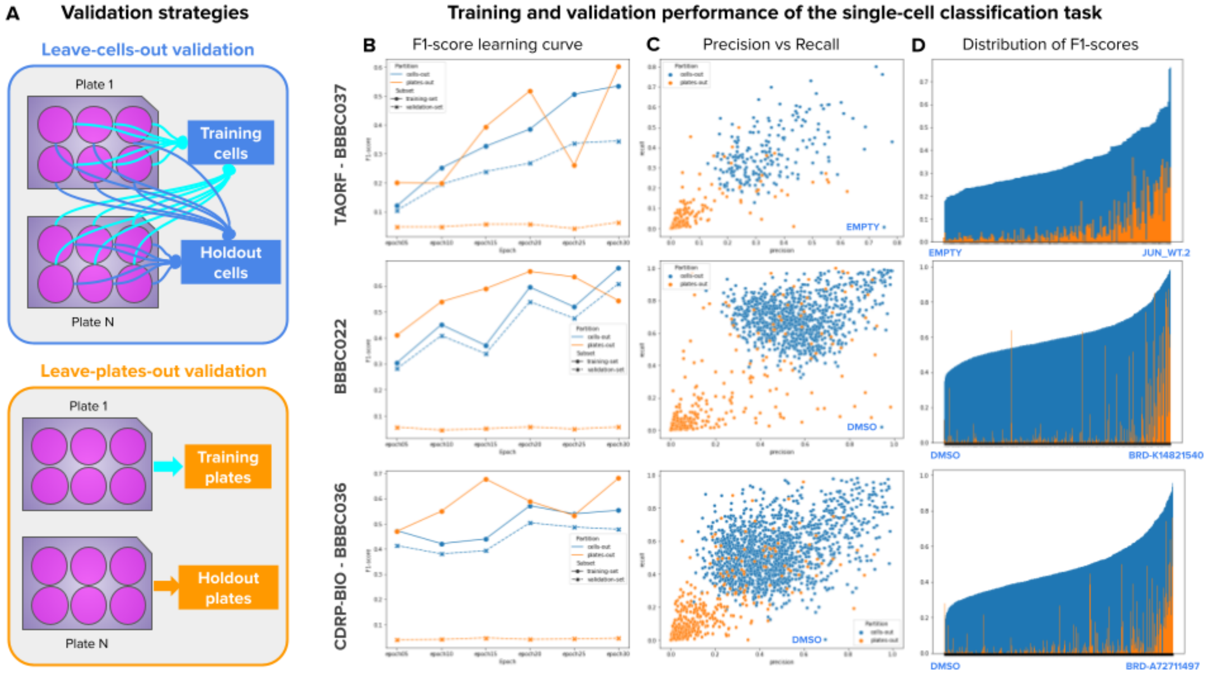


Figure 3. Validation strategies for the single-cell classification task in weakly supervised learning. *A) Illustration of the two strategies: leave-cells-out (in blue) uses cells from all plates in the dataset for training and leaves a random fraction out for validation. Leave-plates-out (in orange) uses cells from certain plates for training, and leaves entire plates out for validation. The difference in performance is primarily due to confounding factors. B) Learning curves of models trained with WSL for 30 epochs with all treatments from each dataset. The x-axis is the number of epochs and the y axis is the average F1-score. The color of lines indicates the validation strategy, and the style of lines indicates training (solid) or validation (dashed) data. C) Precision and recall results of each treatment in the single-cell classification task. Each point is a treatment (negative controls are labeled in blue), and the color corresponds to the validation strategy. D) Distribution of F1-scores for the single-cell classification task for each treatment: the x axis enumerates treatments (sorted by performance according to the leave-cells-out validation scheme), and the y-axis is the F1-score. The first and last treatments in the x axis are labeled.*

Treatments with strong phenotypic effect can improve performance

The WSL model depicted in Figure 1C captures direct associations between images (**O**) and treatments (**T**) in the causal graph, while encoding technical (**C**) and phenotypic (**Y**) variation as latent variables because both are valid paths to find correlations. Given that controlling the distribution of confounding factors does not change overall performance, in this section we explore the impact of controlling the distribution of phenotypic diversity. Our reasoning is that WSL learning favors correlations between treatments and images through the path in the causal graph that makes it easier to minimize the empirical error in the pretext task. Therefore, the variation of treatments with a weak phenotypic response is overpowered by confounding factors that are stronger relative to the phenotype.

We start by splitting treatments into two broad categories: treatments with strong phenotypic effect, and treatments with weak or no phenotypic effect. To measure the phenotypic strength of treatments we calculate the Euclidean distance between control and treatment profiles in the CellProfiler feature space after batch correction with a sphering transform. Then, we rank treatments in descending order and take a fraction of those most different from controls in the strong category (Methods). We interpret this procedure as a crude approximation of the average treatment effect (ATE), a causal parameter of intervention outcomes, because the Euclidean distance calculates the difference in expected values (aggregated profiles) of the outcome variable (phenotype) between the control and treated conditions. However, since we do not observe the control and treated condition in the same cells, this remains only an approximation of the ATE, even if the cells are isogenic clones of each other. We chose distances in the CellProfiler feature space as an independent prior for estimating treatment strength because these are non-trainable, and because in our experiments CellProfiler features exhibit more robustness to confounding factors (Supplementary Figure 2, and Supplementary Figure 3).

Next, we evaluated the performance of WSL models trained on strong treatments only and we found that performance tends to improve in the downstream biological matching task with respect to training with the full distribution of treatments in each dataset (Supplementary Figure 1). These results were obtained by training under a leave-plates-out validation regime, which also restricts the distribution of confounding factors. The trend indicates that it is possible to break the limitations of WSL for capturing useful associations between images and treatments in the latent variables without being overpowered by confounding factors. However, while the trend is generally positive, the results are still similar to baseline approaches, which suggests that training may need a higher diversity of strong treatments to reach improved performance.

Training on highly diverse experimental conditions yields improved representations

To increase the diversity of experimental conditions in the training set, we created a combined training resource by collecting strong treatments from five different dataset sources, including the three benchmarks evaluated in this work plus two additional publicly available Cell Painting datasets (Figure 1F, and Methods). In total, this combined set has 488 strong treatments, two types of negative controls, and examples from more than 200 plates, resulting in training data with high experimental diversity with respect to the two latent variables in the causal graph: high technical variation (confounders $\mathbf{C}$) and high phenotypic variation through strong treatments (outcomes $\mathbf{Y}$).

Training on this combined Cell Painting dataset of strong treatments consistently improves performance and yields better results than the baselines in all benchmarks (cyan points in Figure 2B). According to the MAP metric, a WSL model trained on the highly diverse combined set improves performance up to 7%, 8% and 25% relative to CellProfiler features on BBBC037, BBBC022 and BBBC036 respectively (difference of cyan points vs pink points in the x axis of Figure 2B). Similar improvements are observed in other metrics as well. Importantly, this dataset allowed us to train a single model once and profile all the three benchmarks without re-training or fine-tuning on each of them, demonstrating that the model captures features of Cell Painting images relevant to distinguish more effectively the variation of the two latent variables of the causal model.

The dimensionality of the feature space of the Cell Painting CNN is also more compact than CellProfiler and the ImageNet CNN model (Supplementary Figure 4B), with the intermediate layer Conv6A of the EfficientNet B0 network being the best source of latent representations for downstream analysis in all of our experiments (Supplementary Figure 4D). This layer, after spatial average pooling, results in 672 features, compared to 1,700 of CellProfiler and 3,360 of CNN ImageNet (672 for each of the five imaging channels). The dimensionality of single-cell features has an impact on storage space, especially for large scale experiments, making the CNN Cell Painting an efficient choice too (Supplementary Figure 4C).

Learned representations separate factors of variation and facilitate batch correction

Batch-correction is a crucial step for image-based profiling regardless of the feature space of choice. We hypothesized that a rich representation might encode both confounders and phenotypic features in a way that facilitates separating one type of variation from the other, i.e. disentangles the sources of variation. To test this, we evaluated how representations respond to the sphering transform, a linear transformation for batch correction based on singular value decomposition SVD (Methods). Sphering first finds directions of maximal variance in the set of control samples and then reverses their importance by inverting the eigenvalues. This transform makes the assumption that large variation is associated with confounders and subtle variation is associated with phenotypes. Thus, sphering can succeed at recovering the phenotypic effects of treatments if the feature space separates the sources of variation in this way.

The results in Figure 4B show that all the methods benefit from batch correction with the sphering transform, indicated by the upward trend of all curves from low performance with no batch correction to improved performance as batch correction increases. Downstream performance in the biological matching task improves by about 50% on average when comparing against raw features without correction. The UMAP plots in Figure 4A show the Cell Painting CNN feature space for well-level profiles before batch correction. When colored by Plate IDs, the data points are fragmented, and the density functions in the two UMAP axes indicate concentration of plate clusters. After sphering, the UMAP plots in Figure 4C show more integrated data points and better aligned density distributions of plates. The performance of the Cell Painting CNN in the biological matching task also improves upon the baselines (Figure 4B) and displays a consistent ability to facilitate batch correction in all the three datasets, unlike the CNN - ImageNet.

The sphering transform, while effective, is still far from perfect, and further research is needed to better disentangle confounding from phenotypic variation, potentially using nonlinear transformations.

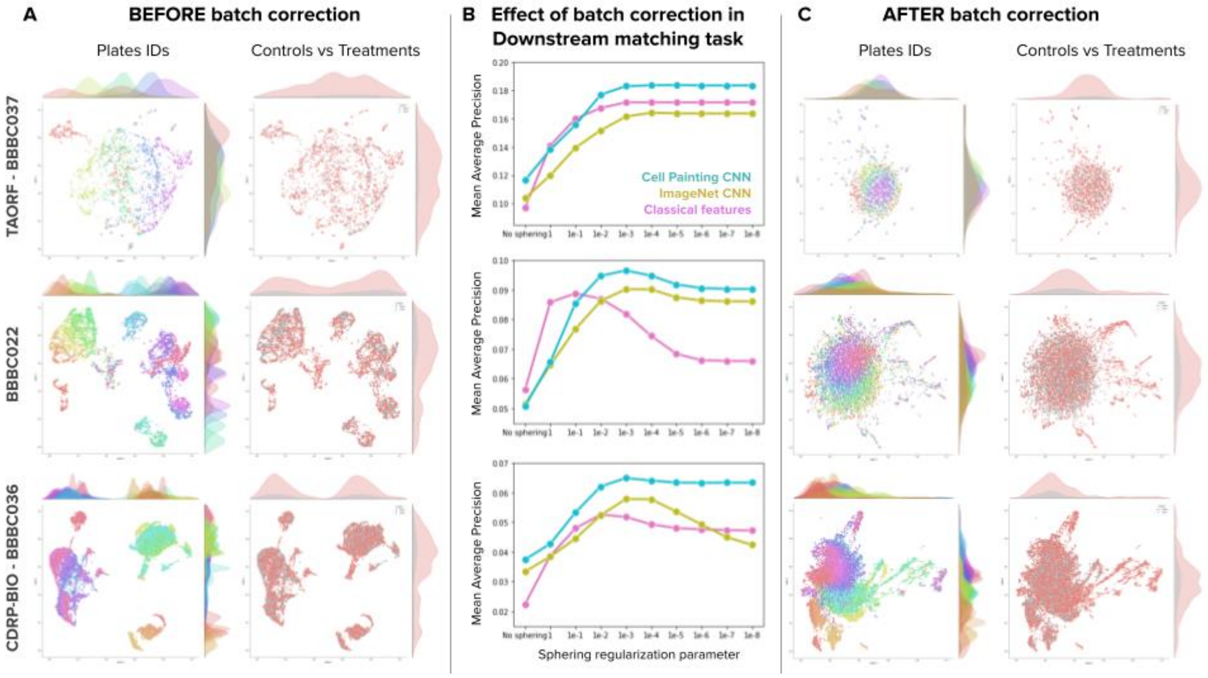


Figure 4. Effect of batch correction on feature representations. *Batch correction is based on the sphering transform and applied at the well-level, before treatment-level profiling (Methods). A) UMAP plots of well-level profiles before batch correction for the three benchmark datasets (rows) colored by plate IDs (left column) and by control vs treatment status (right column). The UMAP plots display density functions on the x and y axes for each color group to highlight the spread and clustering patterns of data. B) Effect of batch correction in the biological matching task. The x axis indicates the value of the regularization parameter of the sphering transform (smaller parameter means more regularization), with no correction in the leftmost point and then in decreasing order. The y axis is Mean Average Precision in the biological matching task. C) UMAP plots of well-level profiles after batch correction for the three benchmark datasets with the same color organization as in A.*

Discussion

In this study, we used three large Cell Painting benchmarks to evaluate training strategies for carrying out successful queries identifying matching compounds or genes using image-based profiling. Using a causal graph as a conceptual framework for analyzing the results, we found that WSL captures confounding and phenotypic variation in the latent variables. We also found that treatments with strong estimated average treatment effect have the potential to improve the quality of representations. Thus, we constructed a large training resource by combining five sources of Cell Painting data to maximize phenotypic and technical variation for training a reusable model for learning features. This model successfully improved performance in all the benchmarks and while also being computationally efficient.

The fact that the best-performing strategy involved training a single model once and profiling all the three benchmarks without re-training or fine-tuning has a remarkable implication: it indicates that generating large experimental datasets with a diversity of phenotypic impacts could be used to create a single model for the community that could be transformational in the same way that models trained on ImageNet have enabled transfer learning on natural image tasks. Private companies with large image sets might create and share such models. Alternatively, the upcoming public release of the JUMP-Cell Painting Consortium's dataset of more than 100,000 chemical and genetic perturbations, collected across 12 different laboratories in academia and industry, would be an excellent target for such an effort [35].

Methods

1. Datasets

1.1 Benchmarks and ground truth annotations

For this study, we employ five publicly available Cell Painting datasets: gene overexpression (BBBC037[34] and BBBC043 [11]) and compound screening (BBBC022 [31], BBBC036 [32] and LINCS [5]). BBBC037, BBBC022 and BBBC036 are U2OS cell-line datasets and were used for training and evaluation of profiling, while BBBC043 and LINCS are A549 cell-line and those were only partially used to construct a combined Cell Painting dataset (discussed further). Compounds in BBBC022, BBBC036 and LINCS partially overlap. In the compound screening datasets, DMSO was used as a negative control, in gene overexpression datasets genes were not perturbed in the negative control samples.

BBBC036 and BBBC037, BBBC022 did undergo quality control by analyzing the extracted features with principal component analysis. The outliers observed in the first two principal components were suspected to be candidates for exclusion. Those were visually inspected and found to be noisy or empty and not suitable for training and evaluation. With this quality control, two wells were removed from BBBC037 and 43 wells from BBBC022. Nothing was filtered from BBBC036.

If treatment was present in multiple concentrations in BBBC022 and BBBC036, we kept only the maximum concentration in the dataset for further training and evaluation.

The ground-truth annotations were reused from [34] with minor modifications (see Data Availability section). Only treatments with at least two replicates left after quality control are included in the ground-truth.

1.2 Selection of the strong treatments

The strongest treatments are selected using batch-corrected features obtained with CellProfiler (with sphering regularization parameter 1e-2) in the following way:

For each plate:

1. Calculate the median profile of the control subgroup of wells in the plate (median control profile)
2. For each treatment, calculate Euclidean distances between the treatment well-level feature vector and the median control profile feature vector.
3. Calculate Euclidean distances between the median control profile feature vector and each control well feature vector from the same plate.
4. Use the distances from the previous step to calculate the mean and standard deviation.
5. Using statistics obtained in step 4, apply Z-scoring to the distances obtained in step 2.

After obtaining the Z-scores for each plate, calculate the mean Z-score for each treatment. Rank treatments by mean Z-score.

1.3 Combined Cell Painting dataset

Motivated to train robust neural network models for feature extraction and an, we decided to take advantage of combining five publicly available Cell Painting datasets to maximize both phenotypic and technical variation. Treatments for the combined Cell Painting dataset are selected using the approach described in the previous section.

Technical variation is primarily encoded by control wells that are present in each plate of the experiment in several wells. Additionally, the negative controls are different between compound screening datasets and genetic perturbation datasets. To maximize the biological variation, first, we incorporate Cell Painting datasets of different cell-lines and also try to make an overlap of treatments between different cell lines to highlight the effect of the treatment. Second, we aimed to select the strongest phenotypes out of every dataset.

The selection started with BBBC022 and BBBC036 datasets. We selected the top 500 strongest treatments according to our approach from BBBC022 and took an overlap between BBBC022 and BBBC036, which resulted in 301 treatments. We additionally selected 50 from BBBC022 and 62 from BBBC036. Out of those 413 treatments, 122 overlapped with the LINCS dataset. We additionally selected 7 random treatments from LINCS, from top 20 (by number of associated treatments) MoAs.

Selection from BBBC037 and BBBC043 was similar, we selected 28 overlapping genes To increase overlapping we considered the selected “wildtype” genes as the same from different datasets, even though in BBBC037 the wildtypes were annotated to have different sub-populations. Additionally, we selected 29 top strongest perturbations from the BBBC037 dataset and the top strongest perturbations 32 from BBBC043 from non-overlapping subsets.

Negative controls from compound screening datasets and negative controls from gene overexpression datasets are considered as different classes in the combined dataset. Not all control wells were included from the LINCS dataset in the final dataset, as the control cells would dominate the final dataset, so we randomly sampled three control wells per plate.

As the final step, the treatments with less than 100 cells were filtered out.

In total the dataset contains 490 classes (488 for treatments and 2 for negative controls), 8 423 455 individual single-cells (47% treatment and 53% control cells), the diagrams are in Figure 1C.

2. Profiling workflow

2.1 Segmentation

The cell segmentation for all evaluated (BBBC037, BBBC022 and BBBC036) datasets is performed with methods built in CellProfiler v2 based on Otsu thresholding [36] and propagation method [37] based on Voronoi diagrams [38] or watershed from [39]. The segmentation is two-stepped: first, the images stained with Hoechst (DNA channel) were segmented using global Otsu thresholding. This prior information is then used in the second step: cell segmentation with the propagation or watershed method. The input channel for the second step depends on the dataset, as well as the other specific parameters of segmentation. The segmentation part of the pipelines is available in the published CellProfiler pipelines (see Code availability section).

2.2 Single-cell feature extraction with CellProfiler

Feature extraction for evaluated datasets was performed with CellProfiler 2. The feature extraction steps described in CellProfiler pipelines that are published together with used datasets. and can be grouped in several stages: 1) Data loading - load full image 2) Illumination correction for each channel 3) Identification of cell nuclei 4) Identification of cells using identified nuclei 4) Measurements: intensity, context, radial distribution, size and shape, texture 5) Export the features and cell outlines. Parameters of feature extraction can be found in CellProfiler pipelines which are available in published pipelines (see Code availability section).

2.3 CellProfiler features

The features of CellProfiler are designed to be human-readable and grouped into three large groups: “Cell”, “Cytoplasm” and “Nuclei”. Each of those feature groups has several common subgroups, such as shape features, intensity-based features, texture features and context features [12]. The resulting size of a feature vector is approximately 1800.
In this analysis, we reused well-level CellProfiler features from S3 buckets (see Data availability section) to generate the baseline results.

2.4 Feature aggregation

Profiles of single-cells are aggregated using median at field-of-view (image) level, fields-of-view are aggregated using mean to a well-level profile and, finally, wells are aggregated to treatment level profiles using mean aggregation. The feature aggregation steps are the same for CellProfiler and deep learning features. CellProfiler well-level features with NA values were filtered in the aggregation pipeline.

2.5 Batch correction using sphering transform

One of the possible post-processing methods to reduce batch effects and remove other possible nuisance variations is Typical Variation Normalization (TVN) [18]. Axes of variation are computed using Principal Component Analysis (PCA), then the axes with little variation are amplified and the importance of axes with large variation is reduced by axis normalization. We can control the strength of signal amplification\reduction by using a regularization parameter. The computation at this point involves only profiles of negative controls and as a result, we get the TVN transformation. Then, this transformation is applied to all well-level feature vectors in the analyzed dataset.

Sphering transformation used in this paper, similarly it n well-level profiles of negative controls with vector size d as an input $X^{n\times d}$. Then covariance matrix is calculated $\Sigma = \frac{X^T X}{n}$ and its eigendecomposition is $Q = U\Delta U^T$, where Δ are eigenvalues. Then we divide the diagonal by the square root of eigenvalues in addition with a regularization parameter λ. The final ZCA-transformation [40] matrix is $U\,(\Delta + \lambda)^{-1} U^T$.

The effect of sphering and its regularization on representations and profiling performance is demonstrated in Figure 4.

2.6 Similarity matching

To assess the similarity between resulting treatment profiles cosine similarity is measured for each pair of treatments. Cosine similarity is one of several similarity metrics [12] in profiling and used in [33].

$$\text{cosine similarity} = \frac{A \cdot B}{||A||\ ||B||}$$

3. Deep learning models

3.1 EfficientNet

All deep learning experiments were conducted with EfficientNet [28]. We use the base model EfficientNet-B0 as the size of single-cell crops is only 128x128. It consists of 9 stages: input, seven inverted convolutional blocks from MobileNetV2 [41] (with the addition of squeeze and excitation optimization) and final layers. The usage of convolutional blocks from MobileNetV2 in combination with neural architecture search gave EfficientNet an advantage in terms of computational efficiency and accuracy compared to ResNet50.

EfficientNet was used in biological tasks, for instance, there is an effort to create a model, pre-trained on cellular images called CytoImageNet [42]. Pre-trained EfficientNet was used to extract single-cell embeddings [11] of A549 cells. EfficientNets were also used in cell classification Kaggle competition of Recursion Pharmaceuticals (https://www.kaggle.com/competitions/recursion-cellular-image-classification/), the highest scoring solution based on EfficientNet took 6th place in the private leaderboard.

3.2 Image preprocessing

The raw Cell Painting images are 16-bit TIFF format. To ease the processing for deep learning models, we perform image compression and normalization. The normalization for each image is the following:

- 99.99 percentile of the plate's illumination distribution.
- Compute 0.05 and 99.95 percentiles of illumination distribution in the current image.
- Clip illumination intensity using 0.05 percentile and the minimum of 99.99 percentile of the plate's illumination distribution and 99.95 percentile of illumination distribution in the current image.
- Rescale intensity, create a histogram with 256 bins and save as PNG.

This preprocessing pipeline is included in DeepProfiler and can be run with metadata which lists all the images in the dataset, together with plate, well and site (image or field of view) information.

3.3 Feature extraction with ImageNet pretrained models

One approach adopted to extract representations is to use pre-trained models on a dataset of similar or different domains. Most of the popular existing deep learning architectures are pre-trained on the ImageNet dataset [43], which is composed of natural images.
The idea to use such pre-trained models for morphological profiling starts in the experiments demonstrated in [17] with VGG-16, Inception v3 and ResNet models on the BBBC021 dataset. A similar approach was used in a subset of the lung adenocarcinoma dataset [20] and then in full with pre-trained EfficientNet-B0 [11]. On a large scale, pre-trained models were also used [10], though the dimensionality of extracted features was reduced.

We use DeepProfiler to extract features of single-cells with the pre-trained EfficientNet. The input single-cell images of size 128x128 are cropped out from full images. To fit the expected input size of 224x224, the crops are resized. The pixel values are then rescaled using min-max normalization and adjusted to have values [-1:1] to match the required input range.
As Cell Painting images are five-channel and ImageNet pre-trained models are for three-channel (RGB) images, additional changes are required. For that, we simply replicate each channel three times and pass through the model (so each cell requires five inference passes). Features extracted for each channel are concatenated and the resulting feature vector size is 3360 (672 is the size of the block6a_activation layer in the used EfficientNet implementation).

3.4 Weakly Supervised Learning

Weakly supervised learning (WSL) in this analysis means that we train the models not to classify single-cells by mechanism of action or pathway (that we don't know in a real-world setting), but with auxiliary task of single-cell classification by treatments (in a real-world setting we always know the treatments). Even though we know the treatments, the data is noisy due to several reasons [20]: 1) treatments might not significantly affect the phenotype of the cell (the treatment is neutral or technical problems occurred in the experiment) 2) Applying different treatments might yield cells with similar phenotype 3) Cells might not react uniformly to particular treatments [44], which yields not necessarily meaningful subpopulations of cells.

3.5 Training of Cell Painting models

The Cell Painting models are trained on crops of single-cells, those are obtained from full images using cell locations and full-image metadata. To gather the cropped version of the dataset, we use DeepProfiler, it saves five unfolded channels as a single stripe and generates metadata for single-cells. The stripe can also contain a single-cell segmentation mask so it can be used to filter the crop from neighboring cells and noise (cell masking is not used in the reported analysis). The crop size can be determined by the user and depends on the dataset.

DeepProfiler performs rebalancing of the data for each epoch of training to have an approximately equal number of training examples for each target class (treatment). That is especially important to prevent the overrepresentation of controls and a few treatments during training.

The data augmentation pipeline in DeepProfiler consists of three steps:

1. Random crop and resize. This augmentation is applied with 0.5 probability. The crop region size is random, 80% to 100% of the size of the original image, then resized back to the original size.
2. Random horizontal flips and 90-degree rotations.
3. Brightness and contrast adjustments with built-in Tensorflow methods, each channel is adjusted separately for both augmentation steps.

The parameters for training in all experiments were the following: single-cell crop size 128x128, SGD optimizer, learning rate 0.005, batch size 32, augmentations on, no label smoothing or online label smoothing, 30 epochs of training, the results for all experiments are reported for the last epoch. The model is initialized with ImageNet weights. All models were trained with categorical cross-entropy loss.

We tested two data-split approaches for train and validation subsets for the treatment classification task:

- Leave plates out: we selected the plates in a way that the data of one subset of plates is only the train data and another one is in validation.
- Leave cells out: in all plates present in the training and validation set, we split the data randomly on the well-level, meaning that approximately 60% of cells from each well would be in the training set and the rest in the validation set.

The training of deep learning models was performed on NVIDIA DGX with NVIDIA V100 GPUs; a single GPU was used to train each model.

3.6 Feature extraction with trained Cell Painting models

We use DeepProfiler to extract features, profiling yields one NumPy file (one per image) with an array of vectors (one per cell). DeepProfiler uses full image metadata and location files for feature extraction. As the model is now natively trained for five-channel images, no need to replicate each channel separately, thus, each cell requires one inference pass and the feature vector contains representation of all channels simultaneously. The size of the feature vectors is 672 for reported results, which were extracted from EfficinetNet's block6a_activation layer.

3.7 DeepProfiler

We developed DeepProfiler, a tool for learning and extracting representations from high-throughput microscopy images using convolutional neural networks (CNNs). DeepProfiler proposes a standardized workflow which unifies image pre-processing, training of CNNs and feature extraction discussed in the previous sections. DeepProfiler is implemented in Tensorflow [45] (version 2), the code is publicly available on GitHub (see Code Availability).

4. Evaluation

The evaluation task mostly replicates the one from [34], we check if the most similar treatments above a certain threshold do share the same gene pathway or mechanism of action (MoA). For that, we use modified folds of enrichment metric (based on [34]), in addition we adopt mean average precision and propose a new metric: top hits in the top 1%, which can be measured at different levels of profiling. Features of single-cell are extracted and used for evaluation disregarding their belonging to the training or validation subset of the pretext task.

For folds of enrichment, precision-recall metrics and first-hit on treatment-level we define query and the response as follows:

- Query treatments - the treatment which belongs to an MoA\pathway with at least two treatments, thus it is possible to find a match.
- Response treatments - all the treatments, but the query treatment. Response treatments include the treatments belonging to MoAs\pathways with one treatment.

4.1 Folds of enrichment

For each query treatment we calculate odds ratio in a one-sided Fisher's exact test. The test is calculated using a 2x2 contingency table: the first row contains a number of treatments with the same MoAs\pathways (matches) and different MoAs\pathways at a selected threshold, the second row is the same, but beyond the threshold. Odds ratio is a sum of the first row divided by the sum of the second row. It demonstrates the likelihood of observing the treatment with the same MoA\pathway in the top connections.

We report the mean odds ratio over all query treatments. The threshold we use is 1% of connections. This metric in the text is referred to as "Folds of Enrichment".

The implementation of the metric is available as a part of analysis pipelines (see Code availability section).

4.2 Mean Average Precision

For each query treatment average precision (area under precision-recall curve) is computed for the information retrieval task. The search starts from the most similar treatments to the query and continues until all positive pairs (response treatments with the same MoA\pathway) are found, precision and recall are computed at each step of the search.

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

where TP - true positive, FP - false positive, FN - false negative. Interpolated precision at recall r As the number of treatments per MoA\pathway is not balanced, the precision-recall curve has a different number of recall points; precision and recall are interpolated to cover the maximum number of recall points possible in the dataset. Interpolated precision at each recall point is defined as follows [46]:

$$p_{inter}(r) = max_{r\prime \geq r}\, p(r')$$

Average precision for query treatment is the mean of p_{inter} at all recall points. The reported mean average precision (mAP) is the mean average precision over all queries.

4.3 Hits in the top 1%

The purpose of computational drug discovery is to select a small portion of candidate treatments from the full set of treatments in physical lab experiments. To simulate this, we introduce ***hits in the top 1%*** metric, which measures the number of query treatments which have a treatment of the same MoA/pathway (a hit) in the top 1% of responses (responses are ranked by similarity measure).

This metric is applicable for several levels of profiling: we report results for image level, well level and treatment level. For image-level and well-level we search for an image or well of the same treatment as the treatment of the query image or well. In case of image-level the images belonging to the same well as the query image, are excluded from possible responses. In the well-level analysis, the query well is also removed from possible responses. All treatments in the dataset can be queries on those two levels of profiling.

Acknowledgements

We thank Salil Bhate for the valuable discussions and feedback provided throughout the development of this project.

NM acknowledges the short-term scientific mission grants provided by eCOST action CA15124 (NEUBIAS) in 2019 and 2020. NM and PH acknowledge support from the LENDULET-BIOMAG Grant (2018-342), from the European Regional Development Funds (GINOP-2.2.1-15-2017-00072), from the H2020 and EU-Horizont (ERAPERMED-COMPASS, ERAPERMED-SYMMETRY, DiscovAIR, FAIR-CHARM) from TKP2021-EGA09, from the ELKH-Excellence grants and from the Cooperative Doctoral Programme (2020-2021) of the Ministry for Innovation and Technology. Researchers in the Carpenter–Singh lab were supported by NIH R35 GM122547 to AEC. Researchers in the Cimini lab were supported by NIH P41 GM135019. JCC was supported by the Schmidt Fellowship program of the Broad Institute and by the NSF Award 2134695.

Data availability

Ground truth files for evaluation are available on GitHub https://github.com/broadinstitute/DeepProfilerExperiments.
Link to the pre-trained EfficientNet model https://github.com/Callidior/keras-applications/releases/download/efficientnet/efficientnet-b0_weights_tf_dim_ordering_tf_kernels_autoaugment.h5

The Cell Painting datasets (raw images and CellProfiler profiles) are available at public S3 buckets:
BBBC037 gene overexpression dataset in U2OS cells [30]
s3://cytodata/datasets/TA-ORF-BBBC037-Rohban/profiles_cp/TA-ORF-BBBC037-Rohban/

BBBC022 compound screening in U2OS cells [31]
s3://cytodata/datasets/Bioactives-BBBC022-Gustafsdottir/profiles/Bioactives-BBBC022-Gustafsdottir/

BBBC036 compound screening in U2OS cells [32] s3://cytodata/datasets/CDRPBIO-BBBC036-Bray/profiles_cp/CDRPBIO-BBBC036-Bray/

BBBC043 gene overexpression dataset in A549 cells [11]
s3://cytodata/datasets/LUAD-BBBC043-Caicedo/profiles_cp/LUAD-BBBC043-Caicedo/

LINCS compound screening in A549 cells [5]
s3://cellpainting-gallery/cpg0004-lincs/broad/images/2016_04_01_a549_48hr_batch1/

Code availability

The DeepProfiler code is available on GitHub https://github.com/cytomining/DeepProfiler.
The analysis pipelines (Jupyter notebooks and Python scripts to analyze features) are available on GitHub https://github.com/broadinstitute/DeepProfilerExperiments.
DeepProfiler documentation is available here: https://cytomining.github.io/DeepProfiler-handbook/
In DeepProfiler we used the following EfficientNet implementation: https://github.com/qubvel/efficientnet.

The code and CellProfiler pipelines for three evaluated datasets can be found in the associated GitHub repositories:
BBBC037: https://github.com/carpenterlab/2017_rohban_elife
BBBC036: https://github.com/gigascience/paper-bray2017
BBBC022: Supplementary materials in [31].

References

1. Bray, M.-A. *et al.* Cell Painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes. *Nat. Protoc.* **11**, 1757–1774 (2016).
2. Cimini, B. A. *et al.* Optimizing the Cell Painting assay for image-based profiling. *bioRxiv* 2022.07.13.499171 (2022) doi:10.1101/2022.07.13.499171.
3. Wawer, M. J. *et al.* Toward performance-diverse small-molecule libraries for cell-based phenotypic screening using multiplexed high-dimensional profiling. *Proceedings of the National Academy of Sciences* **111**, 10911–10916 (2014).
4. Cuccarese, M. F. *et al.* Functional immune mapping with deep-learning enabled phenomics applied to immunomodulatory and COVID-19 drug discovery. 2020.08.02.233064 (2020) doi:10.1101/2020.08.02.233064.
5. Way, G. P. *et al.* Morphology and gene expression profiling provide complementary information for mapping cell state. *bioRxiv* 2021.10.21.465335 (2021) doi:10.1101/2021.10.21.465335.
6. Simm, J. *et al.* Repurposing High-Throughput Image Assays Enables Biological Activity Prediction for Drug Discovery. *Cell Chem Biol* **25**, 611–618.e3 (2018).
7. Way, G. P. *et al.* Predicting cell health phenotypes using image-based morphology profiling. *Mol. Biol. Cell* mbcE20120784 (2021).
8. Moshkov, N. *et al.* Predicting compound activity from phenotypic profiles and chemical structures. *bioRxiv* 2020.12.15.422887 (2022) doi:10.1101/2020.12.15.422887.
9. Rohban, M. H. *et al.* Virtual screening for small molecule pathway regulators by image profile matching. *bioRxiv* 2021.07.29.454377 (2022) doi:10.1101/2021.07.29.454377.
10. Schiff, L. *et al.* Integrating deep learning and unbiased automated high-content screening to identify complex disease signatures in human fibroblasts. *bioRxiv* 2020.11.13.380576 (2021) doi:10.1101/2020.11.13.380576.

11. Caicedo, J. C., Arevalo, J. & Piccioni, F. Cell Painting predicts impact of lung cancer variants. *Mol. Biol. Cell* (2022).

12. Caicedo, J. C. *et al.* Data-analysis strategies for image-based cell profiling. *Nat. Methods* **14**, 849–863 (2017).

13. Caicedo, J. C., Singh, S. & Carpenter, A. E. Applications in image-based profiling of perturbations. *Curr. Opin. Biotechnol.* **39**, 134–142 (2016).

14. Chandrasekaran, S. N., Ceulemans, H., Boyd, J. D. & Carpenter, A. E. Image-based profiling for drug discovery: due for a machine-learning upgrade? *Nat. Rev. Drug Discov.* (2020) doi:10.1038/s41573-020-00117-w.

15. McQuin, C. *et al.* CellProfiler 3.0: Next-generation image processing for biology. *PLoS Biol.* **16**, e2005970 (2018).

16. Stirling, D. R. *et al.* CellProfiler 4: improvements in speed, utility and usability. *BMC Bioinformatics* **22**, 433 (2021).

17. Pawlowski, N., Caicedo, J. C., Singh, S., Carpenter, A. E. & Storkey, A. Automating Morphological Profiling with Generic Deep Convolutional Networks. *bioRxiv* 085118 (2016) doi:10.1101/085118.

18. Michael Ando, D., McLean, C. Y. & Berndl, M. Improving Phenotypic Measurements in High-Content Imaging Screens. *bioRxiv* 161422 (2017) doi:10.1101/161422.

19. Schiff, L. *et al.* Integrating deep learning and unbiased automated high-content screening to identify complex disease signatures in human fibroblasts. *Nat. Commun.* **13**, 1590 (2022).

20. Caicedo, J. C., McQuin, C., Goodman, A., Singh, S. & Carpenter, A. E. Weakly Supervised Learning of Single-Cell Feature Embeddings. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* **2018**, 9309–9318 (2018).

21. Lu, A. X., Kraus, O. Z., Cooper, S. & Moses, A. M. Learning unsupervised feature representations for single cell microscopy images with paired cell inpainting. *PLoS Comput.*

Biol. **15**, e1007348 (2019).

22. Hofmarcher, M., Rumetshofer, E. & Clevert, D. A. Accurate prediction of biological assays with high-throughput microscopy images and convolutional networks. *Journal of chemical* (2019).
23. Yang, S. J. *et al.* Applying Deep Neural Network Analysis to High-Content Image-Based Assays. *SLAS Discov* **24**, 829–841 (2019).
24. Mao, C. *et al.* Generative Interventions for Causal Learning. in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2021). doi:10.1109/cvpr46437.2021.00394.
25. Schölkopf, B. *et al.* Toward Causal Representation Learning. *Proc. IEEE* **109**, 612–634 (2021).
26. Rubin, D. B. Estimating causal effects of treatments in randomized and nonrandomized studies. *J. Educ. Psychol.* **66**, 688–701 (1974).
27. Johansson, F., Shalit, U. & Sontag, D. Learning Representations for Counterfactual Inference. in *Proceedings of The 33rd International Conference on Machine Learning* (eds. Balcan, M. F. & Weinberger, K. Q.) vol. 48 3020–3029 (PMLR, 20--22 Jun 2016).
28. Tan, M. & Le, Q. V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *arXiv [cs.LG]* (2019).
29. Becht, E. *et al.* Dimensionality reduction for visualizing single-cell data using UMAP. *Nat. Biotechnol.* (2018) doi:10.1038/nbt.4314.
30. Rohban, M. H. *et al.* Systematic morphological profiling of human gene and allele function via Cell Painting. *Elife* **6**, (2017).
31. Gustafsdottir, S. M. *et al.* Multiplex cytological profiling assay to measure diverse cellular states. *PLoS One* **8**, e80999 (2013).
32. Bray, M.-A. *et al.* A dataset of images and morphological profiles of 30 000 small-molecule treatments using the Cell Painting assay. *Gigascience* **6**, 1–5 (2017).

33. Ljosa, V. *et al.* Comparison of methods for image-based profiling of cellular morphological responses to small-molecule treatment. *J. Biomol. Screen.* **18**, 1321–1329 (2013).

34. Rohban, M. H., Abbasi, H. S., Singh, S. & Carpenter, A. E. Capturing single-cell heterogeneity via data fusion improves image-based profiling. *Nat. Commun.* **10**, 2082 (2019).

35. Chandrasekaran, S. N. *et al.* Three million images and morphological profiles of cells treated with matched chemical and genetic perturbations. *bioRxiv* 2022.01.05.475090 (2022) doi:10.1101/2022.01.05.475090.

36. Otsu, N. A Threshold Selection Method from Gray-Level Histograms. *IEEE Trans. Syst. Man Cybern.* **9**, 62–66 (1979).

37. Jones, T. R., Carpenter, A. & Golland, P. Voronoi-Based Segmentation of Cells on Image Manifolds. in *Computer Vision for Biomedical Image Applications* 535–543 (Springer Berlin Heidelberg, 2005).

38. Voronoi, G. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les parallélloèdres primitifs. *J. Reine Angew. Math.* **1908**, 198–287 (1908).

39. van der Walt, S. *et al.* scikit-image: image processing in Python. *PeerJ* **2**, e453 (2014).

40. Kessy, A., Lewin, A. & Strimmer, K. Optimal Whitening and Decorrelation. *Am. Stat.* **72**, 309–314 (2018).

41. Sandler, Howard & Zhu. Mobilenetv2: Inverted residuals and linear bottlenecks. *Proc. Estonian Acad. Sci. Biol. Ecol.*

42. Hua, S. B. Z., Lu, A. X. & Moses, A. M. CytoImageNet: A large-scale pretraining dataset for bioimage transfer learning. *arXiv [cs.CV]* (2021).

43. Deng, J. *et al.* ImageNet: A large-scale hierarchical image database. in *2009 IEEE Conference on Computer Vision and Pattern Recognition* 248–255 (2009).

44. Gough, A. *et al.* Biologically Relevant Heterogeneity: Metrics and Practical Insights. *SLAS*

Discov **22**, 213–237 (2017).

45. Developers, T. *TensorFlow*. (Zenodo, 2021). doi:10.5281/ZENODO.4724125.

46. Manning, C. D. *Introduction to information retrieval*. (Syngress Publishing, 2008).

E Article (pre-print) ‘Predicting compound activity from phenotypic profiles and chemical structures‘

Predicting compound activity from phenotypic profiles and chemical structures

Authors

Nikita Moshkov, Tim Becker, Kevin Yang, Peter Horvath, Vlado Dancik, Bridget K. Wagner, Paul A. Clemons, Shantanu Singh, Anne E. Carpenter, Juan C. Caicedo

Abstract

Recent advances in deep learning enable using chemical structures and phenotypic profiles to accurately predict assay results for compounds virtually, reducing the time and cost of screens in the drug-discovery process. We evaluate the relative strength of three high-throughput data sources—chemical structures, images (Cell Painting), and gene-expression profiles (L1000)—to predict compound activity using a sparse historical collection of 16,170 compounds tested in 270 assays for a total of 585,439 readouts. All three data modalities can predict compound activity with high accuracy in 6-10% of assays tested; replacing million-compound physical screens with computationally prioritized smaller screens throughout the pharmaceutical industry could yield major savings. Furthermore, the three profiling modalities are complementary, and in combination they can predict 21% of assays with high accuracy, and 64% if lower accuracy is acceptable. Our study shows that, for many assays, predicting compound activity from phenotypic profiles and chemical structures might accelerate the early stages of the drug-discovery process.

Introduction

Drug discovery is very expensive and slow. To identify a promising treatment for specific disease conditions, the theoretical landscape of possible chemical structures is prohibitively large to test in physical experiments. Pharmaceutical companies synthesize and test many millions of compounds, yet even these represent a small fraction of possible structures. Furthermore, although complex phenotypic assay systems have proven extremely valuable for identifying useful drugs for diseases where an appropriate protein target is unknown [1–3], their reliance on expensive or limited-supply biological materials, such as antibodies or human primary cells, often hinders their scalability.

What if computational models could predict the results of hundreds of expensive assays across millions of compounds at a fraction of the cost? Predictive modeling shows some promise. Most attempts so far have used various representations of chemical structure alone to predict assay activity; this requires no laboratory work for the compounds whose activity is to be predicted, and the compounds do not even need to exist physically, so this is dramatically cheaper than physical screens and enables a huge search space. Promising compounds can then be synthesized and tested. Deep learning in particular has substantially advanced the state of the art in recent years [4–17], and was recently used to discover a novel antibiotic [18]. As impressive as these capabilities are, chemical structures alone do not seem to contain enough information to predict all assay readouts — their performance may be limited by the lack of experimental information revealing how living organisms respond to these treatments.

Considerable improvements might come from augmenting chemical structure-based features with biological information associated with each small molecule, ideally information available in inexpensive, scalable assays that could be run on millions of compounds once, then used to predict assay results virtually for hundreds of other individual assays. Most profiling techniques, such as those measuring a subset of the proteome or metabolome, are not scalable to millions of compounds. One exception is transcriptomic profiling by the L1000 assay [19], which has shown success for mechanism of action (MOA) prediction [20], but is untested for predicting assay outcomes.

Image-based profiling is an even less expensive high-throughput profiling technique [21]. It has proven successful in MOA prediction (reviewed in [22]) as well as compound bioactivity determination during structure activity relationship synthetic chemistry cycles [23]. In a novel study, Simm et al. [24] successfully repurposed images from a compound library screen to train machine learning models to predict unrelated assays; their prospective tests yielded 60- to 250-fold increased hit rates while also improving structural diversity of the active compounds. More recently, Cell Painting [25,26] and machine learning have been used to predict the outcomes of other assays as well [27,28].

The complementarity and integration of profiling methodologies and chemical structures to predict compound bioactivity holds promise to improve performance, and has been studied in various ways. The relationships between chemical structures and phenotypic profiles (including cell morphology and transcriptional profiles) has been investigated to predict chemical library diversity [29]. Other studies have looked at combinations of profiles, such as integrating imaging and chemical structures to complete assay readouts in a sparse matrix [30], combining L1000 and Cell Painting for MOA prediction [20], and integrating morphology, gene expression and chemical structure for mitochondrial toxicity detection [31].

In this work, we aim to evaluate the predictive power of chemical structures, cell morphology profiles, and transcriptional profiles, to determine assay outcomes computationally at large scale. This study does not aim to make predictions in specific assays, which may result in anecdotal success, but rather aims to assess the relative potential of data sources for assay prediction, to guide the design of future projects. Our goal is to train machine learning models that predict compound bioactivity taking as input high-dimensional encodings of chemical structures combined with two different types of experimentally-produced phenotypic profiles, imaging (Cell Painting assay) and gene expression (L1000 assay) (Figure 1). Our hypothesis is that data representations of compounds and their experimental effects in cells have complementary strengths to predict assay readouts accurately, and that they can be integrated productively to improve compound prioritization in drug-discovery projects.

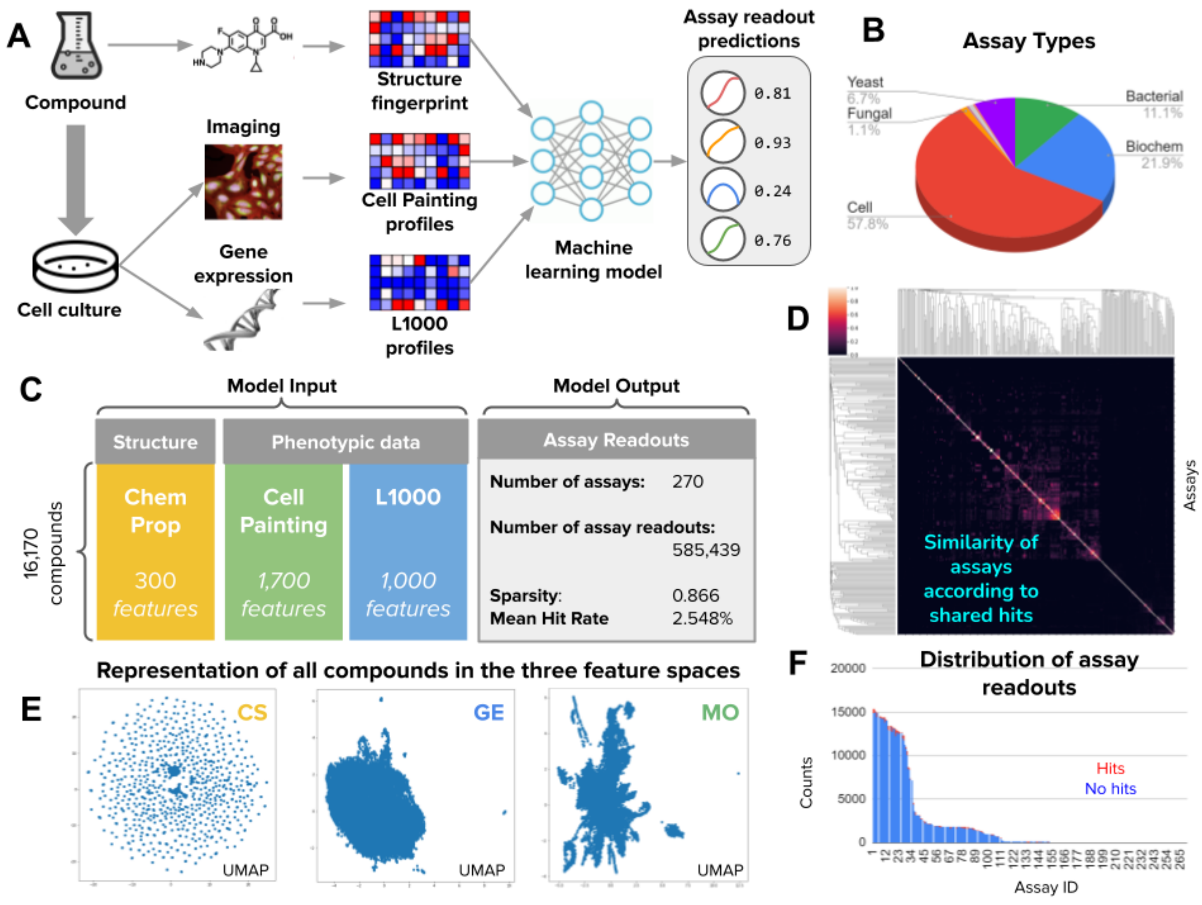


__Figure 1. Overview of the workflow and data.__ A) Workflow of the methodology for predicting diverse assays from perturbation experiments (more details in Supplementary Figures 1 and 2). B) Types of assay readouts targeted for prediction, which include a total of eight categories (Supplementary Figure 14). C) Structure of the input and output data for assay prediction. D) Similarity of assays according to the Jaccard similarity between sets of positive hits. Most assays have independent activity (Supplementary Figure 12). E) UMAP visualizations of all compounds in the three feature spaces evaluated in this study

(Supplementary Figure 9). F) Distribution of assay readouts for assays in the horizontal axis sorted by readout counts. The available examples follow a long tail distribution and the average ratio of positive hits to tested compounds (hit rate) is 2.548%.

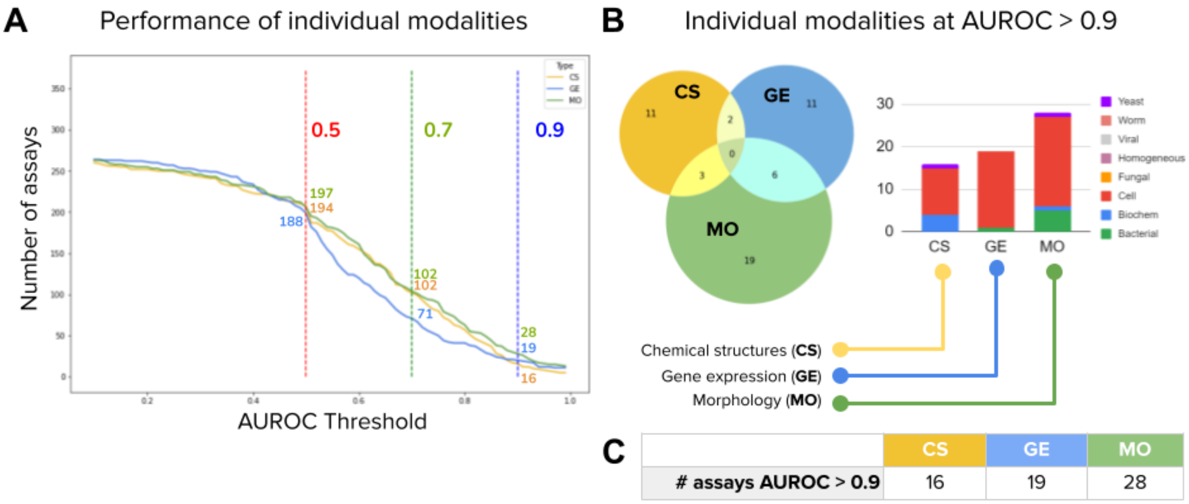
Results

Chemical structure, morphology, and gene expression profiles provide complementary information for prediction

We first selected 270 assays performed at the Broad Institute over more than a decade (Figure 1); the assays were filtered to reduce similarity (Figure 1D) but not selected based on any metadata and thus are representative of the activity of an academic screening center. Then, we extracted a complete matrix of experiment-derived profiles for 16,170 compounds, including gene-expression profiles (GE) from the L1000 assay [32,33] and image-based morphological profiles (MO) from the Cell Painting assay [33,34]. We also computed chemical structure profiles (CS) using graph convolutional nets [16] (Figure 1 and Methods). Finally, assay predictors were trained and evaluated following a 5-fold cross-validation scheme using scaffold-based splits (Methods and Supplementary Figures 1 and 9). This evaluation aims to quantify the ability of the three data modalities to independently identify hits in the set of held-out compounds (which had compounds of dissimilar structures to the training set, to prevent learning assay outcomes for highly structurally similar compounds).

We found that all three profile types (CS, GE, and MO) can predict different subsets of assays with high accuracy, revealing a lack of major overlap among the prediction ability by each profiling modality alone (Figure 2B). This indicates significant complementarity, that is, each profiling modality captures different biologically relevant information. In fact, only 11 of the 270 assays “overlapped” and were predictable using more than one of the single modalities, and none could be accurately predicted by all three of the single profiling modalities (median overlap over 5-fold cross validation is zero). CS shares three well-predicted assays in common with MO and two with GE, while MO and GE share six, indicating that CS captures slightly more independent activity. MO profiles predicted 19 assays that are not captured by chemical structures or gene expression alone, the largest number of *unique* predictors among all modalities (Figure 2B).

MO is able to predict the largest number of assays individually (28 vs 19 for GE and 16 for CS) (Figure 2C), although if a lower accuracy threshold is sufficient (AUROC > 0.7), CS can predict around the same number of assays as MO, while GE still trails (Figure 2A). We use the count of predictors with AUROC > 0.9 as our primary evaluation metric, following past studies of assay prediction [18,20,24], although 0.7 is not unreasonable in practice; one would need to cherry pick more compounds to obtain sufficient hits in followup testing. The results in Figure 2 reveal the extent to which profiling modalities capture specific bioactivity and confirm that they are indeed mostly different from each other.



	CS	GE	MO
# assays AUROC > 0.9	16	19	28

Figure 2. *Number of assays that can be accurately predicted using single profiling modalities. All reported numbers are the median result of the five-fold cross-validation experiments run in the dataset. Detailed results of each partition are reported in Supplementary Figure 4 and Supplementary Table 1. A) Performance of individual modalities measured as the number of assays (vertical axis) predicted with AUROC above a certain threshold (horizontal axis). With higher AUROC thresholds, the number of assays that can be predicted decreases for all profiling modalities. We define accurate assays as those with AUROC greater than 0.9 (dashed vertical line in blue). B) The Venn diagrams on the right show the number of accurate assays (median AUROC > 0.9) that are in common or unique to each profiling modality. The bar plot shows the distribution of assay types correctly predicted by single profiling modalities. C) Number of assays well predicted (median AUROC > 0.9) by each individual modality (same as in Figure 3B).*

Combining phenotypic profiles with chemical structures improves assay prediction ability

Ideally, combining modalities should leverage their strengths and predict more assays jointly, by productively integrating data. Morphology and gene-expression profiles require wet lab experimentation, whereas chemical structures are always available, even for theoretical compounds, with the only cost being computing their fingerprints. Therefore, we took CS as a baseline and explored the value of adding phenotypic profiles to it.

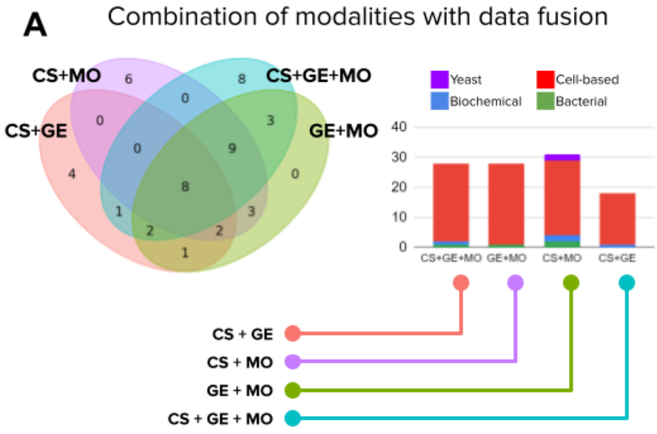
We first integrated data from different profiling methods using late data fusion and evaluated the performance of combined predictors using the same 5-fold cross validation protocol described for individual profiling modalities. We found that adding morphological profiles to chemical structures yields 31 well-predicted assays (CS+MO) as compared to 16 assays for CS alone (Figure 3C). By contrast, adding gene expression profiles to chemical structures by late data fusion increased the number of well-predicted assays as compared to CS alone only by two assays (18 vs 16 respectively, Figure 3C). For both phenotypic profiling modalities, early fusion (concatenation of features before prediction) performed worse than late fusion (integration of probabilities after separate predictions, see Methods), yielding fewer predictors with AUROC > 0.9 for all combinations of data types (Supplementary Figure 8 and Supplementary Table 1). The results represent an opportunity for enhancing computational fusion strategies (see Methods - Data fusion).

Next, we counted the number of unique assays predicted by *any* of the individual profiling modalities using a *retrospective* assessment, which estimates the performance of an ideal data fusion method that perfectly synergizes all modalities. Note that this retrospective assessment is not blind, and simulates a decision maker that chooses the best predictor for an assay after looking at their performance in the hold-out set. It is used here to report the total number of assays that can be successfully predicted using one or another strategy. For example, we found that using the best profiling modality from a given pair can predict around 40 assays (Figure 3D, row “Single”). We use the ★ symbol to denote choosing the best between profiling modalities in retrospect, and the + symbol to denote combining modalities by data fusion.

In retrospect, there are six unique assays that are well predicted using fused CS+MO that could not be captured by either modality alone, indicating complementarity to improve performance for these five assays. Adding them to the list of assays that can be predicted using the single best from CS★MO would yield 41 well-predicted assays total (Figure 3C, row “Plus fusion”), resulting in potential to predict more than twice the assays compared to CS alone (16). Improvements when adding MO to CS were consistently found across other evaluation metrics (AUROC > 0.7 in Supplementary Figure 3 and Supplementary Table 1) and when adding morphological profiles to all other data types and combinations (Figure 3D).

At an AUROC > 0.9, the 41 unique assays that are well predicted with CS★MO represent 13% of the total. An AUROC of 0.7 could be acceptable to find useful hits in real world projects [18,24]; we found that for assays with a low baseline hit rate, this accuracy level may be sufficient to increase the ability to identify useful compounds in the screen (Supplementary Figure 3). If a cutoff of AUROC > 0.7 was found to be acceptable, 58% of assays would be well predicted with CS★MO (157 out of 270, Supplementary Figure 3).

The performance of CS★GE also increased the number of assays that CS can predict alone from 16 to 33 at AUROC > 0.9. There are four more assays that are well predicted using fused CS+GE, which results in 37 unique assays well predicted by both modalities in retrospect. Gene expression also yields similar results when combined with morphology, yielding 41 assays with GE★MO, and predicting seven additional assays jointly when using data fusion (GE+MO) for a total of 48 unique assays together.



B Performance of individual modalities

	CS	GE	MO
# assays AUROC > 0.9	16	19	28

C Performance of data fusion by cross-validation (predictive)

	CS+MO	CS+GE	GE+MO	CS+GE+MO
# Median AUC > 0.9	31	18	28	28

D Retrospective analysis of predictors with AUROC > 0.9 (oracle counts)

	CS★MO	CS★GE	GE★MO	CS★GE★MO
Single	41	33	41	52
Plus fusion	44	37	48	57

Figure 3. *Number of assays that can be accurately predicted using combinations of profiling modalities. Accurate predictors are defined as models with accuracy greater than 0.9 AUROC. We considered all four modality combinations using late data fusion in this analysis: CS+MO (chemical structures and morphology), CS+GE (chemical structures and gene expression), GE+MO (gene expression and morphology), and CS+GE+MO (all three modalities). A) The Venn diagram shows the number of accurately predicted assays that are in common or unique to fused data modalities. The bar plots in the center show the distribution of assay types correctly predicted by the fused models. All counts are the median of results in the holdout set of a five fold cross-validation experiment. B) Performance of individual modalities (same as in Figure 2C). C) The number of accurate assay predictors (AUROC > 0.9) obtained for combinations of modalities (columns) using late data fusion following predictive cross-validation experiments. D) Retrospective performance of predictors using oracle counts. These counts indicate how many unique assays can be predicted with high accuracy (AUROC > 0.9), either by single or fused modalities. "Single" is the total number of assays reaching AUROC > 0.9 with any one of the specified modalities, i.e., take the best single-modality predictor for an assay in a retrospective way. This count corresponds to the simple union of circles in the Venn diagram in Figure 2B, i.e., no data fusion is involved. "Plus fusion" is the same, except that it displays the number of unique assays that reach AUROC > 0.9 with any individual or data-fused combination. This count corresponds to the union of circles in the Venn diagram in Figure 2B plus the number of additional assays that reach AUROC > 0.9 when the modalities are fused. For example, the last column counts an assay if its AUROC > 0.9 for any of the following: CS alone, GE alone, MO alone, data-fused CS+GE, data-fused GE+MO, data-fused CS+MO, and data-fused CS+GE+MO.*

Complementarity across all three profiling types

We had hypothesized that data fusion of all three modalities would provide the best assay prediction ability than any individual or subset. However, data-fused CS+GE+MO yielded 28 well-predicted assays (Figure 3C), fewer than could be obtained by data-fused CS+MO (31 assays), which was the same as MO alone (28 assays). All of these fall short of the 52 unique assays that, in retrospect, could be identified by taking the single best of any of the three data types CS★MO★GE (Figure 3D). This highlights the need for designing improved strategies for data fusion; early fusion did not improve the situation (Supplementary Figure 8 and Supplementary Table 1).

Likewise, considering the best single, pairwise and all-fused predictors and their combinations, the three data modalities have the potential to accurately predict 57 assays jointly at 0.9 AUROC, not a dramatic improvement compared to 52 unique assays that, in retrospect, could be identified by taking the single best of any of the three data types using CS★MO★GE (Figure 3D). Nevertheless, 57 assays represents 21% of the 270 assays considered in this study. With a threshold of 0.7 AUROC (Supplementary Figure 3), the three modalities can predict 117 assays using data fusion (43% of all 270), and with their retrospective combinations the list grows to 174 assays (64% of all 314). We therefore conclude that if all modalities are available, they are all useful to increase predictive ability, as they appear to capture different aspects of perturbed cell states.

Models can predict a diversity of assay types

The morphological and gene-expression profiles used for model training derive from cell-based profiling assays. They can correctly predict compound activity for mammalian cell-based assays, which were the most frequent in this study (Figure 1B, Supplementary Figure 14), but also other assay types, such as bacterial and biochemical (Figure 2B, 3A, Supplementary Figure 14, Supplementary Table 3). Still, cell-based assays were the best-predicted by the phenotypic profiles as well as by chemical structures: from 156 cell-based assays, 11, 18 and 21 are accurately predicted by CS, GE and MO respectively (7%, 11%, 13%); by contrast, from 59 biochemical assays, 4, 0 and 1 were predicted by CS, GE and MO respectively (6%, 0%, 1.7%).

We nevertheless conclude that well-predicted assays include diverse assay types, i.e., phenotypic profiling strategies are not constrained to predict cell-based assays only, even though both profiling methods are cell-based assays themselves. Each modality predicted assays in 2-4 of the 8 assay categories when used alone (Figure 2B).

As noted above, only a few assays benefit from combining information of various profiling modalities. We examined four assays with increased fused accuracy more closely (Figure 4). The *CFTR activity* assay, a cell-based assay, can be predicted with an AUROC of 0.88 using CS alone, but when combined with MO using data fusion, the performance increases to AUROC 0.97. Similarly, the *Ras selective lethality assay* reaches a maximum accuracy of 0.69 using CS alone, but when MO and GE are combined together, accuracy increases to 0.90 AUROC, increasing performance from low to highly accurate. These two assays have rare hits and benefit more from data fusion, compared to the other two examples in Figure 4 (*esBAF inhibitor* and *SirT5 activity*) which also benefit from data fusion but to a lesser degree (e.g. increasing performance from 0.79 to 0.83). These examples indicate that fusing information from various modalities can improve predictive performance, but the fusion result may depend on several factors such as the diversity and availability of training examples and the biology measured by the specific assay.

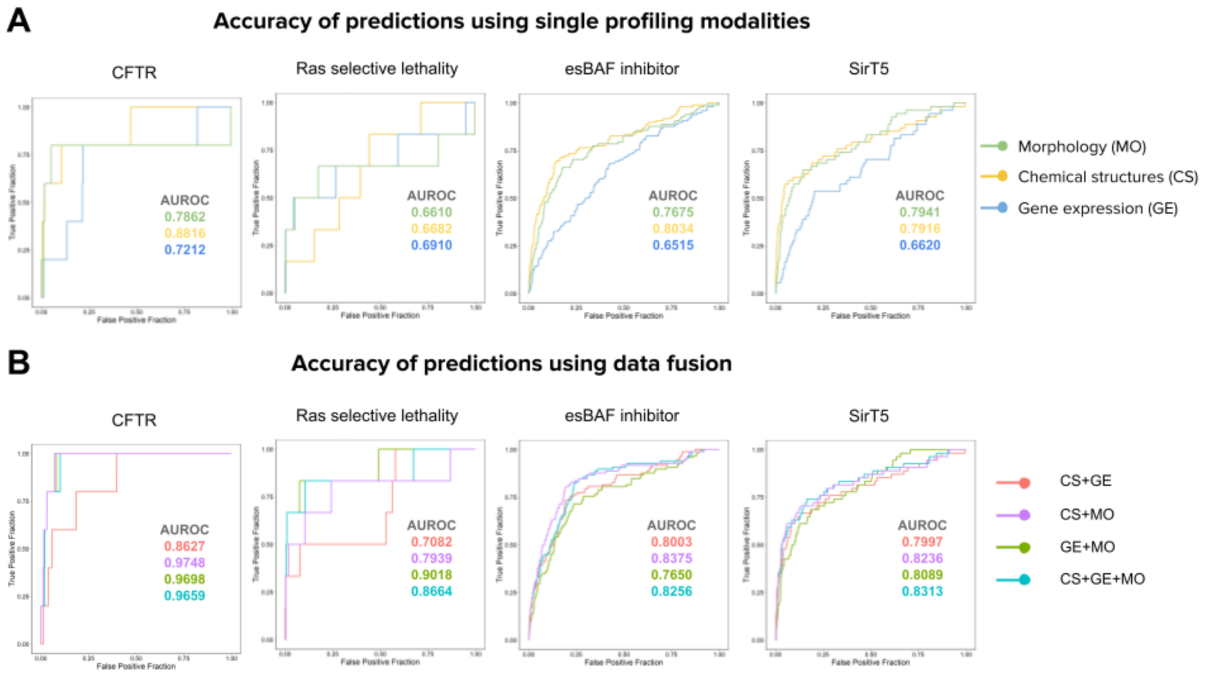


***Figure 4**. Prediction performance of example assays where prediction accuracy benefits from fusion. The plots are Receiver Operating Characteristic (ROC) curves and the area under the curve (AUROC) is reported for each modality with the corresponding color. A) Four example assays from left to right: Cystic fibrosis transmembrane conductance regulator CFTR (cell based), Ras selective lethality (cell based), esBAF inhibitor (cell based), SirT5 (biochemical). B) Performance of predictors for the same assays when using combinations of profiling methods.*

Assay predictors trained with phenotypic profiles can improve hit rates

Predictive modeling using machine learning to reuse phenotypic profiles in a large library of compounds can enable virtual screening to identify candidate hits without physically running the assays. Here, we compare the hit rate of testing only the top predicted candidates obtained with a computational model, vs the empirical hit rate of testing a large subset of candidate compounds physically in the lab (Supplementary Figure 6). The ratio between these two hit rates is what we term *folds of improvement*, a factor indicating the expected experimental efficiency if the computational model identifies relevant compounds to test in follow up experiments.

We found that predictors meeting AUROC > 0.9 in our experiments produce on average a 25 to 70-fold improvement in hit rate (i.e., compounds with the desired activity, see Supplementary Figure 7) for assays with a baseline hit rate below 1%. A baseline hit rate below 1% means that hits are rare for such assays, i.e., in order to find a hit we need to test at least 100 compounds randomly selected from the library. Assays with low hit rates are the goal in real world screens, and therefore, more expensive to run in practice. With computational predictions improving hit rates by ~50 fold, the speed and return of investment could be potentially high. We also note that for assays with less extreme baseline hit rates (e.g. 10% to 50%), the machine learning models can reach the theoretical maximum fold of improvement by accurately predicting all the hits in the top of the list (Supplementary Figure 7). We conclude that when assay predictors are accurate enough, they make cherry-picking and testing predicted compounds worthwhile and can thus significantly accelerate compound screening and reduce the resources required to identify useful hits.

Discussion

Predicting bioactivity of compounds could become a powerful strategy for drug discovery in light of ever-improving computational methods (particularly, deep learning) and ever-increasing rich data sources (particularly, from profiling assays). Here, we used the Chemprop model for learning predictors from chemical structures, and to combine the molecular fingerprint with phenotypic profiles obtained from images (Cell Painting) and gene expression (L1000). We conducted this study using baseline feature representations, and arguably, the results could be improved in future research by using alternative chemical structure embeddings [35–37], learned image features [27,38], or latent spaces for gene expression [39].

We discovered that all three profile modalities—chemical structure, morphology, and gene expression—offer independently useful information about perturbed cell states that enables predicting different assays. Chemical structure is always readily available for a given compound. The two profiling modalities that require physical experimentation bring different strengths to the assay prediction problem, and if available, they can be leveraged to run virtual screens to prioritize compound candidates in drug-discovery projects.

In retrospect, we found that data fusion strategies increased the number of well-predicted assays by only 7-17%, depending on the subset of modalities tested, as compared to simply using each profiling modality independently for prediction. We believe this argues for further research on how best to integrate disparate profiling modalities, capturing the strengths of each individually as well as the complementarity of their combinations. Nevertheless, using late data fusion to combine each subset of available modalities does offer some improvement versus each individually and is likely worthwhile given its ease of implementation.

Given the low cost of carrying out Cell Painting, it is practical in many settings to profile an entire institution's compound library. Then, a modest-sized library of a few thousand compounds would be tested in each new assay of interest, providing sufficient data to assess whether an accurate predictor could be trained on these data, using CS alone, MO alone, or a data-fused combination of CS+MO. Taking into account the baseline hit rate for the assay, researchers could decide whether the predictor will increase the hit rate sufficiently to warrant a virtual screen against a large compound library for which morphological profiles are already available (within an institution, or publicly available profiles [40]), followed by cherry picking a small set of predicted hits and testing them for actual activity in the assay.

Although we suggest a few thousand compounds for the training set based on the data shown in Supplementary Figure 5, it remains to be fully evaluated how many training points are needed to achieve strong predictivity — in fact, it is likely that the number and structural diversity of hits in the training set more strongly influences predictivity than the total number of assay data points. Nevertheless, in most academic and industry screening centers, preparing a training/test set of ~16,000 compounds, as we used here, is practical. It also remains to be explored what determines if an assay is likely to be predictable, such as the target and assay type (unavailable for this dataset), and characterizing correlations between the bioactivity of interest and profiling modalities, as well as the assay activity distribution.

Based on our results, and depending on whether an AUROC of 0.9 or 0.7 is the threshold for accuracy needed given the baseline hit rate of the assay, 21-64% of assays should be predictable using a combination of chemical structures, morphology and gene expression, saving the time and expense of screening these assays against a full compound library. Especially considering potential improvements in data integration techniques and deep learning for feature extraction, this strategy might accelerate the discovery of useful chemical matter.

Methods

Profiling datasets

For this study, we began with a compound library of over 30,000 compounds screened in high-throughput [33]. Of these compounds, about 10,000 came from the Molecular Libraries Small Molecules Repository, another 2,200 were drugs and small molecules, and the remaining 18,000 were novel compounds derived from diversity oriented synthesis. U2OS cells were plated in 384-well plates and treated with these compounds in 5 replicates, using DMSO as a negative control. The Cell Painting and L1000 platforms were used to generate morphological and transcriptional profiling data, respectively, as previously described [33].

Assay readouts

We collected a list of 529 assays from drug discovery projects conducted at the Broad Institute at different scales, and we kept those where at least a subset of the small molecules in the compound library described above was tested. After administrative filtering and metadata consistency, we kept a subset of 496 candidate assays for this study. We prepared assay performance profiles following a double sigmoid normalization procedure to ensure that all readouts are scaled in the same range [41]. Then, we computed the Jaccard similarity of hits between pairs of assays to estimate the common set of compounds detected by them, and then removed assays that measure redundant compound activity (Supplementary Figure 12). That resulted in a final list of 270 assays with their corresponding readout results (Supplementary Figure 11), and the compound-assay matrix had 13.4% of known entries (86.6% sparsity).

Training / Test splits

The total number of compounds in the library that had the three types of information required to conduct the experiments in our project (Cell Painting images, L1000 profiles, and assay readouts) was 16,978. We applied all pan-assay interference (PAINS) filters [42] implemented in RDKit, which removed 786 compounds, resulting in 16,210 compounds. Next, we removed all assays without hits reducing the set of candidate assays from 496 to 437. Then, we calculated the Jaccard score between assay hits to identify redundant assays, i.e., assays that measure similar activity resulting in the same hits. The Jaccard similarity matrix (437x437) was thresholded at 0.7 to remove highly redundant assays, and hierarchical clustering with the cosine distance metric was applied for determining further groups of redundant assays. Finally, we removed frequent hitters, defined as compounds that are positive hits in at least 10% of the assays (by being hits in 30 assays or more) and an additional step of removing assays that remain without any hit. In the end, the final dataset consists of 16,170 compounds and 270 assays.

We aimed to evaluate the ability of each data modality to predict assays for chemical structures that are *distinct* relative to training data. This is because there is little practical value to screen for additional, similar structures (scaffolds) to compounds already known to have activity; in drug discovery, any compounds with positive activity undergo medicinal chemistry where small variations in structure are synthesized and tested to optimize the molecule. We therefore report results using cross-validation partitions that ensure that similar classes of structures are not included in both the training and hold-out sets, given that this scheme corresponds to the most practical, real world scenario (Supplementary Figure 9).

We used 5-fold cross-validation using Bemis-Murcko clustering [43,44], and assigned clusters to training or test in each fold accordingly. The main experimental design for the results reported in the main text is illustrated in the Supplementary Figure 1. The distribution of chemical structure similarity according to the Tanimoto coefficient metric on Morgan fingerprints (radius=2) is reported in Supplementary Table 10 for each of the 5 cross-validation groups. As additional control tests, we run 5-fold cross-validation experiments following the same design as above but splitting the data according to k-means clusters in the morphology feature space and in the gene-expression space (Supplementary Figure 9 and Supplementary Table 2), as well as a control experiment with fully random splits (Supplementary Table 2).

The control splits based on randomized data as well as the MO and GE modalities were used to check for and identify potential biases in the data. These splits do not have practical applications in the lab, and were used as computational simulations to test the alternative hypothesis that predictors have a disadvantage when the training data are drawn from a distribution that follows similarities in CS, MO or GE. The results in the Supplementary Table 2 indicate that there is no major change in performance when using CS, GE or random splits; however, MO splits reduced performance significantly for all data modalities. This process revealed the need to correct for batch effects in MO data to minimize the influence of technical artifacts. All results presented in the main text were obtained from MO data that has been batch corrected (see Image-based morphological profiles below).

Representation of chemical structures (CS) using Chemprop

We used the Chemprop software (http://chemprop.csail.mit.edu/) to train directed-message passing neural networks for learning chemical structure embeddings. The software reconstructs a molecular graph of chemicals from their SMILES string representation, where atoms are nodes and bonds are edges. From this graph, a model applies a series of message passing steps to aggregate information from neighboring atoms and bonds to create a better representation of the molecule. For more details about the model and the software, we refer the reader to prior work [16,18,45]. In addition to learning representations for chemical structures, we used Chemprop to run all the machine learning models evaluated in this work to base all the experiments on the same computational framework. Also, we evaluated the predictive models for CS using learned features as well as Morgan fingerprints computed with the RDKit software (radius=2), and we found that both yield comparable results in our main experiments (Supplementary Table 2, columns CS-GC [Graph Convolutions] and CS-MF [Morgan Fingerprints]).

The representation of chemical structures is learned from the set of ~13,000 training examples, unlike morphological or gene-expression features, which were obtained without learning methods (hand-engineered features). The scaffold split used in our experiments may pose an apparent disadvantage to the learning of chemical structure representations because it may not learn to represent important chemical features in new scaffolds. Previous research by Yang et al. [16] has shown that Chemprop can generalize to new scaffolds accurately. In addition, the chemicals may also generate new phenotypes in the morphological and gene-expression space, which are not seen by the models during training, resulting in a fair comparison of representation power among all modalities. We tested the effect of creating partitions with other modalities other than scaffolds from chemical structures, and we discuss these results in the Train / Test splits subsection above as well as in Supplementary Table 2 and Supplementary Figure 9.

Image-based morphological (MO) profiles from the Cell Painting assay

The Cell Painting assay [25,26,29,33] captures fluorescence images of cells using six dyes to label eight major cell compartments. The five-channel, high-resolution images are processed using the CellProfiler software (https://cellprofiler.org/) to segment the cells and compute a set of

1,700+ morphological features at the single-cell level. These features are aggregated into well- and treatment-level profiles that capture the central statistics of the response of cells to the treatment.

Before computing treatment-level profiles, we used the Typical Variation Normalization (TVN) [46] transform to correct for batch effects using well-level profiles (see Supplementary Figure 9). TVN is calculated using DMSO control wells from all plates to compute a sphering transform that reduces the data to a white noise distribution by inverting all the non-zero eigenvalues of the matrix. This transformation is later used to project all treatment wells in a new space, where controls have a neutral representation and treatments may have phenotypic variations highlighted. This transform minimizes batch effects by obtaining a feature space where the technical variations sampled from controls are neutralized to enhance the biological signal.

After applying the TVN transform at the well-level profiles, we aggregate them into treatment-level profiles to conduct our assay prediction experiments. Supplementary Figure 8 shows UMAP plots of the morphology data before and after the TVN transformation. In our study, we used treatment-level profiles in all experiments. For more details about Cell Painting [26], CellProfiler [47], and the profiling steps [48], see the corresponding references.

Gene-expression (GE) profiles from the L1000 assay

The L1000 assay measures transcriptional activity of perturbed populations of cells at high-throughput. These profiles contain mRNA levels for 978 landmark genes that capture approximately 80% of the transcriptional variance [19]. The assay was used to measure gene expression in U2OS cells treated with the set of compounds in our library. Both the profiles and the tools to process this information are available at https://clue.io/ .

Predictive model

Model architecture: The predictive model is a feedforward, fully connected neural network with up to three hidden layers and ReLU activation functions. This simple architecture takes as input compound features (or phenotypic profiles) and produces as output the hit probabilities for all assays (see Supplementary Figure 8). When the representation of chemical structures is learned, additional layers are created before the predictive model to compute the message passing graph convolutions. These extra layers and their computation follow the default configuration of Chemprop models [16] and are only used for chemical structures.

Loss and training: The model architecture described above is trained in a multi-task manner [5], allocating a binary output for each assay. We used the logistic regression loss function on each assay output and the total loss is the sum over all assays. During training, the model computes this loss for each assay output independently using the available readouts. If the assay readout is not available for some compounds in the mini-batch, these outputs are ignored and not taken into account to calculate gradients. This setup facilitates learning predictive models with sparse assay readouts. We use a mini-batch size of 50 compounds with a sparse matrix of 270 labels, and no explicit class balancing was applied during training.

Hyperparameter optimization: The hyperparameters of the network are optimized on the training data for each feature grouping and for each cross-validation round. These parameters are: number of fully connected layers (choice between 1, 2 or 3), dropout rate for all layers (between 0 and 1), and hidden layer dimensionality (if applicable, between 100 and 2,500). The best parameters are identified by further splitting the training set in three parts, with proportions 80% for training, 10% for validation and 10% for reporting hyperparameter optimization performance. Then, these parameters are used to train a final model that is used to make predictions in the hold-out partition of the corresponding cross-validation set.

Data fusion

The input to the neural network can be the features of one or all modalities used in our experiments. To combine information from multiple data modalities, we used two strategies (Supplementary Figure 8): A) early data fusion, where feature vectors from two or three modalities are concatenated into a single vector. B) Late data fusion, where each modality is used to train a separate model, and then the prediction scores for a new sample are aggregated using the maximum operator. Our results show that, despite its simplicity, late data fusion works best in practice (see Supplementary Table 2), but the results also suggest that more research needs to be done to effectively combine multiple data modalities.

Combining disparate data modalities (sometimes called multimodal or multi-omic data integration) is an unmet computational challenge especially when not all the assays can be accurately predicted. Our results indicate that the three data modalities do not predict any assays in common (Figure 2B, no assays are predicted by all modalities when used independently), suggesting that in most cases, at least one of the data modalities will effectively introduce noise for predicting a given assay. When one of the data modalities cannot signal the bioactivity of interest, the noise-to-signal ratio in the feature space increases, making it more challenging for predictive models to succeed. This explains why late fusion, which independently looks at each modality, tends to produce better performance.

Performance metrics

To evaluate the performance of assay predictors we used the area under the receiving operating characteristic (ROC) curve, also known as the AUROC metric, which has a baseline random performance of 0.5. During the test phase, we run the model over all compounds in the test set to obtain their hit probabilities for all assays. With these probabilities, we compute AUROC for each assay using only the compounds that have ground truth annotations (either positive hits or negative results), and we ignore the rest of the compounds that have no annotation for that assay (unknown result or compound never tested).

We define a threshold of AUROC > 0.9 to identify assays that can be accurately predicted, and with this threshold, our second performance metric is focused on counting how many assays, from the list of 270 in our study, can be accurately predicted. For comparison, we also calculated Average Precision (AP) and area under the precision-recall curve (AUPRC) which are reported in Supplementary Tables 1, 2 and 3.

In addition, we measured hit-rate improvement for individual assays as the ratio between the hit rate obtained using the computational predictors and the hit rate observed in the lab (the “baseline” hit rate):

$$Improvement\ = \frac{Predictor\ Hit\ Rate}{Baseline\ Hit\ Rate}$$

Predictor hit rates are calculated as the proportion of positive hits observed in the top 1% of the ranked list of predictions, while *baseline hit rates* are calculated as the number of hits identified in the complete set of compounds tested for that assay in the original experiment. For an illustration of this performance metric see Supplementary Figure 6 and Supplementary Figure 7 for the results.

Data and code availability

The morphological and gene-expression profiles were originally created and published by Wawer, M. J. et al. [33], and can be downloaded from: http://www.broadinstitute.org/mlpcn/data/Broad.PNAS2014.ProfilingData.zip

The Cell Painting images were also made available by Bray et al. [34], and can be obtained from the following link: http://gigadb.org/dataset/100351

The latest version of morphological profiles is also available in the following AWS S3 bucket: https://registry.opendata.aws/cell-painting-image-collection/

The Chemprop software and source code used for training machine learning models can be found in the following link: http://chemprop.csail.mit.edu/

The analysis code to reproduce the experiments reported in the paper can be found in the following link: https://github.com/carpenterlab/puma_project

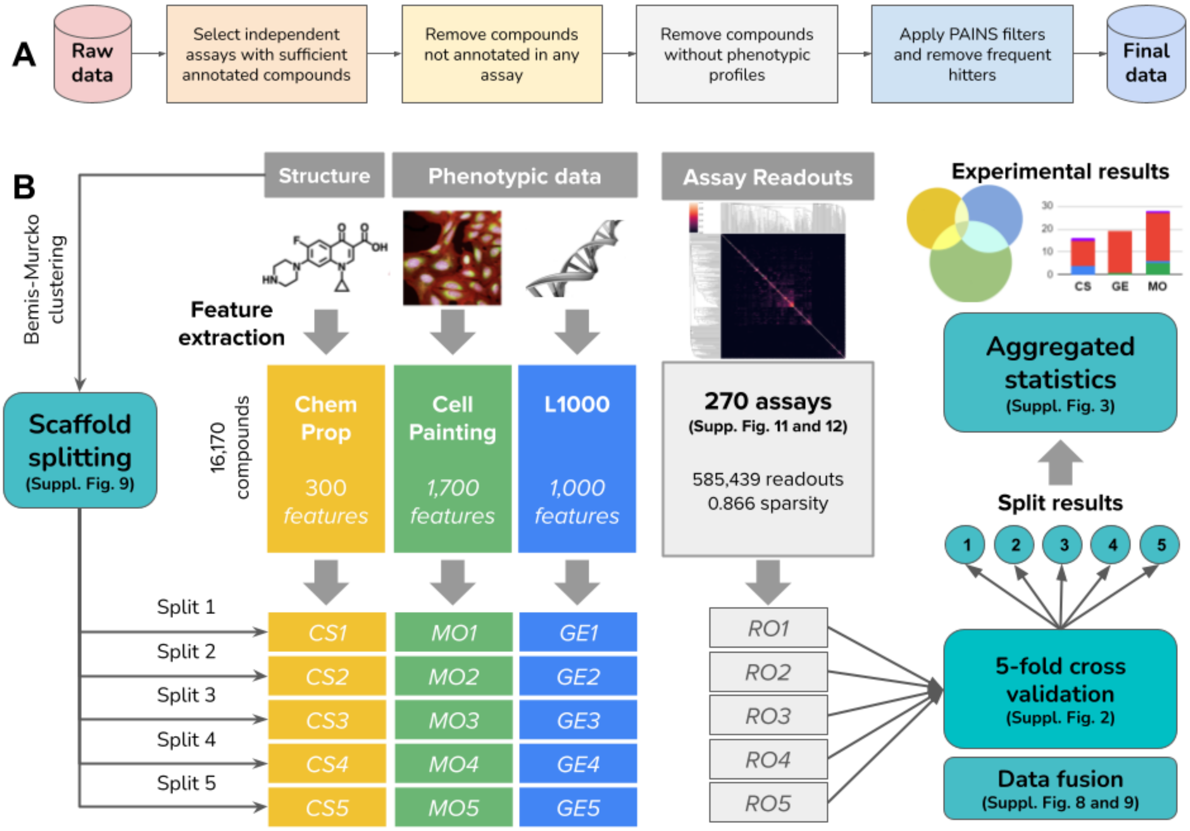
The assay data to reproduce the analysis in the paper is available in the project GitHub repository: https://github.com/carpenterlab/puma_project/tree/main/data

Acknowledgements

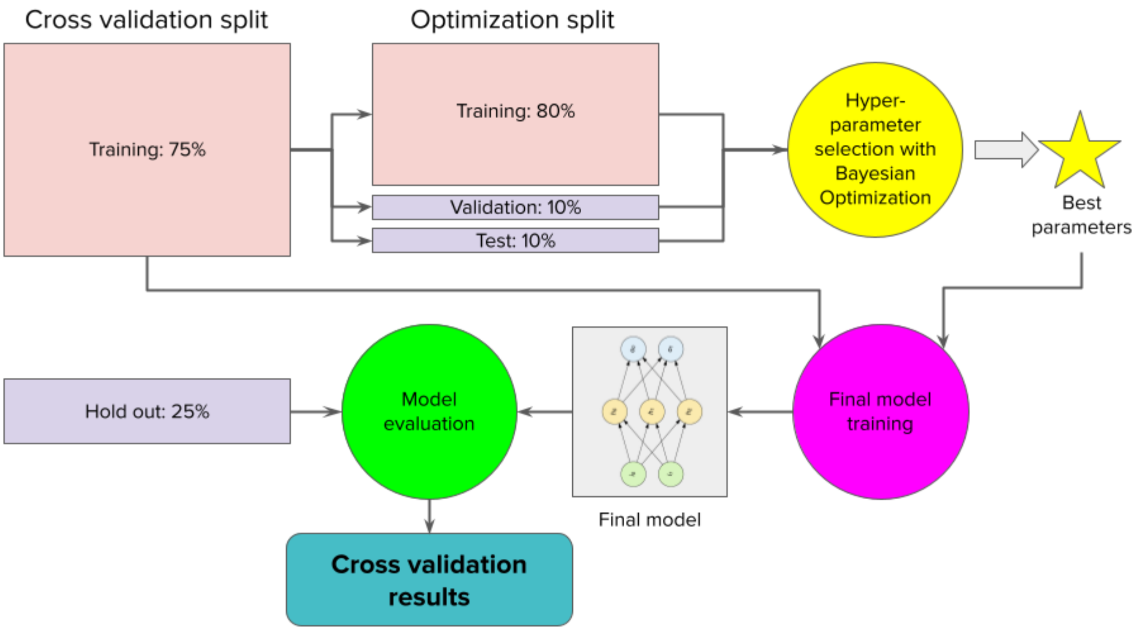
This study was supported by a grant from the National Institutes of Health (R35 GM122547 to AEC), and by the Broad Institute Schmidt Fellowship program (JCC). The authors are grateful for guidance and thoughtful discussions with Regina Barzilay and Tommi Jaakkola which improved the analysis and manuscript.

Supplementary Material

Experimental design

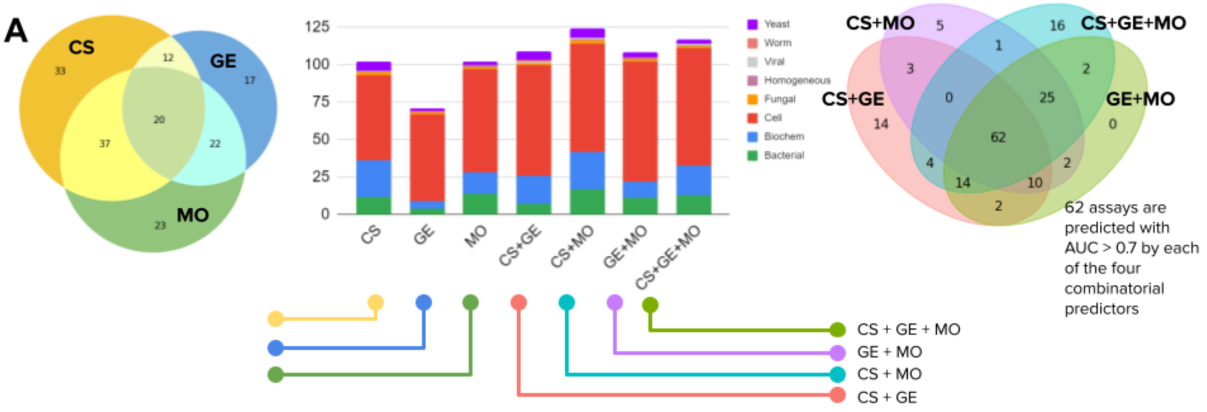


Supplementary Figure 1. Illustration of the experimental design in this study. A) Data selection and filtering pipeline to construct the dataset used in this study. The process is linear and the order of steps is followed one at a time. We first select 270 assays from more than 500 available (see Supplementary Figure 11 and 12), and with those targets fixed, we proceed to clean the list of compounds with various other filters. B) We considered the problem of assay prediction from three compound representations: features of the chemical structure, and phenotypic features of the effect of compounds measured by imaging (Cell Painting) and gene expression (L1000). We conducted a 5-fold cross-validation experiment splitting the compounds in 5 groups according to scaffold similarity using the Bemis-Murcko clustering. The profiles for compounds in each of these groups were separated together with the corresponding assay readouts. The training of models and test of predictions is carried out independently for each fold, and the results are aggregated to generate summarized statistics of the experimental results.



Supplementary Figure 2. Pipeline of cross-validation experiments. The models trained and evaluated in our experiments are conducted following this protocol: for each split in the 5-fold validation scheme, we take the training dataset and split it again in three parts: 80% for training, 10% for validation and 10% for testing. In this partition, we run hyperparameter search using Bayesian optimization to calibrate the parameters described in the Methods section, subsection Predictive model and data fusion. The Bayesian optimization model uses the 10% assigned for validation to search better parameters at each iteration, and when the search is complete, a final evaluation is performed on the 10% test set with a subset of the best candidates to identify the hyperparameters with better out of sample generalization. These best hyperparameters are used to train a final model with the entire training data in the original split, which is later evaluated with the subset held out for test. The results out of this evaluation are reported in the main text as well as in the rest of the manuscript.

Additional results

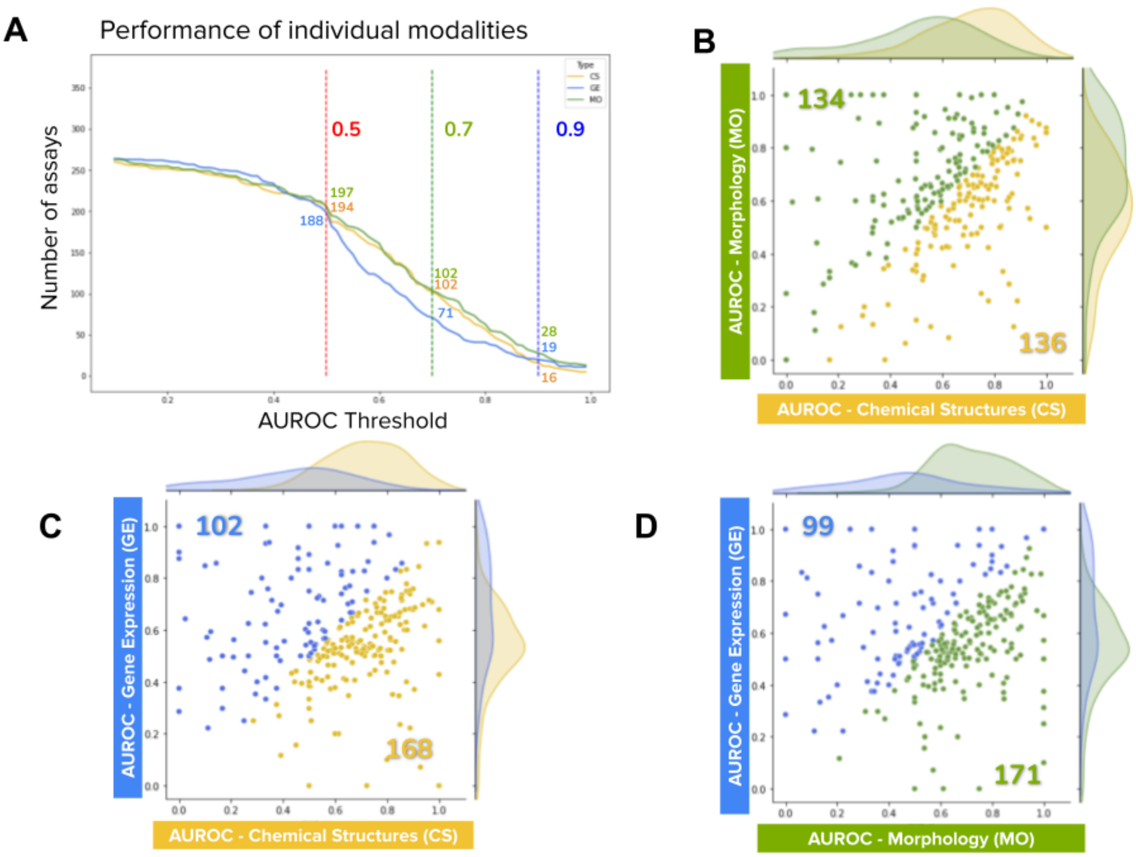


62 assays are predicted with AUC > 0.7 by each of the four combinatorial predictors

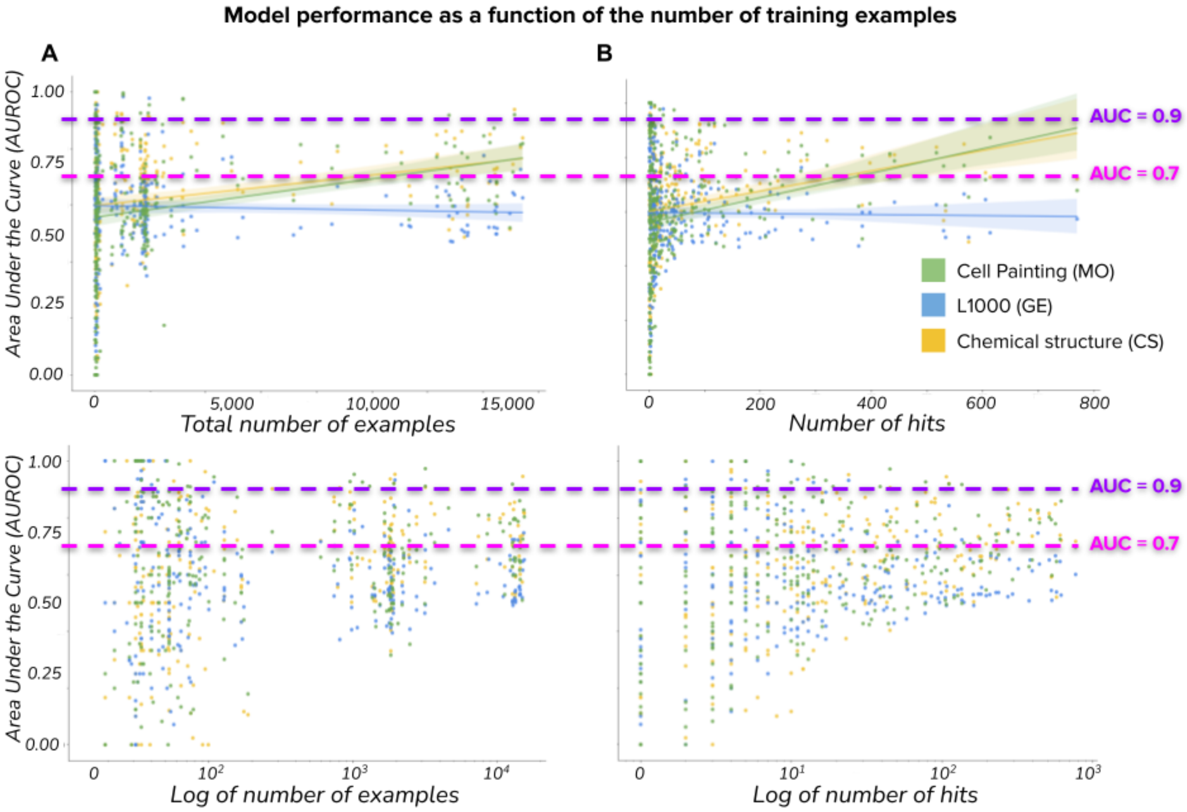
B

	CS	GE	MO	CS+GE	CS+MO	GE+MO	CS+GE+MO
Mean AUC	0.630	0.592	0.637	0.645	0.661	0.652	0.665
Mean AUC > 0.7	98	63	100	97	115	106	111
Median AUC > 0.7	102	71	102	109	108	124	117
Retrospective	-	-	-	151	157	139	174

Supplementary Figure 3. Summary of the number of assays predicted with models that have AUROC > 0.7, which is a lower performance threshold than the one used throughout our study. The total number of acceptable assay predictors increases when the threshold is lower, and chemical structures can yield more predictors that meet this level of performance. Importantly, predictors that reach performance above 0.7 AUROC are also capable of improving hit rates in many cases (see yellow points in Supplementary Figure 7). The row “Retrospective” in Table B presents the number of assays with AUROC > 0.7 that would be predicted by any of the modalities individually or their combinations.

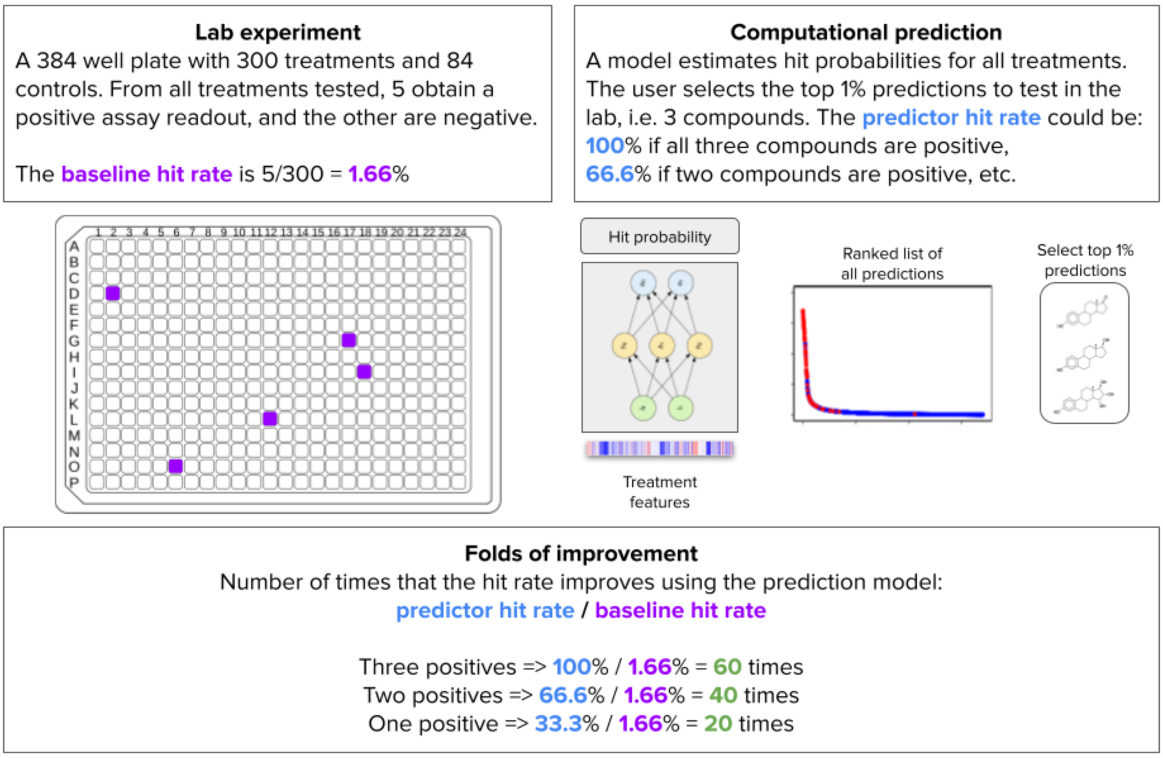


Supplementary Figure 4. Area under the curve (AUROC) performance of the three individual modalities evaluated in our study: Chemical Structures (CS), Gene Expression (GE), and Morphology (MO). A) Number of assays predicted by each modality at specific AUROC thresholds. As the AUROC threshold is increased, the number of assays meeting the threshold decreases for all modalities. The two thresholds discussed in this paper are highlighted in green (0.7) and blue (0.9). B, C, D) Scatter plots of AUROC for pairs of modalities. Each point in the plots represents an assay, the x coordinate indicates the AUROC obtained in one modality, and the y axis represents the AUROC obtained in the other modality. Colors represent the three individual modalities: CS (yellow), GE (blue) and MO (green). Points (assays) above or below the diagonal (equal performance) are colored according to the modality that has the highest AUROC. The two colored numbers inside the plot indicate the total number of assays with higher AUROC with respect to the other modality in the same plot. The counts of points indicate the number of assays where one modality is better than the other. Note that there are many points far off the diagonal, indicating high AUROC in one modality but low in the other. This indicates potential for complementary and fusion among the different data modalities.



Supplementary Figure 5. The performance of predictive models is slightly correlated with the number of available training examples; several assays can be predicted with high accuracy (AUROC > 0.9) using only a few example hits (points above the purple line). The plots show on the vertical axis the test set accuracy as a function of (A) the total number of example readouts, and (B) the number of hits available for training. Plots in the bottom row show the same data with log scale in the horizontal axis to highlight the trend with few examples. Each point is an assay predictor and its color indicates what data modality was used for training it. Note that assay prediction accuracy can vary from very low to very high with a small number of training examples, indicating that performance depends on the specific activity measured by the assay.

Folds of improvement



Lab experiment

A 384 well plate with 300 treatments and 84 controls. From all treatments tested, 5 obtain a positive assay readout, and the other are negative.

The **baseline hit rate** is 5/300 = **1.66**%

Computational prediction

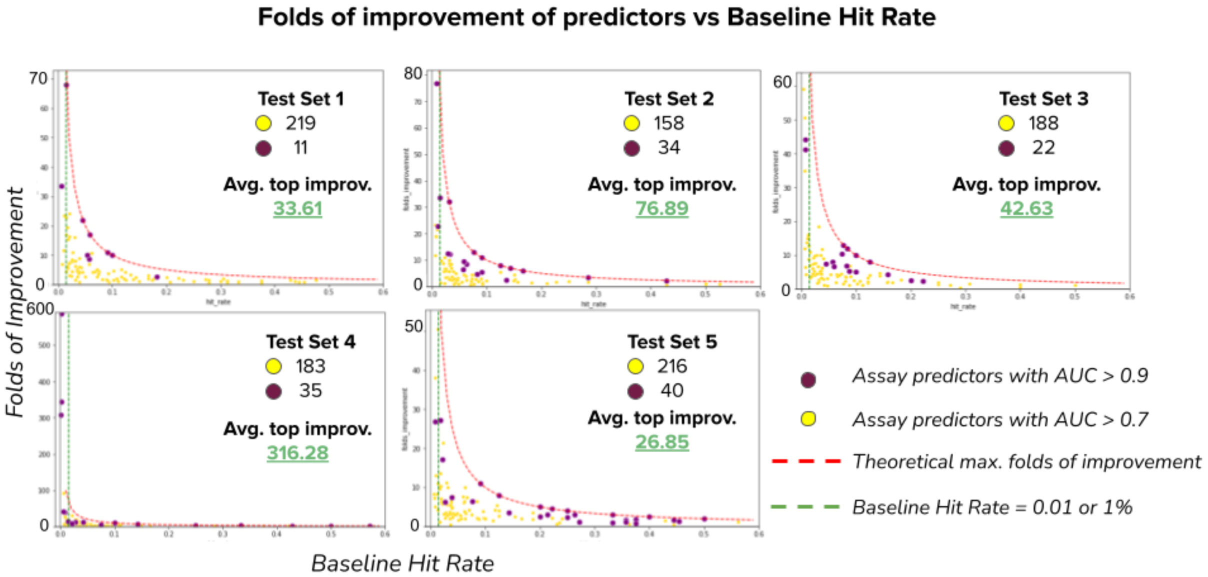
A model estimates hit probabilities for all treatments. The user selects the top 1% predictions to test in the lab, i.e. 3 compounds. The **predictor hit rate** could be:
100% if all three compounds are positive,
66.6% if two compounds are positive, etc.

Folds of improvement

Number of times that the hit rate improves using the prediction model:
predictor hit rate / **baseline hit rate**

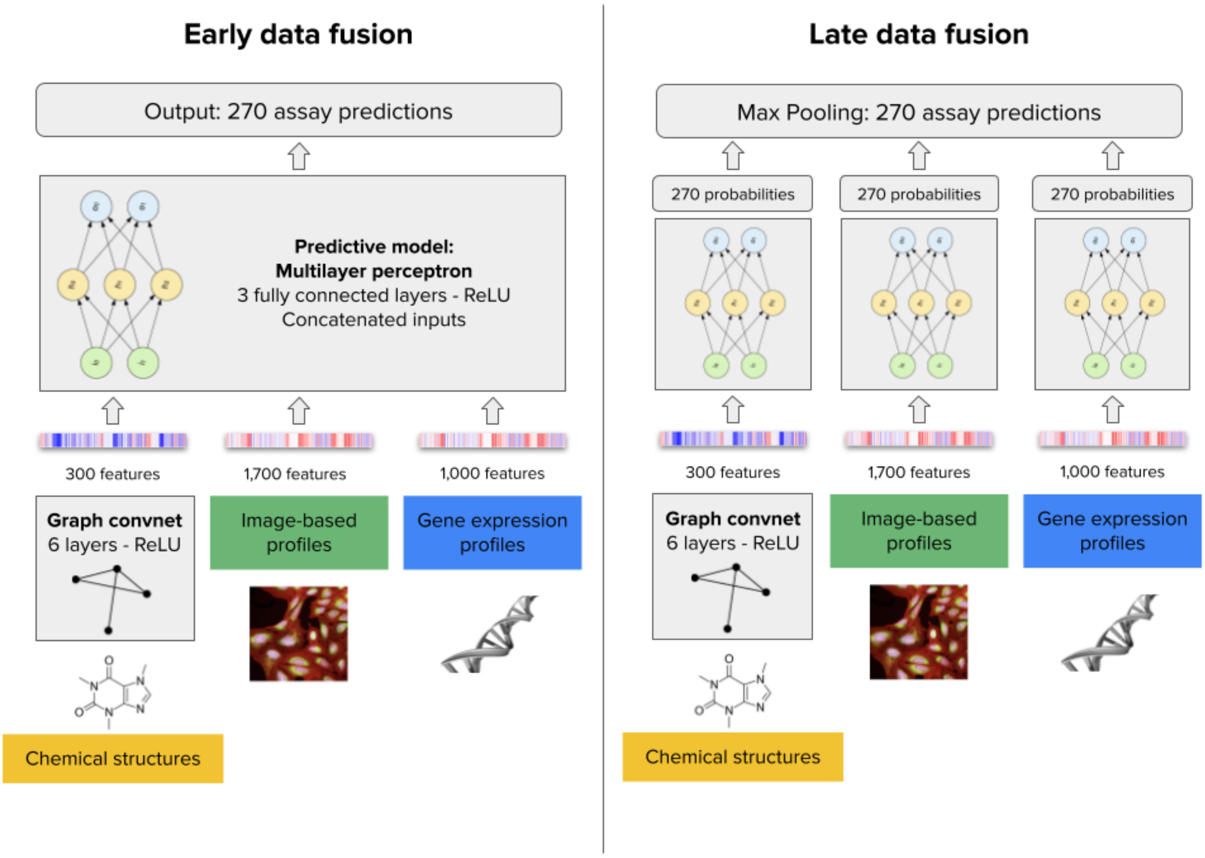
Three positives => **100**% / **1.66**% = **60** times
Two positives => **66.6**% / **1.66**% = **40** times
One positive => **33.3**% / **1.66**% = **20** times

Supplementary Figure 6. Illustration of the “Folds of improvement” metric. The example assumes a chemist testing a set of 300 candidate compounds where only 5 of them are positive hits. The ratio of hits vs tested compounds is a rough estimate of the probability of finding a hit by chance. A pre-trained computational predictor could rank the same compounds in silico from high probability of being a hit to low probability. We simulate the case where the chemist only selects the top 1% predictions for further wet lab testing, which is a reasonable cut off in real world high-throughput screens with very large compound libraries. By estimating the ratio of hits found in the top 1% subset that is actually tested in vitro, we then compute the folds of improvement as the ratio of the hit rates in each approach. Folds of improvement can be understood as the number of times that the experimental efficiency improves by using a predictor to filter unlikely hits and bring promising candidates to the top of the list.

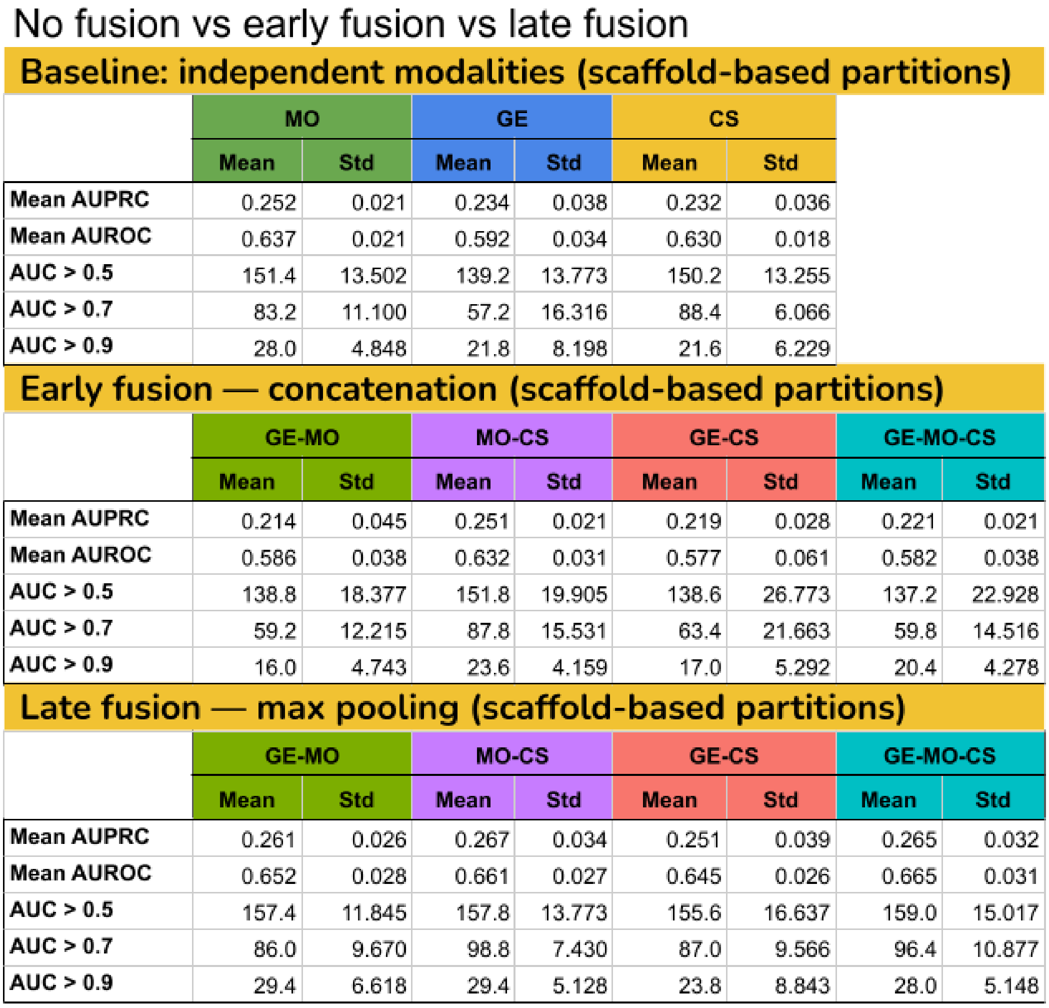


Supplementary Figure 7. Improvement of hit rates for the assays in the dataset. Each plot corresponds to the results in one split of the 5-fold cross-validation experiment (see Supplementary Figure 1). The points in the plots represent one assay predictor that uses one of the three data modalities (CS, GE or MO) or combinations of them. Assay predictors with AUROC > 0.7 are displayed in yellow and predictors with AUROC > 0.9 are displayed in purple. Assay predictors with AUROC < 0.7 are not shown. The horizontal axis represents the baseline hit rate, i.e., the proportion of compounds found to be hits in the set of tested compounds for an assay (see Supplementary Figure 6). The vertical axis presents the folds of improvement of assay predictions obtained with a machine learning predictor as a function of the baseline hit rate. Accurate predictors (AUROC > 0.9) often offer improvements up to the theoretical maximum (100% divided by the assay's baseline hit rate), and higher-fold improvements are only possible for assays with a lower baseline hit rate, i.e. with rare hits.

Data fusion



Supplementary Figure 8. Architecture of early and late data fusion models. The early data fusion model takes the three data modalities as input by obtaining features from each and then concatenating their representations. The architecture is a multilayer perceptron with three fully connected layers, 2,000 input features and 270 output predictions. The late data fusion model has one multilayer perceptron with three fully connected layers independently for each data modality. The three feature vectors are analyzed separately to produce 270 output probabilities in each case, which are later aggregated with a max-pooling operator to reduce them into a single vector of 270 assay predictions.



No fusion vs early fusion vs late fusion

Baseline: independent modalities (scaffold-based partitions)

	MO		GE		CS	
	Mean	Std	Mean	Std	Mean	Std
Mean AUPRC	0.252	0.021	0.234	0.038	0.232	0.036
Mean AUROC	0.637	0.021	0.592	0.034	0.630	0.018
AUC > 0.5	151.4	13.502	139.2	13.773	150.2	13.255
AUC > 0.7	83.2	11.100	57.2	16.316	88.4	6.066
AUC > 0.9	28.0	4.848	21.8	8.198	21.6	6.229

Early fusion — concatenation (scaffold-based partitions)

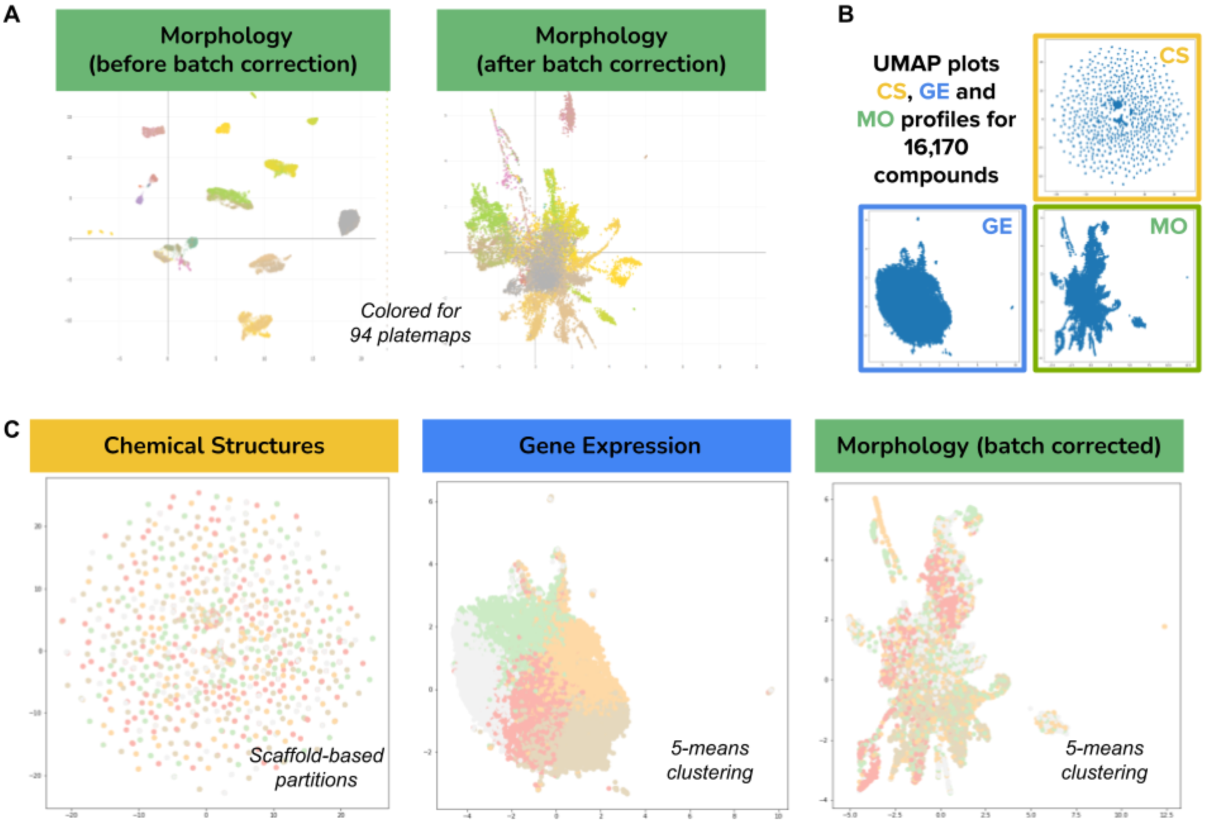
	GE-MO		MO-CS		GE-CS		GE-MO-CS	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Mean AUPRC	0.214	0.045	0.251	0.021	0.219	0.028	0.221	0.021
Mean AUROC	0.586	0.038	0.632	0.031	0.577	0.061	0.582	0.038
AUC > 0.5	138.8	18.377	151.8	19.905	138.6	26.773	137.2	22.928
AUC > 0.7	59.2	12.215	87.8	15.531	63.4	21.663	59.8	14.516
AUC > 0.9	16.0	4.743	23.6	4.159	17.0	5.292	20.4	4.278

Late fusion — max pooling (scaffold-based partitions)

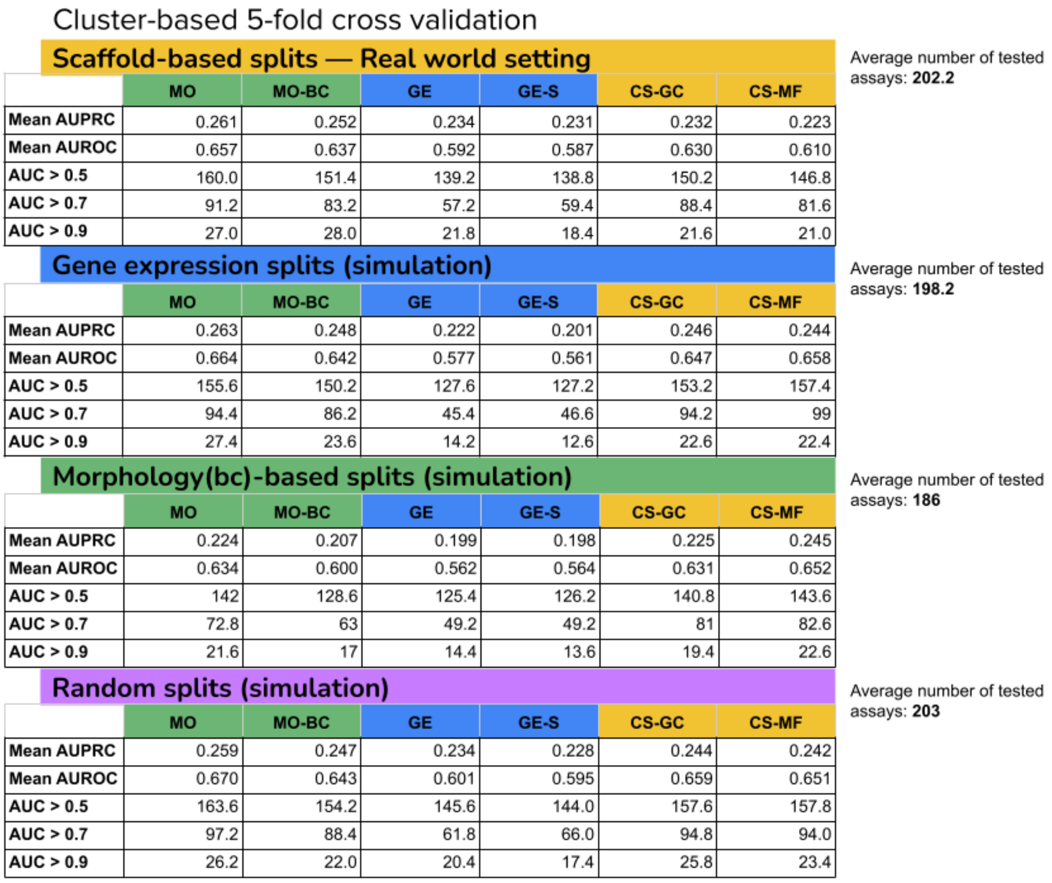
	GE-MO		MO-CS		GE-CS		GE-MO-CS	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Mean AUPRC	0.261	0.026	0.267	0.034	0.251	0.039	0.265	0.032
Mean AUROC	0.652	0.028	0.661	0.027	0.645	0.026	0.665	0.031
AUC > 0.5	157.4	11.845	157.8	13.773	155.6	16.637	159.0	15.017
AUC > 0.7	86.0	9.670	98.8	7.430	87.0	9.566	96.4	10.877
AUC > 0.9	29.4	6.618	29.4	5.128	23.8	8.843	28.0	5.148

Supplementary Table 1. Overall performance of profiling modalities and their combinations presented in the columns of the tables. Early fusion refers to concatenation of feature vectors before training predictive models, while late fusion refers to keeping the maximum prediction of individual models (see Supplementary Figure 8). The tables present four performance metrics in the rows: Mean AUPRC, mean AUROC, number of assays predicted with AUROC > 0.7, and number of assays predicted with AUROC > 0.9. For each experiment, we obtain the mean and standard deviation of the metric. In the case of the mean value for all metrics, higher numbers indicate better performance. Late fusion yields the largest number of predictors with AUROC > 0.9 overall, and also for all combinations of descriptors.

Data modalities



Supplementary Figure 9. Compound embeddings in three different feature spaces. Visualization of the high-dimensional feature vectors of all compounds using UMAP projections for the three data modalities used in this work. A) The morphology feature space originally was grouped by technical variation (plate maps), which was corrected using the Typical Variation Normalization (TVN) approach (see Methods) to report all experiments in the manuscript. The color palette for the 94 plate maps is continuous and may have similar tones for consecutive plates. B) Overview of the three feature spaces for all the 16,170 compounds included in the evaluation. Note that chemical structures (CS), gene expression (GE), and morphology (MO), all have very distinctive ways of organizing the signatures of compounds. While CS has many diverse small clusters, GE presents a single cloud, and MO has a central cloud with some medium clusters and branches. C) The same visualization as in B, but colored by clusters obtained for cross-validation experiments (see Supplementary Table 2). We partitioned each feature space using clustering to identify 5 groups for training and test splits. CS was split using Bemis-Murcko clustering, which is based on scaffold similarity, while the corresponding UMAP plot projects data points using the features of the full chemical structure (a different metric, which explains why the colors don't reveal scaffold clusters). GE and MO were split using k-means clustering, with k=5 for cross-validation in simulated control experiments to determine the influence of the data partition in the results (see Supplementary Table 2).



Cluster-based 5-fold cross validation

Scaffold-based splits — Real world setting

Average number of tested assays: **202.2**

	MO	MO-BC	GE	GE-S	CS-GC	CS-MF
Mean AUPRC	0.261	0.252	0.234	0.231	0.232	0.223
Mean AUROC	0.657	0.637	0.592	0.587	0.630	0.610
AUC > 0.5	160.0	151.4	139.2	138.8	150.2	146.8
AUC > 0.7	91.2	83.2	57.2	59.4	88.4	81.6
AUC > 0.9	27.0	28.0	21.8	18.4	21.6	21.0

Gene expression splits (simulation)

Average number of tested assays: **198.2**

	MO	MO-BC	GE	GE-S	CS-GC	CS-MF
Mean AUPRC	0.263	0.248	0.222	0.201	0.246	0.244
Mean AUROC	0.664	0.642	0.577	0.561	0.647	0.658
AUC > 0.5	155.6	150.2	127.6	127.2	153.2	157.4
AUC > 0.7	94.4	86.2	45.4	46.6	94.2	99
AUC > 0.9	27.4	23.6	14.2	12.6	22.6	22.4

Morphology(bc)-based splits (simulation)

Average number of tested assays: **186**

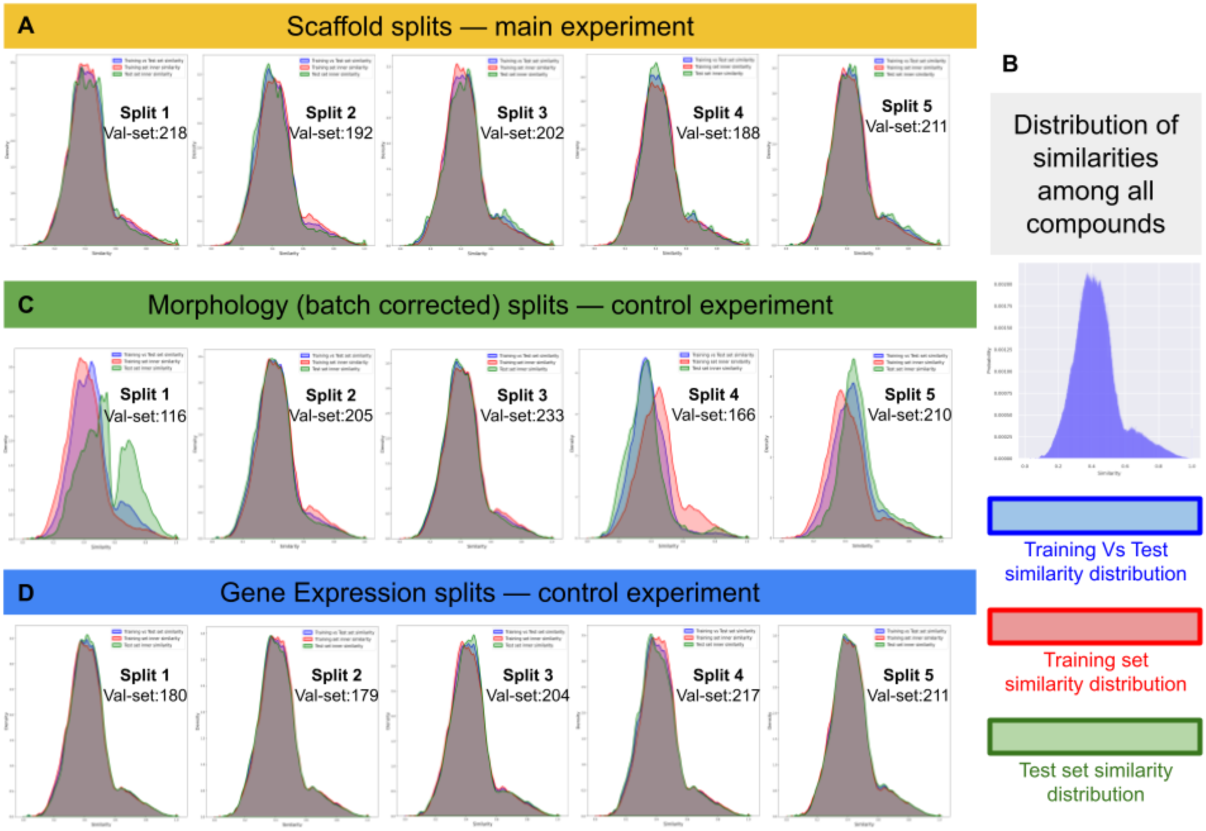
	MO	MO-BC	GE	GE-S	CS-GC	CS-MF
Mean AUPRC	0.224	0.207	0.199	0.198	0.225	0.245
Mean AUROC	0.634	0.600	0.562	0.564	0.631	0.652
AUC > 0.5	142	128.6	125.4	126.2	140.8	143.6
AUC > 0.7	72.8	63	49.2	49.2	81	82.6
AUC > 0.9	21.6	17	14.4	13.6	19.4	22.6

Random splits (simulation)

Average number of tested assays: **203**

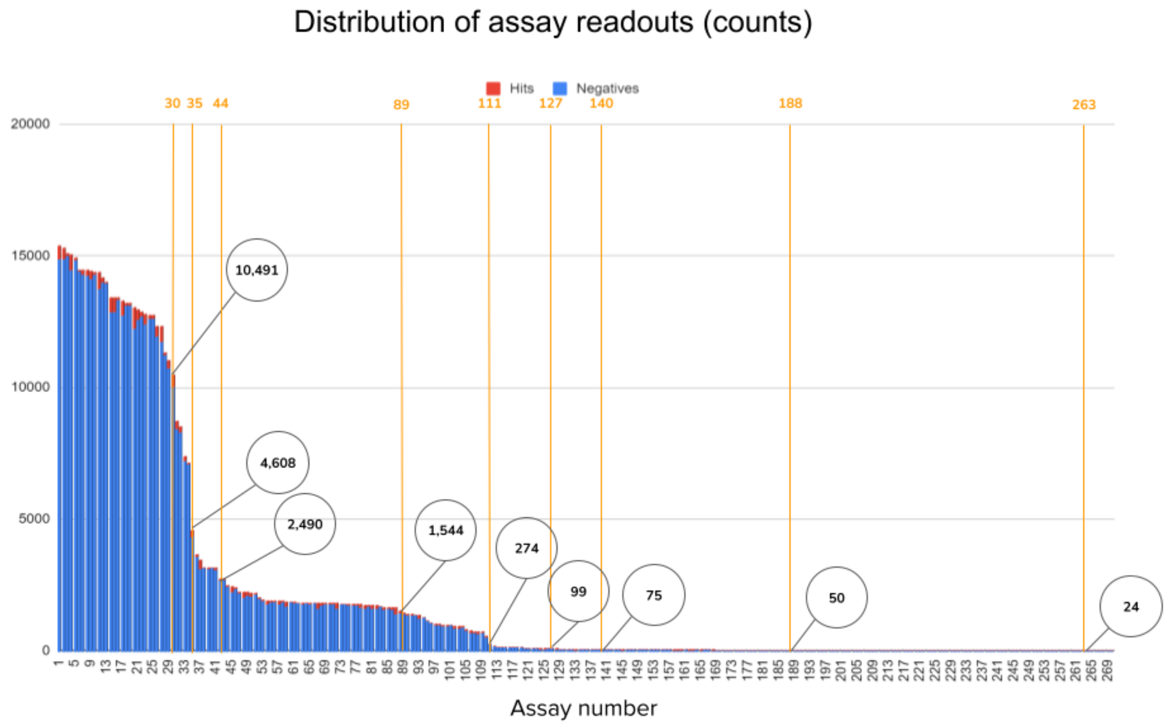
	MO	MO-BC	GE	GE-S	CS-GC	CS-MF
Mean AUPRC	0.259	0.247	0.234	0.228	0.244	0.242
Mean AUROC	0.670	0.643	0.601	0.595	0.659	0.651
AUC > 0.5	163.6	154.2	145.6	144.0	157.6	157.8
AUC > 0.7	97.2	88.4	61.8	66.0	94.8	94.0
AUC > 0.9	26.2	22.0	20.4	17.4	25.8	23.4

Supplementary Table 2. Results of 5-fold cross-validation control experiments. The tables present the mean results of 5-fold cross-validation experiments according to different data partition policies (see Supplementary Figure 9). The scaffold-based splits reflect the real world scenario more closely, while other split policies are useful as control experiments to identify potential artifacts or biases in the data. For each data modality, we used two encoding versions as follows: MO: original features and batch corrected (BC) features. GE: original features and scaled (S) or renormalized features using the L1 norm. CS: graph convolutional (GC) features and Morgan fingerprints (MF). We use as baseline the results of scaffold-based splits, which are reported in the main text and were used to complete all the analysis in this work. Compared to scaffold-based splits, gene expression and random splits yield slightly higher mean AUROC for all other modalities, which confirms that separating training and test compounds randomly makes the prediction problem easier while not being fully informative in a real setting. Morphology splits decrease performance for all modalities, indicating that the k-means splitting by morphology features (see Supplementary Figure 9) disrupts effective learning by bringing together most compounds of certain assays into only one fold. This can be explained partially by the presence of technical artifacts and by real biological signal that could not be entirely separated with the adopted batch correction method. Finally, the difference in performance between graph convolutional representations of chemical structures and Morgan fingerprints is minor across all experiments. Graph convolutions (CS-GC) have slightly better performance in the real world setting, and comparable performance in other splits. We used GC across all the reported experiments in the main manuscript.

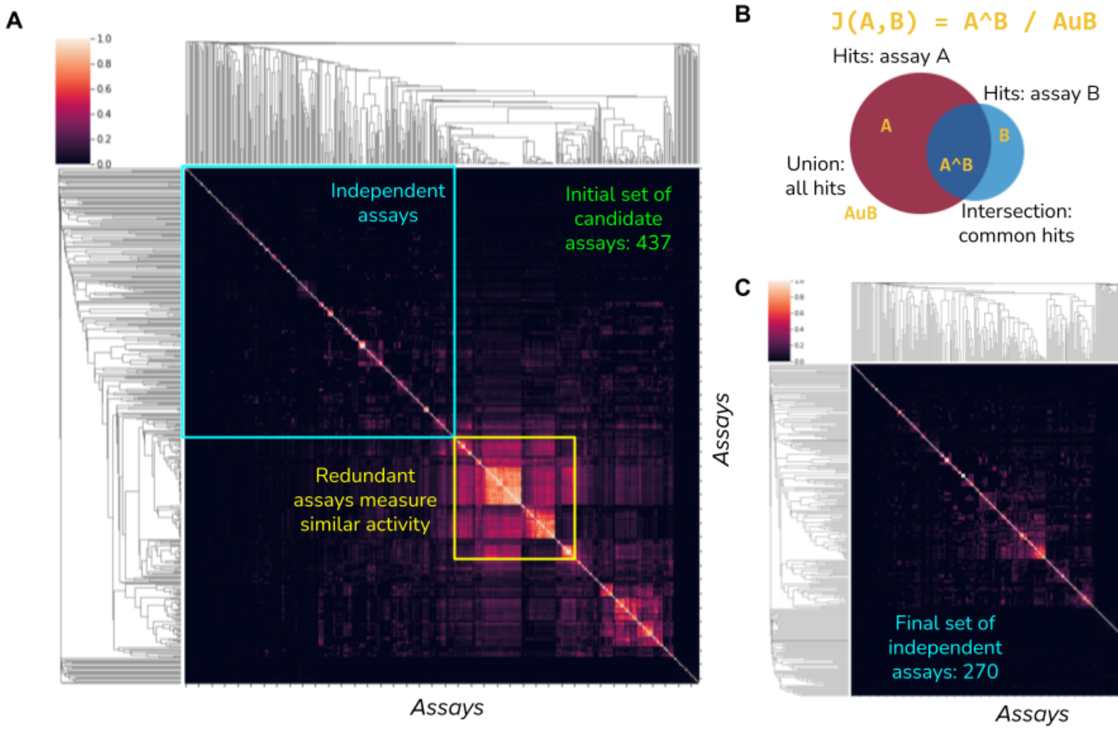


Supplementary Figure 10. Distribution of compound similarities across training-test splits. We computed the Tanimoto coefficient between Morgan fingerprints of all compounds in the dataset and obtained the distribution of scores (B), which indicates that most compounds are relatively equidistant to each other (consistent with Supplementary Figure 9C). After scaffold-based splitting, this distribution is preserved in training and test partitions in all five folds (A). No major distribution shift is observed with gene-expression splits (D), but two groups in the morphology splits (split 2 and 4) show larger differences likely explained by confounded signal between technical artifacts and biological effects (see Supplementary Table 2 and Supplementary Figure 9).

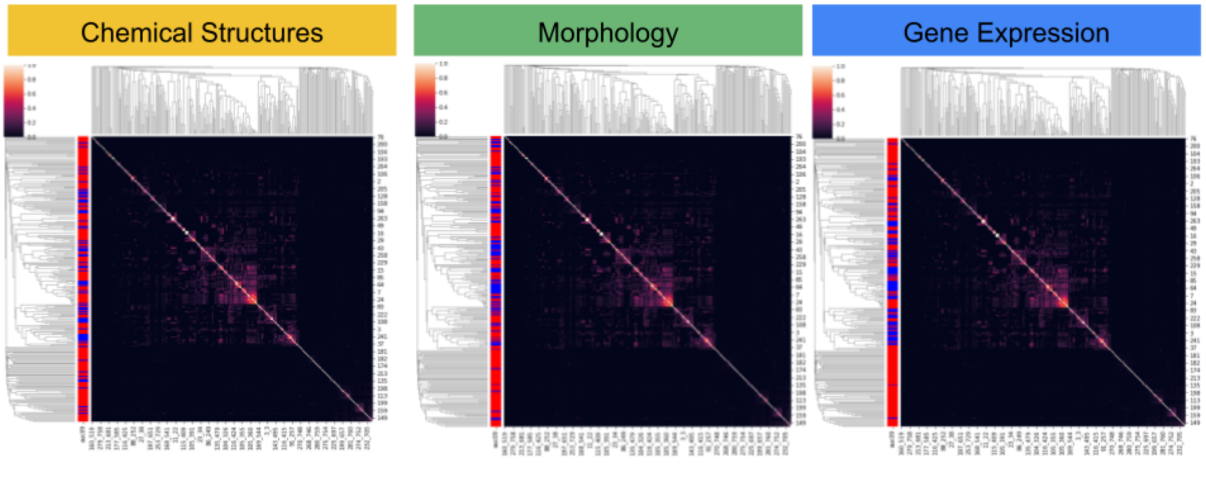
Assay data



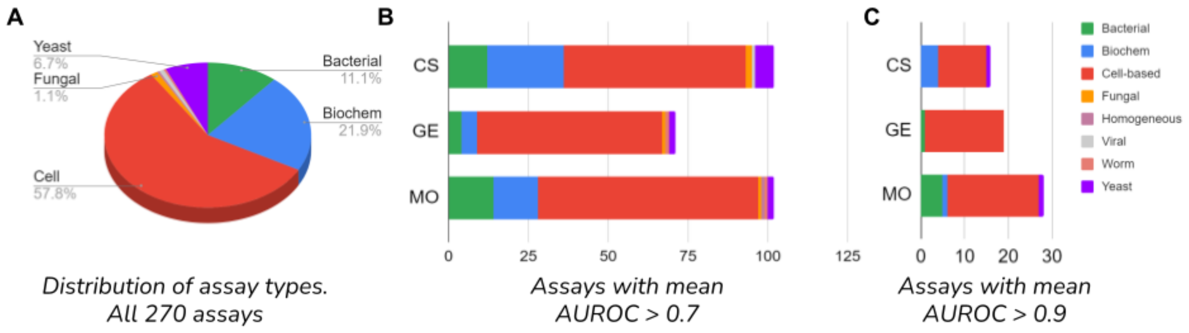
Supplementary Figure 11. Distribution of assay readouts. The plot shows in the horizontal axis assay identifiers sorted by readout count in decreasing order, and in the vertical axis the count of available readouts for each assay. Readouts can be positive hits (red) or negatives (blue). The circles in the plot indicate the readout count for specific assays in the distribution. Assay readouts follow a long tail distribution, with more than half of the assays having less than a few hundred readouts for training predictive models. Note that the ratio between hits and negative compounds is very small in general (average hit ratio 2.5%).



Supplementary Figure 12. Assay similarity. A) Matrix of assay similarities according to the Jaccard similarity between the sets of positive hit compounds. This matrix presents all the assays initially available for analysis (437). Groups of redundant assays were removed, defined as those with Jaccard index above 0.7 (for more details see Methods: Assay readouts). B) Illustration of the Jaccard similarity *J(A,B)* between two assays *A* and *B*. Each assay has a set of positive hits and we compute the ratio of the intersection (hits in common) over the union (count of all total hits) as a metric of similarity between assays. Assays that have many hits in common are likely measuring the same biological activity, and were excluded from our analysis.

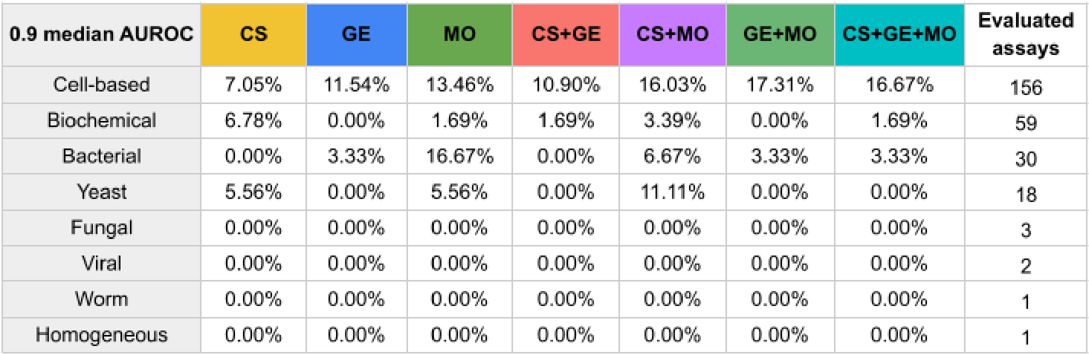


Supplementary Figure 13. Groups of assays predicted by each modality. The matrix of assay similarities is the same in the three cases: rows and columns are assays and the matrix values are the Jaccard index between the set of hits from two assays. The matrices are clustered in the rows and columns using hierarchical clustering to reveal groups of highly correlated assays. The only difference between the matrices is the coloring pattern of the left-hand side bar that indicates whether an assay is correctly predicted by the corresponding modality (chemical structures (CS), morphology (MO), and gene expression (GE)) in any of the cross-validation partitions (blue, red otherwise). This visualization is useful to reveal if the data modalities have preference for making better predictions with certain groups of assays that may have common biological activity. This result indicates that there are no major groups of activation, although accurate predictors tend to be close to each other in the cluster map. The dendrograms reveal a few assay clusters in the center of the matrices, and the visualization indicates that each modality tends to make accurate predictions in different groups; the accuracy patterns in the left of the matrices are different from modality to modality.



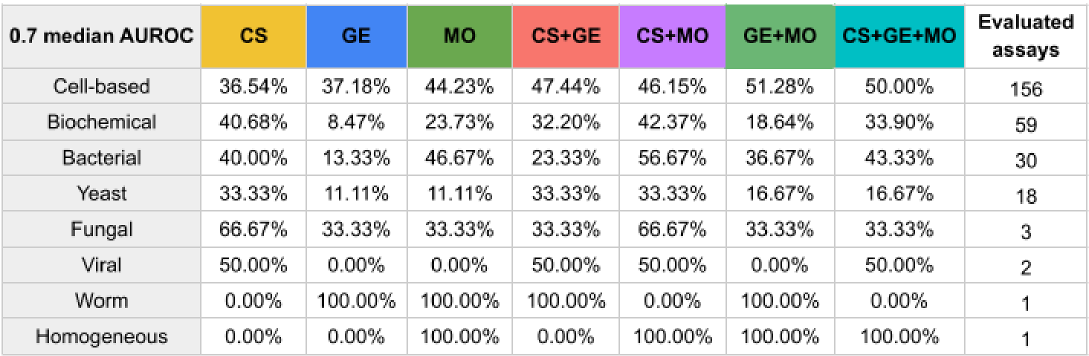
Supplementary Figure 14. Distribution of assay types as the performance threshold is decreased. The assays used in our study can be one of the seven types listed in the right hand side of the figure. A) Distribution of assays according to their type. B) Distribution of assays that can be predicted with a minimum accuracy of 0.7 AUROC by each of the three data modalities. C) Distribution of assays that can be predicted with a minimum accuracy of 0.9 AUROC by each of the three data modalities. These distributions show that none of the modalities has a strong preference for one type of assay, and that they can predict a diverse array of biological activity.

A



0.9 median AUROC	CS	GE	MO	CS+GE	CS+MO	GE+MO	CS+GE+MO	Evaluated assays
Cell-based	7.05%	11.54%	13.46%	10.90%	16.03%	17.31%	16.67%	156
Biochemical	6.78%	0.00%	1.69%	1.69%	3.39%	0.00%	1.69%	59
Bacterial	0.00%	3.33%	16.67%	0.00%	6.67%	3.33%	3.33%	30
Yeast	5.56%	0.00%	5.56%	0.00%	11.11%	0.00%	0.00%	18
Fungal	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	3
Viral	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	2
Worm	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	1
Homogeneous	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	1

B



0.7 median AUROC	CS	GE	MO	CS+GE	CS+MO	GE+MO	CS+GE+MO	Evaluated assays
Cell-based	36.54%	37.18%	44.23%	47.44%	46.15%	51.28%	50.00%	156
Biochemical	40.68%	8.47%	23.73%	32.20%	42.37%	18.64%	33.90%	59
Bacterial	40.00%	13.33%	46.67%	23.33%	56.67%	36.67%	43.33%	30
Yeast	33.33%	11.11%	11.11%	33.33%	33.33%	16.67%	16.67%	18
Fungal	66.67%	33.33%	33.33%	33.33%	66.67%	33.33%	33.33%	3
Viral	50.00%	0.00%	0.00%	50.00%	50.00%	0.00%	50.00%	2
Worm	0.00%	100.00%	100.00%	100.00%	0.00%	100.00%	0.00%	1
Homogeneous	0.00%	0.00%	100.00%	0.00%	100.00%	100.00%	100.00%	1

Supplementary Table 3. Predicted assays by type at the performance thresholds. A) Percentage of assays (out of 270 evaluated) that can be predicted by one modality or their combinations (columns) at high accuracy (>0.9 AUROC) grouped by assay type (rows). B) Same information as A but with an accuracy threshold of 0.7.

References

1. Moffat JG, Vincent F, Lee JA, Eder J, Prunotto M. Opportunities and challenges in phenotypic drug discovery: an industry perspective. Nat Rev Drug Discov. 2017 Aug;16(8):531–543. PMID: 28685762

2. Haasen D, Schopfer U, Antczak C, Guy C, Fuchs F, Selzer P. How Phenotypic Screening Influenced Drug Discovery: Lessons from Five Years of Practice. Assay Drug Dev Technol. 2017 Aug 11;15(6):239–246. PMID: 28800248

3. Warchal SJ, Unciti-Broceta A, Carragher NO. Next-generation phenotypic screening. Future Med Chem. 2016 Jul;8(11):1331–1347. PMID: 27357617

4. Bruna J, Zaremba W, Szlam A, LeCun Y. Spectral Networks and Locally Connected Networks on Graphs [Internet]. arXiv [cs.LG]. 2013. Available from: http://arxiv.org/abs/1312.6203

5. Unterthiner T, Mayr A, Klambauer G, Steijaert M, Wegner JK, Ceulemans H, Hochreiter S. Deep learning as an opportunity in virtual screening. Proceedings of the deep learning workshop at NIPS. datascienceassn.org; 2014. p. 1–9.

6. Duvenaud DK, Maclaurin D, Iparraguirre J, Bombarell R, Hirzel T, Aspuru-Guzik A, Adams RP. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In: Cortes C, Lawrence ND, Lee DD, Sugiyama M, Garnett R, editors. Advances in Neural Information Processing Systems 28. Curran Associates, Inc.; 2015. p. 2224–2232.

7. Li Y, Tarlow D, Brockschmidt M, Zemel R. Gated Graph Sequence Neural Networks [Internet]. arXiv [cs.LG]. 2015. Available from: http://arxiv.org/abs/1511.05493

8. Kearnes S, McCloskey K, Berndl M, Pande V, Riley P. Molecular graph convolutions: moving beyond fingerprints. J Comput Aided Mol Des. Springer; 2016 Aug;30(8):595–608. PMCID: PMC5028207

9. Defferrard M, Bresson X, Vandergheynst P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In: Lee DD, Sugiyama M, Luxburg UV, Guyon I, Garnett R, editors. Advances in Neural Information Processing Systems 29. Curran Associates, Inc.; 2016. p. 3844–3852.

10. Kipf TN, Welling M. Semi-Supervised Classification with Graph Convolutional Networks [Internet]. arXiv [cs.LG]. 2016. Available from: http://arxiv.org/abs/1609.02907

11. Battaglia P, Pascanu R, Lai M, Rezende DJ, Others. Interaction networks for learning about objects, relations and physics. Advances in neural information processing systems. papers.nips.cc; 2016. p. 4502–4510.

12. Schütt KT, Arbabzadah F, Chmiela S, Müller KR, Tkatchenko A. Quantum-chemical insights from deep tensor neural networks. Nat Commun. nature.com; 2017 Jan 9;8:13890. PMCID:

PMC5228054

13. Gilmer J, Schoenholz SS, Riley PF, Vinyals O. Neural message passing for quantum chemistry. Proceedings of the 34th [Internet]. dl.acm.org; 2017; Available from: https://dl.acm.org/citation.cfm?id=3305512

14. Coley CW, Barzilay R, Green WH, Jaakkola TS, Jensen KF. Convolutional Embedding of Attributed Molecular Graphs for Physical Property Prediction. J Chem Inf Model. ACS Publications; 2017 Aug 28;57(8):1757–1772. PMID: 28696688

15. Wu Z, Ramsundar B, Feinberg EN, Gomes J, Geniesse C, Pappu AS, Leswing K, Pande V. MoleculeNet: a benchmark for molecular machine learning. Chem Sci. Royal Society of Chemistry; 2018;9(2):513–530.

16. Yang K, Swanson K, Jin W, Coley C, Eiden P, Gao H, Guzman-Perez A, Hopper T, Kelley B, Mathea M, Palmer A, Settels V, Jaakkola T, Jensen K, Barzilay R. Analyzing Learned Molecular Representations for Property Prediction. J Chem Inf Model. 2019 Aug 26;59(8):3370–3388. PMCID: PMC6727618

17. Fernández-Torras A, Comajuncosa-Creus A, Duran-Frigola M, Aloy P. Connecting chemistry and biology through molecular descriptors. Curr Opin Chem Biol. 2022 Feb;66:102090. PMID: 34626922

18. Stokes JM, Yang K, Swanson K, Jin W, Cubillos-Ruiz A, Donghia NM, MacNair CR, French S, Carfrae LA, Bloom-Ackermann Z, Tran VM, Chiappino-Pepe A, Badran AH, Andrews IW, Chory EJ, Church GM, Brown ED, Jaakkola TS, Barzilay R, Collins JJ. A Deep Learning Approach to Antibiotic Discovery. Cell. 2020 Feb 20;180(4):688–702.e13. PMID: 32084340

19. Subramanian A, Narayan R, Corsello SM, Peck DD, Natoli TE, Lu X, Gould J, Davis JF, Tubelli AA, Asiedu JK, Lahr DL, Hirschman JE, Liu Z, Donahue M, Julian B, Khan M, Wadden D, Smith IC, Lam D, Liberzon A, Toder C, Bagul M, Orzechowski M, Enache OM, Piccioni F, Johnson SA, Lyons NJ, Berger AH, Shamji AF, Brooks AN, Vrcic A, Flynn C, Rosains J, Takeda DY, Hu R, Davison D, Lamb J, Ardlie K, Hogstrom L, Greenside P, Gray NS, Clemons PA, Silver S, Wu X, Zhao W-N, Read-Button W, Wu X, Haggarty SJ, Ronco LV, Boehm JS, Schreiber SL, Doench JG, Bittker JA, Root DE, Wong B, Golub TR. A Next Generation Connectivity Map: L1000 Platform and the First 1,000,000 Profiles. Cell. 2017 Nov 30;171(6):1437–1452.e17. PMCID: PMC5990023

20. Lapins M, Spjuth O. Evaluation of Gene Expression and Phenotypic Profiling Data as Quantitative Descriptors for Predicting Drug Targets and Mechanisms of Action [Internet]. bioRxiv. 2019 [cited 2020 Feb 19]. p. 580654. Available from: https://www.biorxiv.org/content/10.1101/580654v2

21. Chandrasekaran SN, Ceulemans H, Boyd JD, Carpenter AE. Image-based profiling for drug discovery: due for a machine-learning upgrade? Nat Rev Drug Discov. in-press;

22. Chandrasekaran SN, Ceulemans H, Boyd JD, Carpenter AE. Image-based profiling for drug discovery: due for a machine-learning upgrade? Nat Rev Drug Discov. Nature Publishing Group; 2020;1–15.

23. Gerry CJ, Hua BK, Wawer MJ, Knowles JP, Nelson SD Jr, Verho O, Dandapani S, Wagner BK, Clemons PA, Booker-Milburn KI, Boskovic ZV, Schreiber SL. Real-Time Biological Annotation of Synthetic Compounds. J Am Chem Soc. 2016 Jul 20;138(28):8920–8927. PMCID: PMC4976700

24. Simm J, Klambauer G, Arany A, Steijaert M, Wegner JK, Gustin E, Chupakhin V, Chong YT, Vialard J, Buijnsters P, Velter I, Vapirev A, Singh S, Carpenter AE, Wuyts R, Hochreiter S, Moreau Y, Ceulemans H. Repurposing High-Throughput Image Assays Enables Biological Activity Prediction for Drug Discovery. Cell Chem Biol. 2018 May 17;25(5):611–618.e3. PMCID: PMC6031326

25. Gustafsdottir SM, Ljosa V, Sokolnicki KL, Anthony Wilson J, Walpita D, Kemp MM, Petri Seiler K, Carrel HA, Golub TR, Schreiber SL, Clemons PA, Carpenter AE, Shamji AF. Multiplex cytological profiling assay to measure diverse cellular states. PLoS One. 2013 Dec 2;8(12):e80999. PMCID: PMC3847047

26. Bray M-A, Singh S, Han H, Davis CT, Borgeson B, Hartland C, Kost-Alimova M, Gustafsdottir SM, Gibson CC, Carpenter AE. Cell Painting, a high-content image-based assay for morphological profiling using multiplexed fluorescent dyes [Internet]. bioRxiv. 2016 [cited 2016 Aug 12]. p. 049817. Available from: http://biorxiv.org/content/early/2016/04/28/049817

27. Hofmarcher M, Rumetshofer E, Clevert DA. Accurate prediction of biological assays with high-throughput microscopy images and convolutional networks. Journal of chemical [Internet]. ACS Publications; 2019; Available from: https://pubs.acs.org/doi/abs/10.1021/acs.jcim.8b00670

28. Way GP, Kost-Alimova M, Shibue T, Harrington WF, Gill S, Piccioni F, Becker T, Hahn WC, Carpenter AE, Vazquez F, Singh S. Predicting cell health phenotypes using image-based morphology profiling [Internet]. 2020 [cited 2020 Aug 25]. p. 2020.07.08.193938. Available from: https://www.biorxiv.org/content/10.1101/2020.07.08.193938v1

29. Wawer MJ, Jaramillo DE, Dančík V, Fass DM, Haggarty SJ, Shamji AF, Wagner BK, Schreiber SL, Clemons PA. Automated Structure-Activity Relationship Mining: Connecting Chemical Structure to Biological Profiles. J Biomol Screen. 2014 Jun;19(5):738–748. PMCID: PMC5554950

30. Trapotsi M-A, Mervin LH, Afzal AM, Sturm N, Engkvist O, Barrett IP, Bender A. Comparison of Chemical Structure and Cell Morphology Information for Multitask Bioactivity Predictions. J Chem Inf Model. 2021 Mar 22;61(3):1444–1456. PMID: 33661004

31. Seal S, Carreras-Puigvert J, Trapotsi M-A, Yang H, Spjuth O, Bender A. Integrating Cell Morphology with Gene Expression and Chemical Structure to Aid Mitochondrial Toxicity Detection [Internet]. bioRxiv. 2022 [cited 2022 Apr 10]. p. 2022.01.07.475326. Available from: https://www.biorxiv.org/content/10.1101/2022.01.07.475326v1

32. Golub T. L1000 gene expression profiling assay - DOS small molecule perturbagens [Internet]. Broad Center for the Science of Therapeutics (Broad Institute); 2014. Available from: http://identifiers.org/lincs.data/LDG-1191

33. Wawer MJ, Li K, Gustafsdottir SM, Ljosa V, Bodycombe NE, Marton MA, Sokolnicki KL, Bray M-A, Kemp MM, Winchester E, Taylor B, Grant GB, Hon CS-Y, Duvall JR, Wilson JA, Bittker JA, Dančík V, Narayan R, Subramanian A, Winckler W, Golub TR, Carpenter AE, Shamji AF, Schreiber SL, Clemons PA. Toward performance-diverse small-molecule libraries for cell-based phenotypic screening using multiplexed high-dimensional profiling. Proceedings of the National Academy of Sciences. 2014 Jul 29;111(30):10911–10916.

34. Bray M-A, Gustafsdottir SM, Rohban MH, Singh S, Ljosa V, Sokolnicki KL, Bittker JA, Bodycombe NE, Dancík V, Hasaka TP, Hon CS, Kemp MM, Li K, Walpita D, Wawer MJ, Golub TR, Schreiber SL, Clemons PA, Shamji AF, Carpenter AE. A dataset of images and morphological profiles of 30 000 small-molecule treatments using the Cell Painting assay. Gigascience. 2017 Dec 1;6(12):1–5. PMCID: PMC5721342

35. Karimi M, Wu D, Wang Z, Shen Y. DeepAffinity: interpretable deep learning of compound-protein affinity through unified recurrent and convolutional neural networks. Bioinformatics. 2019 Sep 15;35(18):3329–3338. PMCID: PMC6748780

36. Manica M, Oskooei A, Born J, Subramanian V, Sáez-Rodríguez J, Rodríguez Martínez M. Toward Explainable Anticancer Compound Sensitivity Prediction via Multimodal Attention-Based Convolutional Encoders. Mol Pharm. 2019 Dec 2;16(12):4797–4806. PMID: 31618586

37. Schwaller P, Laino T, Gaudin T, Bolgar P, Hunter CA, Bekas C, Lee AA. Molecular Transformer: A Model for Uncertainty-Calibrated Chemical Reaction Prediction. ACS Cent Sci. 2019 Sep 25;5(9):1572–1583. PMCID: PMC6764164

38. Caicedo JC, McQuin C, Goodman A, Singh S, Carpenter AE. Weakly Supervised Learning of Feature Embeddings for Single Cells in Microscopy Images. IEEE CVPR. 2018;

39. Way GP, Zietz M, Rubinetti V, Himmelstein DS, Greene CS. Compressing gene expression data using multiple latent space dimensionalities learns complementary biological representations. Genome Biol. 2020 May 11;21(1):109. PMCID: PMC7212571

40. Mullard A. Machine learning brings cell imaging promises into focus. Nat Rev Drug Discov. 2019 Sep;18(9):653–655. PMID: 31477870

41. Dančík V, Carrel H, Bodycombe NE, Seiler KP, Fomina-Yadlin D, Kubicek ST, Hartwell K, Shamji AF, Wagner BK, Clemons PA. Connecting Small Molecules with Similar Assay Performance Profiles Leads to New Biological Hypotheses. J Biomol Screen. 2014 Jun;19(5):771–781. PMCID: PMC5554958

42. Baell JB, Holloway GA. New substructure filters for removal of pan assay interference compounds (PAINS) from screening libraries and for their exclusion in bioassays. J Med Chem. 2010 Apr 8;53(7):2719–2740. PMID: 20131845

43. Bemis GW, Murcko MA. The properties of known drugs. 1. Molecular frameworks. J Med Chem. ACS Publications; 1996 Jul 19;39(15):2887–2893. PMID: 8709122

44. Rohrer SG, Baumann K. Maximum unbiased validation (MUV) data sets for virtual screening based on PubChem bioactivity data. J Chem Inf Model. 2009 Feb;49(2):169–184.

PMID: 19434821

45. Yang K, Goldman S, Jin W, Lu A, Barzilay R, Jaakkola T, Uhler C. Improved Conditional Flow Models for Molecule to Image Synthesis [Internet]. arXiv [q-bio.BM]. 2020. Available from: http://arxiv.org/abs/2006.08532

46. Michael Ando D, McLean C, Berndl M. Improving Phenotypic Measurements in High-Content Imaging Screens [Internet]. bioRxiv. 2017 [cited 2017 Jul 10]. p. 161422. Available from: http://www.biorxiv.org/content/early/2017/07/10/161422

47. McQuin C, Goodman A, Chernyshev V, Kamentsky L, Cimini BA, Karhohs KW, Doan M, Ding L, Rafelski SM, Thirstrup D, Wiegraebe W, Singh S, Becker T, Caicedo JC, Carpenter AE. CellProfiler 3.0: next generation image processing for biology. PLoS Comput Biol. 2018 May 25;

48. Caicedo JC, Cooper S, Heigwer F, Warchal S, Qiu P, Molnar C, Vasilevich AS, Barry JD, Bansal HS, Kraus O, Wawer M, Paavolainen L, Herrmann MD, Rohban M, Hung J, Hennig H, Concannon J, Smith I, Clemons PA, Singh S, Rees P, Horvath P, Linington RG, Carpenter AE. Data-analysis strategies for image-based cell profiling. Nat Methods. 2017 Aug 31;14(9):849–863. PMID: 28858338