



THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à MINES ParisTech

Deep learning for computational phenotyping in cell-based assays

Apprentissage profond pour le phénotypage computationnel dans les essais cellulaires

Soutenue par

Joseph BOYD

Le 30 Juin 2020

École doctorale n°621

**Ingénierie des Systèmes,
Matériaux, Mécanique, Énergie**

Spécialité

Bio-informatique

Composition du jury :

Charles KEVRANN Directeur de recherche, INRIA Rennes	<i>Rapporteur</i>
Anna KRESHUK Group leader, EMBL Heidelberg	<i>Rapporteur</i>
Auguste GENOVESIO Directeur de recherche, INSERM ENS	<i>Examineur</i>
Florian JUG Group leader, MPI-CBG	<i>Examineur</i>
Perrine PAUL-GILLOTEAUX Ingénieur de recherche, CNRS	<i>Examineur</i>
Veronique STOVEN Professeur, MINES Paristech	<i>Examineur</i>
David ROUSSEAU Professeur, Université d'Angers	<i>Président</i>
Thomas WALTER Chargé de recherche, MINES Paristech	<i>Directeur de thèse</i>

Acknowledgements



Best Director

Thomas



Best Actor

Hector



Best Supporting Actress

Gehenna



Best Drama

Jacopo



Best Comedy

Ana



Best Short Film

Carlos



Best Hair/Makeup

Deep



Best Cinematography

Linda/Ralf



Best Original Score

Sebastian



Best Editing

Lotfi



Best Screenplay

Peter



Best Foreign Film

Paolo

To *Thomas* for his direction over the years and for reading this thesis. To *Chloé* for the many enjoyable discussions. To *Véronique* and *Jean-Philippe*. I thank you all.

To our collaborators: Alice, Fabien, Éline, Zelia, and Franck. I thank you all.

To the students of CBIO: from my first office mates, *Victor* and *Ilaria*, to the immortal *Beyrem* and his fun-loving, thrill-seeking, high-octane ways, to Marine, to Nino, to my friend and kinsman *Peter*. To the genuine triple threat that is *Benoît* (dancing, singing, overreacting). To the mighty *Judith*. To *Lotfi* and his killer backhand. To *Romain*, a greater talent than I in all domains except chess. To *Rémy*, Zejun, and *PCA*. To the new guard: *Arthur*, *Asma* and beyond. I thank you all.

To all the *Fam* at Institut Cutie: the heart and soul of PhD life, from those dreamlike forays into Paris town in the early days. To the mythic voyage in Iceland and the blood that was spilt there. To ADIC and our retreats to the East bloc. To Happy Friday, Halloween, bateaux mouches, and room 501. To *Hector*, whose thoughtfulness is surpassed only by his renown as an actor and yoga master. To *Deep*, long may his hair retain its dazzling luster. To *Lara* for giving us *Benjamin*. To *Jacopo*, who could never let a huge friendship get in the way of a close argument. To *Lindalf*, a damn fine Latvian double act. To *Sebastian*, whose birthday parties I always miss and probably always will. To *Ana*, *srećan rodjendan*. To *Roberta*, *Paolo*, *Julia*, and the God-given *Carlos*. To the next-gen *Fam* including *Sandra*, *Danny*, *Darine*, *Tommaso*, and the all-knowing, all-powerful *Om*, and uncountable others. I thank you all.

To the Frenchlution crew: *Alizée* and *Élodie*, *Chris*, *Dan*, and *John*, and the legendary American *Jake*. I thank you all.

To Gehenna, who held my hand through this.

And to my family: my parents, especially my Dad for his class and integrity, to *Carys*, *Ceri*, *Cal*. I thank you all.

Contents

1	Introduction	17
1.1	Computational phenotyping	18
1.1.1	High content screening	19
1.1.2	The elements of high content screening	20
1.1.3	High content analysis	22
1.2	Challenges for high content analysis	24
1.2.1	Multi-cell-line data	24
1.2.2	The emergent role of deep learning in high content screening	26
1.3	Contributions	28
2	Deep learning fundamentals	31
2.1	The building blocks of artificial neural networks	31
2.1.1	Backpropagation	33
2.2	Convolutional neural networks	35
2.2.1	AlexNet and the ConvNet revolution	37
2.3	Neural object detection	39
2.3.1	Regions with CNN features	40
2.4	Fighting overfitting in deep learning	41
2.4.1	Data augmentation	42
2.5	Transfer learning	45
2.5.1	Domain-adversarial neural networks	46
2.6	Generative adversarial networks	49
2.6.1	Deep convolutional GANs	51
2.6.2	Conditional GANs	51
2.6.3	Assorted GANs	52
I	Computational phenotyping for multiple cell lines	55
3	High content analysis in drug and wild type screens	57
3.1	Overview	58
3.2	Datasets	58

3.2.1	Wild type screen dataset	60
3.3	Cell measurement pipeline	61
3.3.1	Nuclei segmentation	62
3.3.2	Cell membrane segmentation	64
3.3.3	Feature extraction	65
3.4	Use cases in high content analysis	66
3.4.1	Controlling for spatial effects in the drug screen dataset	66
3.4.2	Viabilities of TNBC cell lines correlate	68
3.4.3	Cell cycle modulates double-strand break rate	70
3.4.4	TNBC cell lines assume distinct wild type morphologies	72
3.5	Discussion	76
4	Domain-invariant features for mechanism of action prediction in a multi-cell-line drug screen	77
4.1	Overview	78
4.2	Phenotypic profiling for mechanism of action prediction	80
4.2.1	MOA prediction	80
4.2.2	Phenotypic profiling	82
4.2.3	Multi-cell-line analysis	85
4.2.4	Model evaluation	88
4.2.5	Software	88
4.3	Results	88
4.3.1	Single cell line analysis	89
4.3.2	Analysis on multiple cell lines	90
4.4	Discussion	95
II	Fluorescence-free phenotyping	97
5	Experimentally-generated ground truth for detecting cell types in phase contrast time-lapses microscopy	99
5.1	Overview	100
5.1.1	Biological context	100
5.1.2	Computational phenotyping for phase contrast images	101
5.2	CAR-T dataset	102
5.2.1	Observations on the dataset	104
5.2.2	Coping with fluorescent quenching	106
5.3	Fluorescence prediction	107
5.3.1	Image-to-image translation models	108
5.3.2	Results	110
5.3.3	Bridging the gap to cell detection	112
5.4	Object detection system	115
5.4.1	Experimentally-generated ground truth	117
5.4.2	Object detection system	118

5.4.3	Results	123
5.5	Discussion	125
6	Deep style transfer for synthesis of images of CAR-T cell populations	129
6.1	Overview	130
6.2	Feasibility study: synthesising cell crops	130
6.2.1	Generative adversarial networks	131
6.2.2	Deep convolutional GANs	131
6.2.3	Conditional GANs	132
6.3	Style transfer for simulating populations of Raji cells	133
6.3.1	Conditional dilation for tessellating cell contours	136
6.4	CycleGANs for cell population synthesis	142
6.4.1	First results with CycleGANs	143
6.5	Fine-tuning a state-of-the art object detection system	145
6.6	Perspectives on synthesising a full object detection dataset	146
6.6.1	Region of interest discrimination	146
6.7	Discussion	148
7	Conclusions	151
7.1	Chapter summaries	151
7.2	The future of high content screening	153
7.2.1	Overcoming massive image annotation	153
7.2.2	The renaissance of label-free microscopy	155
7.2.3	Towards toxicogenetics	156
A	Supplementary figures	157
B	Glossary of neural network architectures	163
B.1	Fully-connected GAN	163
B.2	Deep convolutional GAN	164
B.3	F-Net	165
B.4	PatchGAN	168
C	Analysing double-strand breaks in cultured cells for drug screening applications by causal inference	171
C.1	Introduction	172
C.2	Experimental setup	172
C.3	Approaches to measuring double-strand breaks	173
C.3.1	Counting spots with diameter openings	174
C.3.2	Granulometry-based features	175
C.3.3	Average intensity	175
C.4	Analysis	176
C.4.1	Causal considerations	177

C.5 Results	178
C.6 Conclusions	179
D Supplementary analysis	181
D.1 Discovering phenotypic classes with unsupervised learning . .	181
D.2 Training a RoI classifier	183

List of Figures

1.1	The basis of a systematic high content screen. A microplate from which is registered a fluorescence (single-channel) microscopy image highlighting the nuclei of a cell population. .	20
1.2	A conventional high content analysis follows four ordered stages. Each stage may be accomplished by a variety of algorithms, and some stages may be omitted in certain pipelines, or subsumed to a common framework.	23
1.3	Comparison of cells sampled from negative control wells containing (a) TNBC cell line MDA231 and (b) TNBC cell line MDA468. Cell nuclei appear in blue, cell microtubules appear in red.	25
2.1	Fully-connected neural network with 7 input neurons, a single hidden layer with 5 neurons, and an output layer with 3 neurons.	33
2.2	Convolution operation consisting of (from left to right) an input image, convolutional kernel, and convolved output image. The <i>valid</i> convolution results in the loss of the border pixels. Notice how the output represents the gradient image, obtained by setting the kernel to the finite difference between vertically adjacent pixels.	35
2.3	Each kernel of a convolutional layer (in red) performs a convolution as a tensor-product at each spatial location of an incoming tensor, to produce a single activation map (channel) in the output tensor.	37
2.4	Object detections made by pre-trained R-CNN available in PyTorch (Paszke et al. [2017]). The image was taken in the Kyoto Gogyo ramen restaurant in Kyoto.	40

2.5	Data augmentation can make modest but useful interpolations of the space surrounding real images. CIFAR-10 images marked by red circle; augmented images marked by blue circle; black line represents natural image manifold. Upper augmentation based on conversion to grayscale (an element-wise weighted average of the RGB pixels) and rotation; the lower on colour inversion and horizontal flipping.	43
2.6	Positioning transfer learning with respect to the typical learning setting. Categories of transfer learning differ either with respect to the marginal distribution. Reproduced from https://en.wikipedia.org/wiki/D	
2.7	Overlapping domains (a) and divergent domains separated by a linear decision boundary (b).	47
2.8	A GAN trains a generator network G to fool a discriminator network D in emitting counterfeit images $\mathbf{x} \sim p_g(\mathbf{x})$ by transforming noise input $\mathbf{z}$. In order to do so, G must (implicitly) learn the data-generating distribution, p_{data}	49
3.1	Fluorescence image of MDA231 cells. DAPI highlighting the nuclei in blue, cyanine 5 highlighting the microtubules in red, and cyanine 3 highlighting the double-strand breaks as white spots on the cell nuclei.	60
3.2	Nuclei segmentation on the DAPI channel, with segmentation contours indicated by red bands.	63
3.3	Searching for spatial biases: comparison of cell counts (a), (b); comparison of first principal component of morphological profiles (c), (d); comparison of second principal component (e), (f), arranged according to the plate map of the microplate. Left column (a), (c), and (e) pertain to cell line MDA231; right column (b), (d), (f) to cell line MDA468.	67
3.4	Viability comparison between cell lines for a variety of drugs and negative controls. The neutral DMSO and untreated wells strongly overlap, while showing considerable variation in both cell lines. Viability correlates well between cell lines with Pearson correlation coefficient $\rho = 0.6556$	68
3.5	Comparison of viability by drug mechanism of action. One may observe differential effects by cell line: (a), (b) show differential effects on viability; (c) shows no clear divergence from the control cluster; (d), (e), (f) show a range of correlated effects.	69

3.6	Drug perturbations inducing significant changes to DSB distribution (measured by spot density) on cell line MDA231 at $p = 0.01$ with multiple testing correction, ordered by median spot density. Sample of cells perturbed with significant drug PKC-412 (b), DSBs visible as green spots above nucleus. A bimodal distribution is seen on the DAPI channel revealing a growth of mean intensity post DNA replication (c). An Otsu threshold (red vertical line) can be used to stratify the cell population, further revealing distributional differences in DSB rates.	71
3.7	Morphological profiles of 12 TNBC cell lines with 8 replicates apiece, based on 7 biologically meaningful classes, derived from manual annotation. One observes the phenotypic similarity between cell line families.	74
3.8	UMAP projection from feature space of sample cells from four TNBC cell lines (selected among 12) in a wild type screen. On the right we show a fixed-size (128×128 px) crop centered on an indicative cell from each cell line.	75
4.1	MOA prediction is performed on an image via a phenotypic profile. The development of such a profile spans four ordered stages. Each stage may be accomplished by a variety of algorithms, the combination of which define a unique pipeline. Some stages may be omitted in certain pipelines, or subsumed to a common framework.	80
4.2	Example of how phenotypic profiles may cluster with a hierarchical model and Ward linkage for 40 drugs in 8 mechanism of action classes (including negative control)s from our drug screen data set. Heat map colour indicates distance between profiles, and dendrogram leaf colours indicate mechanism of action class.	81
4.3	Multitask autoencoders used for dimensionality reduction over multi-cell-line data. Clockwise from top left: vanilla autoencoder, multitask autoencoder, and domain-adversarial autoencoder. Colouring indicates separate treatment of each domain (cell line).	87
4.4	t-SNE embeddings of encodings from autoencoder (left) and domain-adversarial autoencoder (right), with cell lines distinguished by colour, and mean silhouette scores of 0.11 and 0.01 respectively.	91

4.5	MDS embedding of drug effect profiles for MDA231 and MDA468 cell lines with DMSO centroid centered on origin. Detection of differential drug effects between cell lines with examples for each category below (MDA231 top, MDA468 bottom). From left to right: no drug effect in either cell line (negative control); drug effect in MDA231 cell line only; drug effect in MDA468 cell line only; similar drug effects in both cell lines; differentiated drug effects in both cell lines. Shown are example images, blue: DAPI, red: microtubules, green: DSB.	92
4.6	t-SNE embeddings of encodings from handcrafted features (left), autoencoder (center) and domain-adversarial autoencoder (right), with cell lines distinguished by colour. Respective silhouette scores of 0.22 and 0.14 and -0.02 confirm the reduced divergence in the adapted domains.	94
5.1	Aligned image channel crops (200×200 px) marking living Raji cells in mCherry (left), dead cells in GFP (center), and phase contrast (right).	104
5.2	Tracking apoptosis of a Raji cell. GFP signal accumulates as a cell undergoes morphological changes (a)-(f).	105
5.3	Raji cell population over time. Proliferation generates cell clusters.	106
5.4	CAR-T cells (devoid of fluorescence) attack Raji B cells by latching onto Raji cell surface antigens and delivering cytotoxic chemicals. The induced lysis of the target Raji cells yields growing clusters of cellular matter.	106
5.5	Comparison of average GFP (left) and mCherry (right) fluorescence measured across different fields of view at 24 hour intervals. One may observe the quenching effect of fluorescence over time, which occurs most rapidly in the first increment.	107
5.6	Pearson correlation coefficients for outputs of three fluorescent labelers, for both mCherry and GFP fluorescence prediction, measured over 80 test images.	111
5.7	Full fluorescence for indicative (512×512 px) crop from an Raji-only experiment. Columns distinguish fluorescent labeler predictions (left) and ground truth (right); rows distinguish times an early frame ($t = 0$) (top) and a later one ($t = 48$) (bottom).	113
5.8	Full fluorescence for indicative (512×512 px) crop from a CAR-T experiment. Columns distinguish fluorescent labeler predictions (left) and ground truth (right); rows distinguish times an early frame ($t = 0$) (top) and a later one ($t = 40$) (bottom).	114

5.9	Fluorescent labeler outputs produce dense clouds of mCherry fluorescence for two manually selected phase contrast inputs. These outputs may be difficult to disambiguate.	116
5.10	Pipeline for automatic construction of ground truth for training object detection system. Basic image processing steps indicated in blue, image inputs indicated in red.	118
5.11	Living and dead Raji cells revealed as distinct modes of a bimodal distribution on mean GFP fluorescence intensity per connected component of cell segmentation output.	119
5.12	Samples of living Raji cells (top), dead cells (middle), background (bottom) annotated with bounding boxes. Fluorescence is included for clarity only and is not used in training. .	120
5.13	Pipeline for training a deployment of a fully convolutional classifier. Training may occur on fixed-sized input (24×24)px, but convolutions permit variable-sized inference.	121
5.14	The processing of a test image: phase contrast input image (a); raw model bounding box predictions (b); non-maximum suppression post-processing (c); finally, for comparison, the corresponding full fluorescence image (d).	122
5.15	Population curves for manually-annotated test set <i>A</i> (a) and test set <i>B</i> (b), compared with detection system outputs. . . .	126
5.16	Comparing cell quantification strategies by accumulation of cell types over time in four well replicates, aggregating over fields of view. The labeling series are normalised to have the same mean as the detection series.	128
6.1	Example cell-centered 24×24 px crops from the ground truth training set.	130
6.2	Sample 24×24 px crops from a fully-connected GAN.	131
6.3	Sample 24×24 px crops from a DCGAN. One may notice the generator has learned to synthesise convincing peripheral cells.	132
6.4	Comparison of cells generated with DCGAN (left-most column) against 9 nearest neighbours from training set.	132
6.5	Sample 24×24 px crops from a conditional DCGAN. The top row are crops produced by conditioning for living Raji cells; the bottom are crops produced by conditioning for dead Raji cells.	133
6.6	A neural algorithm of artistic style uses a pretrained CNN to combine properties from a style source image (a) and a content source image (b) to synthesise a new image combining their characteristics (c). Example taken from https://github.com/jcboyd/vgg-fun	135

6.7	Example of Voronoi tessellation with six sites (blue points) and corresponding regions (yellow lines) (generated in <code>scipy</code> (Virtanen et al. [2020]) (a). Neighbouring cells exhibit a tessellating effect at the border (b). A conditional dilation algorithm can model this bordering effect when it comes to synthesising content images for style transfer.	137
6.8	An example of generating a cell content image by first allocating sites with <code>ALLOCATESITES</code> (a) (site intensity represents radius, dilated for visibility), with radii drawn from Equation 6.5; running the <code>CONDITIONALDILATION</code> algorithm to obtain labeled regions as in Equation 6.7 (b); and extracting the contours with Equation 6.8 (c).	140
6.9	An application of a neural algorithm of artistic style to phase contrast images of CAR-T cells (a) and a style source image (b) to synthesise a new image combining their characteristics in a “simulated” image (c).	140
6.10	More cells, more problems. The style transfer algorithm fails when the content image does not offer adequate “scaffolding” for the target texture patterns.	141
6.11	Examples of CycleGAN generated images. Columns organised with content specification input image X (left), $G(X)$ generated image (center) and reconstruction $F(G(X))$ (right). The number of cells are increased over the rows: 25, 50, 100, and 150.	144
6.12	Comparison of outcomes of Faster R-CNN on a test image when a) fine-tuned on a weakly supervised dataset and b) fine-tuned on synthetic images.	145
6.13	Conceptualisation of an extension to adversarial image-to-image translation networks with a region of interest discriminator D_{roi} . The D_{roi} relies on a library of real crops to compare to cells synthesised by G in the prescribed regions of the content image.	148
A.1	Confusion matrix for a random forest classifier classifying cells from 12 TNBC cell lines.	157
A.2	Plate map of perturbations used for drug screen. Empty cells indicate empty wells. DMSO denotes negative controls. . . .	158
A.3	MDS plots of each category of drug effect. The distances between the profiles are plotted as a line, as well as the respective distances to the centroid (origin).	159
A.4	A living Raji cell (a) is attacked by a CAR-T cell (b) and later dies (c).	160
A.5	Tracking the apoptosis of a Raji cell. Counterpart to Figure 5.2.	160

A.6	Blob detection results. Left column shows ground truth, right column shows <code>pix2pix</code> prediction. With respect to the ground truth: true positives: 31; false positives: 9; and false negatives: 11.	161
A.7	An illustration of the automatic ground truth generation image processing pipeline. The first row are the image inputs. .	162
C.1	Examples of spots (red) on cell nuclei detected with diameter openings on the Cy3 channel (grey).	174
C.2	Granulometry and spot density readouts show high correlation on average over all wells.	175
C.3	Distribution of cell total intensities on the DAPI channel for MDA231 cell line (a). The bimodal distribution is a consequence of the growth sustained between the G_1 (black) and G_2 (red) phases of the cell cycle. A suitable threshold of DAPI intensity significant divergence in the DSB distributions of the respective groups on the Cy3 channel (b).	176
C.4	Causal diagram including selection bias: P is the perturbation; D is the distribution of double-strand breaks and; A is the frequency of apoptosis. Conditioning (represented as a square) on the common effect of treatment P and outcome D creates a selection bias.	177
C.5	Causal diagram showing effect modification of cell cycle phase: G is cell cycle phase; P is the perturbation, and D is the DNA damage.	178
D.1	Examples of spots (red) on cell nuclei detected with diameter openings on the Cy3 channel (grey).	182
D.2	From left to right: canvas input, activation map after Max-Pooling layer, activation map after RoIAlign layer (reduced to single bounding box).	183
D.3	Prediction of 16 digits over canvas, performed in a single forward pass.	184
D.4	Encodings of class and localisation specification (left and center) and corresponding train image for training a conditional GAN with RoI discriminator.	185
D.5	With the same specification of object localisation and class, a ground truth image (a), a synthetic image from a plain <code>pix2pix</code> system (b) and synthetic image from our <code>pix2pix</code> with RoI discrimination.	185

List of Tables

2.1	Single-model test errors on ImageNet for five groundbreaking CNNs. Note AlexNet was evaluated on the ILSVRC 2012 dataset, the others on ILSVRC 2014. Human performance has been estimated to be 5% Top-5 error (Karpathy [2014]).	39
3.1	The drug screen pilot data, consisting of two cell lines on separate 384-well plates, with four fields per well. Four fluorescence channels are captured, under four acquisition modes. The full data set of 49152 images is the outer product of each of the table fields.	59
3.2	The stains used in the fluorescence microscopy of the screen and their corresponding biological markers.	59
3.3	The wild type screen data, consisting of 12 cell lines evenly distributed over a 96-well plate (ordered by column). Four fluorescence channels are captured as in the drug screen, albeit with FITC replaced by rhodamine in the final four rows.	61
3.4	TNBC cell lines and negative control MCF-10A properties referenced from Chavez et al. [2010]. Drug screen cell lines indicated in bold. ¹ Site: NB, normal breast; PE, pleural effusion; PT, primary tumour. ² Pathology: AC, adenocarcinoma; IDC, infiltrating ductal carcinoma; IMC, infiltrating medullary carcinoma.	62
3.5	Morphological classes manually annotated on wild type screen data to create a ground truth for training cell classifier.	73
4.1	Comparison of dimensionality reduction approaches against unreduced baseline for cell lines treated separately. We show mean and standard deviation of accuracies over 60 runs with (*) indicating significant results at the $p = 0.05$ level; (**) at the $p = 0.01$ level.	89

4.2	MOA prediction on multiple cell lines (pooled) with autoencoders trained on handcrafted features. From top to bottom: vanilla autoencoders (baseline), multitask autoencoders and domain-adversarial autoencoders. We compare with the vanilla autoencoder (top row) (**): $p < 0.01$	90
4.3	MOA prediction on multiple cell lines (pooled) with convolutional autoencoders. From top to bottom: vanilla convolutional autoencoders (baseline), multitask convolutional autoencoders and domain-adversarial convolutional autoencoders. We compare with the vanilla convolutional autoencoder (top row) (**): $p < 0.01$	91
5.1	Characteristics of CAR-T experiments studied. Row A studies RAJI cells in isolation; row B studies cocultured Raji and CAR-T cells.	103
5.2	Specification of the network architecture. We distinguish three multitask outputs: Output_o , the probability of object presence; Output_c , the probability of class c given an object; and Output_b , the tuple of object width and height.	120
5.3	Detection performance test set A , stratified by object class. Best results in bold.	124
5.4	Detection performance on test set B , stratified by object class. Best results in bold.	125
5.5	Correlations between object detection and fluorescence prediction time series for alive and dead Raji cells in four experimental replicates.	127
C.1	Number of hits (out of 168) for each DSB quantifier on the MDA231 cell line. Significance at the 0.01 level with the Benjamini-Hochberg correction.	179

Chapter 1

Introduction

Summary: *Computational phenotyping is an emergent set of technologies for systematically studying the role of the genome in eliciting phenotypes, the observable characteristics of an organism and its subsystems. In particular, cell-based assays screen panels of small compound drugs or otherwise modulations of gene expression, and quantify the effects on phenotypic characteristics ranging from viability to cell morphology. High content screening extends the methodologies of cell-based screens to a high content readout based on images, in particular the multiplexed channels of fluorescence microscopy. Screens based on multiple cell lines are apt to differentiating phenotypes across different subtypes of a disease, representing the molecular heterogeneity concerned in the design of precision medicine therapies. These richer biological models underpin a more targeted approach for treating deadly diseases such as cancer. An ongoing challenge for high content screening is therefore the synthesis of the heterogeneous readouts in multi-cell-line screens. Concurrently, deep learning is the established state-of-the-art in image analysis and computer vision applications. However, its role in high content screening is only beginning to be realised. This dissertation spans two problem settings in the high content analysis of cancer cell lines. The contributions are the following: (i) a demonstration of the potential for deep learning and generative models in high content screening; (ii) a deep learning-based solution to the problem of heterogeneity in a multi-cell-line drug screen; and (iii) novel applications of deep image-to-image translation models as an alternative to the expensive fluorescence microscopy currently required for high content screening.*

Résumé: *Le phénotypage computationnel est un ensemble de technologies émergentes permettant d'étudier systématiquement le rôle du génome dans l'obtention de phénotypes, les caractéristiques observables d'un organisme et de ses sous-systèmes. En particulier, les essais cellulaires permettent*

de cribler des panels de petites molécules ou de moduler l'expression des gènes, et de quantifier les effets sur les caractéristiques phénotypiques allant de la viabilité à la morphologie cellulaire. Le criblage à haut contenu étend les méthodologies des criblages cellulaires à une lecture à haut contenu basée sur des images, en particulier les canaux multiplexés de la microscopie à fluorescence. Les cribles basés sur de multiples lignées cellulaires sont aptes à différencier les phénotypes de différents sous-types d'une maladie, représentant l'hétérogénéité moléculaire concernée dans la conception de thérapies médicales de précision. Ces modèles biologiques plus riches sous-tendent une approche plus ciblée pour le traitement de maladies mortelles telles que le cancer. Un défi permanent pour le criblage à haut contenu est donc la synthèse des lectures hétérogènes dans les cribles à multiples lignées cellulaires. Parallèlement, l'état de l'art établi en matière d'applications d'analyse d'images et de vision par ordinateur est l'apprentissage profond. Cependant, son rôle dans le criblage à haut contenu ne fait que commencer à être réalisé. Cette thèse aborde deux problématiques de l'analyse à haut contenu des lignées cellulaires cancéreuses. Les contributions sont les suivantes : (i) une démonstration du potentiel d'apprentissage profond et de modèles générateurs dans le criblage à haut contenu ; (ii) une solution basée sur l'apprentissage profond au problème de l'hétérogénéité dans un criblage de médicaments sur plusieurs lignées cellulaires ; et (iii) de nouvelles applications de modèles de traduction d'image à image comme alternative à la microscopie à fluorescence coûteuse actuellement nécessaire pour le criblage à haut contenu.

1.1 Computational phenotyping

The success of the Human Genome Project ([Lander et al. \[2001\]](#)) in mapping the totality of human genes has inspired similar efforts for the enumeration of biological phenotypes. One useful relation views a *phenotype* as, in conjunction with environmental factors, the manifestation of the genetic code. Natural selection has been said to operate in the “P-space” of all possible phenotypes, with selection propagating to the “G-space” of all possible genotypes. Hence, phenetic information bears directly on important biological questions regarding disease and mortality ([Houle et al. \[2010\]](#)).

Unlike a genotype, which is constrained to the configuration of a fixed number of genes, an organism's phenotype spans the space of its observable characteristics. This notion is somewhat problematic, as what is observable has broadened in time with advancements in technology. As a result, an organism's phenotype today may span its molecular, cellular, tissular, morphological, and behavioural traits, none of which are moreover necessarily fixed in time. For this reason, it is often convenient to refer to a tractable

subset of an organism’s phenotype, for example that pertaining to an organismal subsystem, such as the properties of its cells. As a result, the term *phenotype* and the related *phenome* are in practice want of semantic hygiene (Mahner and Kary [1997]).

Phenomics, by analogy to genomics, refers to the study of high-dimensional readouts across the full spectrum of an organism’s phenotype (Houle et al. [2010]). The interest in phenomics coincides with the rise of bioimage informatics (Myers [2012]). Bioimages, deriving from the various forms of microscopy, contain rich information on the morphological aspects of cells, cell populations in culture or tissue, and complete organisms, that is lost in other biological readouts. Bioimages in time-lapse can additionally reveal behaviour and track phenotypic changes in motion. Thus, bioimages are a favourable medium for phenotypic information.

1.1.1 High content screening

High content screening (HCS) is a methodology for the systematic discovery of phenotypes from image data in cellular assays (Haney [2008]). HCS may be viewed as extending the methodology of high throughput screening (HTS) to the medium of images. HTS is an experimental setup to test many experimental conditions in a systematic way, typically with a very simple readout, for example cell viability or a univariate measure of cytotoxicity. HCS performs such screens with a more complex and informative readout by leveraging, in particular, multiple channels of fluorescence microscopy. HCS thus increases the “content” of the readout to a large number of features (perhaps hundreds), while maintaining the throughput. HCS can be used for fundamental biological research, where gene expression can be modulated via techniques such as RNA interference, or otherwise knocked out entirely, inducing phenotypic effects in cultured cells. HCS has been instrumental in deciphering the molecular basis of a number of diverse biological processes, such as cell division (Neumann et al. [2010]), protein secretion (Simpson et al. [2012]), and endocytosis Collinet et al. [2010]. HCS has also been used to systematically screen for the localisation of biomolecules inside cells Boland et al. [1998]).

HCS also plays a role in the early “hit-to-lead” stages of the drug discovery process (Haney et al. [2006], Pepperkok and Ellenberg [2006a]). In this case, a cell line population is exposed to *small molecule* drug compounds. Thus, the cell line is the model for a disease, and the screening process aims to identify the drugs that are active thereupon. For instance, one may be interested in identifying drugs that specifically kill cancer cells. HCS complements other techniques for drug identification such as biochemical assays. For instance, Swinney and Anthony [2011] differentiate target-based and

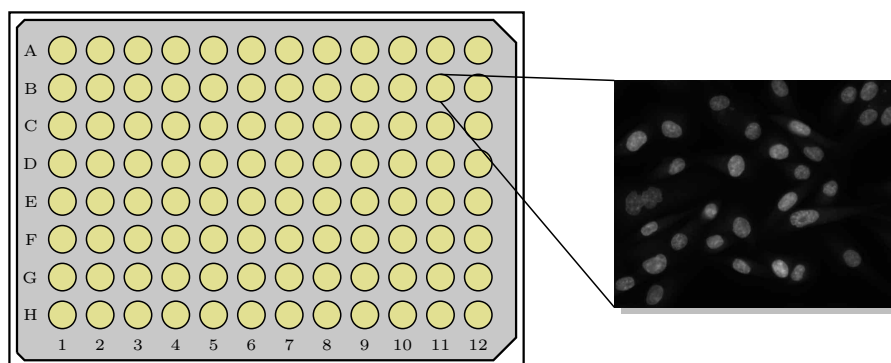


Figure 1.1: The basis of a systematic high content screen. A microplate from which is registered a fluorescence (single-channel) microscopy image highlighting the nuclei of a cell population.

phenotypic screens. The study looks at 257 drugs published between 1999 and 2008. Despite the preeminence of target-based approaches, they find the most common mode of first-in-class drug discovery is phenotypic screening. This is most true of infectious and central nervous system diseases. Cancer treatments are most frequently discovered by biologics¹, which also predominate for diseases of the immune system. Target-based approaches succeed for the discovery of half of all follower drugs.

1.1.2 The elements of high content screening

In order to perform experiments in high throughput, HCS relies to a significant extent on experimental automatisation. Experiments are conducted in wells arranged in a grid structure on a microtiter plate² as depicted in Figure 1.1. The wells are seeded to *confluence*³ with cells of a chosen cell line. Within each well, a different perturbation experiment is conducted. In all biological experiments it is important to rely on proper controls against which the tested perturbations are compared. Screens compare populations of negative controls (unperturbed) with others exposed to perturbation. Positive controls take the form of perturbations with a known effect, such as small molecule cytotoxicity in a drug screen. Some number of images corresponding to non-overlapping *fields of view* (henceforth *fields*) are taken from each well, typically producing from hundreds to thousands of unique images per plate. The following briefly describes the key elements of a screen.

¹Biologics are genetically-engineered proteins that target the immune system.

²Microtiter plates (hereafter *plates*) are small (usually polystyrene) trays divided into a rectangular grid of wells functioning as individual test tubes.

³Such that the well surface is covered with cells.

Cell lines

The object of analysis of an HCS assay is an immortalised cell line (hereafter referred to as a *cell line*), which consists in a population of cells, sustained by division without senescence. As a result, a cell line continues replicating indefinitely from a common ancestor. Cell lines are the biological model acting as a representation of the disease. The first and most well-known cell line generated is the HeLa cell line (Scherer et al. [1953]). Since HeLa, many cell lines have been developed. Cell lines are useful biological models due to their longevity in cell culture, and are used extensively in biomedical research, for example in assessing the cytotoxicity of a drug treatment. However, their accuracy as biological models can be compromised by their inherent mutations, and the effects of repeated passages, cloning, and biochemical contaminants can lead to significant genetic drift from their *in vivo* ancestors (Marx [2014]). Despite these limitations, cell lines remain the most widely used model system in screening, and many scientific and pharmacological discoveries have been made from screens on cell lines.

Microscopy and fluorescence

In high content screening, imaging data derives from one or another of the many types of optical microscopy. A broad range of techniques for performing microscopy exist, each leveraging the principles of optics in different ways, and the technique will be tailored to the experimental objectives. *Bright-field* microscopy passes visible light through a sample, producing a picture in which light is attenuated according to the varying densities of the imaged specimen. *Phase contrast* is a more sophisticated variant of transmitted light microscopy, measuring the phase shift of the visible light traveling through the specimen, producing an image with a greater degree of contrast.

Fundamental to high content screening is *fluorescence microscopy*, which uses a laser to excite fluorescent molecules in organic matter. These molecules are known as *fluorophores* and emit light at a unique wavelength (think colour) upon excitation by a laser. As such, localised fluorophores can be utilised to highlight key cellular regions. The predominant technique used in HCS is immuno-fluorescence, which relies on fluorescently-labeled antibodies. Other widely used techniques include labeling of live samples, stable expression, or fluorescence in situ hybridisation. In most screening applications, the nuclei are stained with one of these techniques, allowing for the subsequent identification of individual cells. The other fluorescent markers are selected in accordance with the research questions, for example, microtubules, Golgi apparatus, or plasma membrane. Thus, the fluorescence

markers of a screen, responding to distinct wavelengths of light, yield a set of multiplexed images painting a composite picture of the cell in its key sub-cellular structures. Image analysis can then be used to extract the high content readouts of the screen.

Project workflow

A typical HTS project workflow commences with a pilot screen that validates the pipeline from experimental protocol through to image analysis (see [Terjung et al. \[2010\]](#)). This is followed by a large scale screen, increasing the number of perturbations tested. It is here that candidate *hits* (drug perturbations registering a significant effect) are identified by automatic analysis (see the following Section 1.1.3). Finally, candidate hits are carried forward to secondary screens and are studied in greater detail.

1.1.3 High content analysis

With a large image dataset in hand (Section 1.1.2), so begins the automated image analysis, or *high content analysis*. Each image depicts a population of cells subject to perturbation or else representing a control case. The aim is to attribute a phenotype to the population in terms of a specific measurement or vector of measurements. Analysis of the pixel values across the various fluorescent channels yields a set of features or *readouts*. Cell phenotypes are compiled through feature extraction of each measured unit of the image. The distribution of perturbed readouts can be compared to control cases in a statistical framework so as to establish screen hits. In order of complexity, the readouts may be categorised according to the following:

- I **Univariate:** In the ideal case, biological functions may be quantitatively described by a single feature. For example, the nuclear area might increase dramatically under certain treatments. Thus, it would be sufficient to measure the corresponding feature (nuclear size) and statistically analyse its distribution under the different experimental conditions. Such a scenario would likely be easier to explain in biological terms.
- II **Multivariate:** A more complex case arises when one analyses different phenotypic descriptors (biologically meaningful features) and their interdependencies. Then one would contend with the multi-variate distribution of these features, and the analysis would be necessarily more sophisticated.
- III **Machine learning:** In a final case, phenotypes might not be discernible in such basic terms, and would rather require the tools of

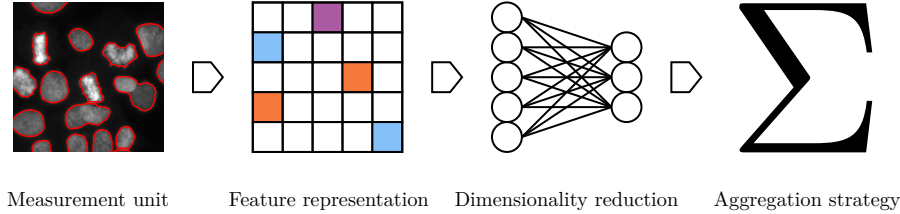


Figure 1.2: A conventional high content analysis follows four ordered stages. Each stage may be accomplished by a variety of algorithms, and some stages may be omitted in certain pipelines, or subsumed to a common framework.

statistical learning to elucidate more subtle patterns in the cell population. In this case, one would rather extract a large number of features without clear biological meaning. Learning can then be supervised or unsupervised. In the supervised case, the biological meaning could be injected (by the analyst) through use of biologically meaningful classes.

These readouts may be taken at various scales. However a logical (and indeed, conventional) starting point is at the level of individual cells (Perlman et al. [2004], Adams et al. [2006]), entailing an initial segmentation of the image. As a side note, it is for this reason that it is favourable to seed cells at a density that will minimise cell overlap. The seeding density must therefore be chosen according to the unique morphological properties of the cell line (Bray et al. [2016]). However, *segmentation-free* approaches directly analysing the full image (Orlov et al. [2008], Uhlmann et al. [2016]), or image segments, have proven successful, in particular through application of deep learning (Kraus et al. [2016]). The tradeoff is between a fine-grained analysis at the cellular level, where careful consideration of cell structure and fluorescent colocalisation is a focus (Slack et al. [2008]), and a coarse analysis of the cell population, where population densities and dynamics can be measured. Attempts to benefit from both scales have been made (Godinez et al. [2017]). After these early stages of image analysis, a screen dataset is usually subject to a range of *quality control* procedures, which may use automatic techniques to detect artifacts such as image blur or saturation, or else detect outliers among cells that have been under- or over-segmented (Caicedo et al. [2017]).

After these initial stages, particularly a type III analysis may proceed towards a *phenotypic profiling* of the cell population. Figure 1.2 encapsulates a conventional approach to profiling. Dimensionality reduction is used variously to eliminate redundant features, as well as to compress the data into its

essential components (for example, the set of principal components derived over the population of cell feature vectors). Here, the options abound and the choice of method is determined by the analytical objectives. A simple, motivating example is the case of counting classes of cells in a population. Here, a classifier (such as in [Neumann et al. \[2010\]](#)), performs the role of dimensionality reduction: the classifier maps the feature vector of each cell to a scalar or one-hot encoding representing cell class,

$$f : \mathbf{x} \rightarrow \{0, 1\}^K, \quad (1.1)$$

for K classes of cells. The population phenotype is then summarised in $\mathbf{p} \in \mathbf{R}^K$, obtained by aggregation as a simple summation or average, giving the number or proportion of cells per cell class respectively,

$$\mathbf{p} = \frac{1}{N} \sum_{i=1}^N f(\mathbf{x}_i), \quad (1.2)$$

for the N cells in the population.

1.2 Challenges for high content analysis

High content analysis offers many interesting research directions. Those most relevant to this dissertation are described in the following.

1.2.1 Multi-cell-line data

As mentioned before, the use of a single cell line limits HCS as a drug screening approach. Diseases, in particular, cancer, are often heterogeneous on a molecular level. This translates therapeutically to scenarios in which a treatment effective against one molecular subtype is ineffective against (or even promotes genetic instability in) another subtype. Hence, a single cell line introduces a bias towards a particular disease subtype. This motivates the validation of discoveries against multiple cell lines, representing different subtypes of the same disease. A screen based on multiple cell lines, representing multiple disease subtypes, can help in formulating hypotheses on, for example, the mechanism of resistance to disease. This is furthermore a step in the direction of the emerging paradigm of precision medicine ([Ashley \[2016\]](#)), where machine learning will play a decisive role (see, for example, [Krittanawong et al. \[2017\]](#)). Genomics has led the way so far, but other data sources including images are expected to become increasingly part of the picture [Hulsen et al. \[2019\]](#). This has motivated a large consortium of

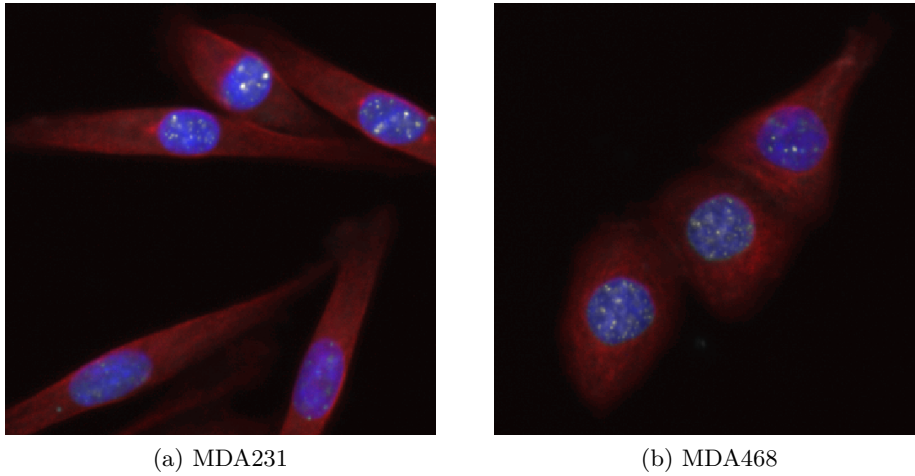


Figure 1.3: Comparison of cells sampled from negative control wells containing (a) TNBC cell line MDA231 and (b) TNBC cell line MDA468. Cell nuclei appear in blue, cell microtubules appear in red.

research groups to propose multi-cell-line drug screens, and to build models capable of predicting the efficiency of a drug from the transcriptomic and genetic data of the cell-line (Costello et al. [2014]). However, the success of these approaches has been rather modest. One of the reasons for this was the measurement of drug efficiency, that reduced the drug effect to a single number. Drug effect similarities cannot be reasonably calculated from such data. One therefore stands to gain from studying multi-cell-line drug effects rather in high content phenotypic profiles.

A key use case for HCS in drug screens is the related problem of target prediction, that is, the prediction of the pathway or protein whose function is altered by the drug. This invokes a classification task on the image data, namely to determine what mechanism of action (MOA) is responsible for inducing the effects visible in the perturbed populations, often judged with reference to the control populations. Screening several cell lines with different genetic and transcriptomic profiles allows one to test more pathways (as not all genes are expressed in all cell lines) and, in so doing, to get a richer description of the MOA of the drug. Quantifying drug effects with respect to multiple cell lines allows to distinguish cell-line-specific drug effects from unilateral effects. A multi-cell-line screen, in which several cell lines representative of a common disease are subjected to the same set of perturbations, is therefore well motivated. However, such a screen creates the challenge of a data heterogeneity. For example, TNBC cell lines MDA231 and MDA468 manifest different morphologies in their unperturbed state. Figure 1.3 shows samples of cells from these two cell lines, taken from mi-

croplate wells containing negative control substance dimethyl sulfoxide, with the cells imaged by the same set of fluorescent markers. One observes clear differences in these archetypal morphologies, with MDA231 cells exhibiting elongated nuclear and cellular shapes, compared to the isotropic MDA468 cells. This has additional, second-order effects such as the degree of geometric tessellation between groups of cells. Note that in single-cell-line data, population phenotypes aggregated from the single cell constituents already abandon distributional information (Altschuler and Wu [2010]), as multi-modal populations are often reduced to their unrepresentative centroids. This problem is potentially exacerbated in multi-cell-line data and remains a challenge. Combining multi-cell-line image data has so far been unattended, with a handful of exceptions such as Rose et al. [2018] and Warchal et al. [2016].

1.2.2 The emergent role of deep learning in high content screening

Deep learning is touted as a panacea for computer vision problems, and high content data ought not to be an exception. In applications of deep learning, a neural network may perform several of the HCA stages (Figure 1.2) simultaneously, as neural networks naturally incorporate elements of feature extraction and dimensionality reduction (Sommer et al. [2017] is a good example). However, the successful deployment of deep neural networks relies crucially on vast volumes of data to enable effective generalisation. Large annotated datasets such as ImageNet (Russakovsky et al. [2015]) were one of a few key preconditions that fostered the rise of deep learning for object classification in 2012, and extensions of deep learning to other problem domains were likewise accompanied by the curation of large, special-purpose datasets, for example Lin et al. [2014] for object detection. However, annotated data for supervised training is expensive, requiring manual effort, often by domain experts. ImageNet leverages online crowdsourcing platforms (quality is ensured by the consensus of multiple annotators). This is a bottleneck for all deep learning research, and therefore extends to computational phenotyping. The Broad Institute benchmark collection (BBBC) datasets (Ljosa et al. [2012] and, specifically, Caie et al. [2010]) have become a sort of benchmark for developing drug response phenotyping algorithms (for example, Kraus et al. [2016] or Kandaswamy et al. [2016]). However, while benchmark data sets are of course useful and have had an enormous impact on the field, we still face the problem that for new imaging projects, we do not have enough data to train neural networks. This relates to the fact that bio-images tend to be extremely variable between different projects: the visual aspect is heavily influenced by the choice of markers and the mode of microscopy. Indeed, by selecting different markers, one is effectively look-

ing at different objects. For this reason, it seems unlikely that large scale datasets will definitively solve the problem of annotated data, except for the most widely used markers and imaging modalities.

Nevertheless, in recent years, a significant trend in deep learning research has been on making better use of available data. After all, even if annotated data is hard to come by, unlabeled or *weakly labeled* is available in abundance. For example, Mahajan et al. [2018] used 3.5 billion social media images “weakly annotated” with hashtags to pretrain a state-of-the-art system for object classification. Indeed, notable applications of deep learning in HCS thus far have relied on weakly supervised learning (Kraus et al. [2016], Godinez et al. [2017]). Elsewhere, contrastive learning (for example, Chen et al. [2020]) may yet revolutionise the training of deep learning systems, achieving parity with state-of-the-art systems with the efficient use of only a small fraction of the data.

A recent trend in bioimage analysis has been the prediction of one mode of microscopy from another. Microscopes with the capacity of registering multiple modes of microscopy simultaneously (for example, transmitted light and fluorescence images) automatically create an image-to-image translation dataset. Fluorescence labeling as a pixel-wise regression problem has been successfully demonstrated by Christiansen et al. [2018] and Ounkomol et al. [2018], where deep multi-task neural networks are furthermore capable of labeling multiple independent fluorescent channels simultaneously. Earlier, Sadanandan et al. [2017] used fluorescence to construct cell segmentation datasets automatically. These works have shown how one may exploit imaging protocols to bypass the manual annotation bottleneck for deep learning.

Generative models represent another trend in computer vision (Kingma and Welling [2013], Goodfellow et al. [2014a], Oord et al. [2016]), modeling the marginal distribution on data, allowing for data synthesis, in particular image synthesis, as well as unsupervised representation learning. Generative adversarial networks (GANs) (Goodfellow et al. [2014a]) are the most highly developed of deep generative models, and have already found use in high content image data, for example in Osokin et al. [2017]. Elsewhere, generative models such as variational autoencoders (Kingma and Welling [2013]) have found use in phenotyping drug effects for MOA prediction. Image-to-image translation (see above) is also addressed by generative models capable of deep style transfer (Isola et al. [2017], Zhu et al. [2017]). Data synthesis is additionally a form of data augmentation, which in turn is an attempt to make more effective use of a scarce data supply. Moreover, it is one possible route towards the simulation of image data (see Ihle et al. [2019]). Unlike more standard deep learning models, however, generative models are more difficult to train, and may require creative and non-standard solutions.

Therefore, as much as the role of deep learning in high content analysis is not yet fully defined, the role of generative models is ever more so, and their application represents a fertile research direction.

1.3 Contributions

This dissertation encompasses work completed on two unique projects in high content analysis. At is likewise organised into two parts, each containing two chapters. The two parts can be read in any order, but they internally follow a progression of ideas that are intended to be read in order of appearance. Each chapter is oriented around a published paper, with the exception of the final chapter, which describes a work in progress. In addition, the optional Chapter 2 serves as an introduction to the basics of deep learning, with an emphasis on the models encountered throughout the dissertation.

Part I, comprising of Chapters 3 and 4 covers a high content drug screen of multiple TNBC cell lines. TNBC is a molecularly heterogeneous type of breast cancer with poor prognosis and limited treatment options. Its molecular heterogeneity makes finding treatments difficult, and a screen of multiple cell lines is therefore promising. Chapter 3 is introductory and aims to describe the workflows of high content screening and its modes of analysis, while following use cases from the drug screen, including a multivariate analysis on *double strand break* detection, published in the proceedings of ISBI 2018 (Boyd et al. [2018]) (included in Appendix C. It concludes with a use case of *phenotypic profiling* as a segue to Chapter 4, which itself serves as a review of profiling methodologies, and develops a new profiling approach for multiple cell lines. The approach is to combine heterogeneous data from different cell lines using domain adaptation. Cells from the divergent cell line domains are mapped to a *domain-invariant* feature space by an adversarial neural network training strategy (Ajakan et al. [2014]). The chapter is an extended version of a paper published in the journal Bioinformatics (Boyd et al. [2020a]). The dataset for this project is additionally released publicly (Boyd et al. [2019a]) (it seems, the first of its kind), alongside an open source code repository including worked, reproducible workflows⁴. In spite of the idealised workflow presented in Section 1.1.2, Part I is restricted to the pilot phase of a planned larger screen. Nevertheless, ample phenotypes are observed and on which to develop the new methods.

Part II, comprising of Chapters 5 and 6 covers chimeric antigen receptor T-cell (CAR-T) therapy experiments, which study the Raji cell line as a model for lymphoma. Though not strictly based on a screen, Part II uses

⁴<https://github.com/jcboyd/multi-cell-line>

high content analysis and shares many characteristics with Part I. Chapter 5 compares two approaches to utilising fluorescence microscopy as an automatic annotator of paired phase contrast microscopy. The second of the two approaches, based on a customised object detection system, was published in the proceedings of ISBI 2020 (Boyd et al. [2020b]). The datasets are again made public (Boyd et al. [2019b]), as well as source code and scripts for reproducible experiments⁵. A modified version of this paper is included inline in Chapter 5. Finally, the shortcomings of the two approaches are assessed in Chapter 6, and a creative alternative solution is proposed, based on data augmentation via images synthesised with generative models. The final section of the chapter is intended as a prototype for a future publication.

Not included in this dissertation are minor contributions made as a second author to a study on predicting residual cancer burden from TNBC histopathology images, published in the proceedings of ISBI 2019 (Naylor et al. [2019]), and an as-of-yet unpublished paper on a new structured dropout algorithm for regularising neural networks Khalfaoui et al. [2019].

⁵<https://github.com/jcboyd/detecting-lymphocytes>

Chapter 2

Deep learning fundamentals

Summary: *This dissertation makes extensive use of artificial neural networks and deep learning. This chapter explores the basic properties of neural networks and shows how these ideas extend to the deep convolutional networks key to modern image processing. It further briefly traces the history of the development of deep learning, and its extension into the various problem domains of computer vision.*

Résumé: *Cette thèse fait largement appel aux réseaux de neurones artificiels et à l'apprentissage profond. Ce chapitre explore les propriétés de base des réseaux de neurones et montre comment ces idées s'étendent à la variété profonde et convolutionnelle des réseaux, clé du traitement moderne des images. Il retrace ensuite l'histoire du développement de l'apprentissage profond et de son extension dans les différents domaines de la vision par ordinateur.*

2.1 The building blocks of artificial neural networks

An artificial neural network (henceforth *neural network*) is a collection of computational units known as *neurons*, organised into a sequence of *layers*. An input passes *forward* through a neural network, undergoing a series of transformations through combination with a set of *weights* in each layer. During a training procedure, the neural network is shown examples of input data paired with target outputs. After each round of training, the neural network adjusts its (randomly initialised) weights so as to make it a little

more likely to emit the target values given future appearances of the input data.

A simple neural network invokes one or more *hidden* layers between input and output, for example,

$$\begin{aligned} h(\mathbf{x}) &= \sigma(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) && \text{(hidden layer)} \\ f(\mathbf{x}) &= \mathcal{S}(\mathbf{W}^{(2)}\mathbf{h} + \mathbf{b}^{(2)}) && \text{(output layer)} \end{aligned} \quad (2.1)$$

for input vector $\mathbf{x}$, first layer weights and biases matrix $\mathbf{W}^{(1)}$ and vector $\mathbf{b}^{(1)}$, and second layer weights and biases matrix $\mathbf{W}^{(2)}$ and vector $\mathbf{b}^{(2)}$. The function σ is a non-linear *activation function*¹ and $\mathcal{S}$ is the softmax function at the output layer. The softmax is optional but often used for classification problems as it maps its inputs to categorical probabilities. In this case, the network is normally trained against a cross entropy loss function,

$$\mathcal{L} = \sum_i -\log(p_{y_i}), \quad (2.2)$$

where $p_{y_i} = f(\mathbf{x}_i)_{y_i}$ is the network output (probability) for the i th training input $\mathbf{x}_i$, indexed at the (paired) i th target label y_i . Equation 2.2 is a simplification of the cross entropy formula for the case where the y_i are “one-hot”, that is, all the probability mass is allocated to a single, “true” value. Note that neural networks can be trained for classification, regression, or unsupervised feature extraction, depending on the structure of the dataset. Each entails a different output layer and loss function for the network. Figure 2.1 depicts an example of the simple neural network *architecture* in Equation 2.1.

The neural network framework gives freedom over the number of layers and the number of neurons in each layer. The more neurons, the more expressive the network, yet the more likely overfitting becomes². We may extend to a multi-layer network simply by stacking the desired number of layers,

$$f(\mathbf{x}) = \mathcal{S}(\mathbf{W}^{(M)}\sigma(\mathbf{W}^{(M-1)}(\dots\sigma(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})\dots) + \mathbf{b}^{(M-1)}) + \mathbf{b}^{(M)}). \quad (2.3)$$

¹Historically, this was the logistic function, $\sigma(x) = 1/(1 + \exp\{-x\})$, a continuous approximation to the Heaviside step function (think on/off), however, in recent years, it has been superseded by the ramp function, more commonly known as the rectified linear unit, $\text{ReLU}(x) = \max(0, x)$, which provides various training benefits.

²It is rare to require more than three fully-connected layers, however, and exceeding this amount will be done specifically to exploit deep hierarchical structures in the data.

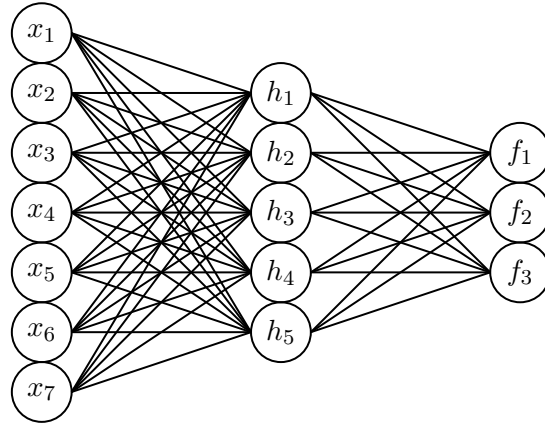


Figure 2.1: Fully-connected neural network with 7 input neurons, a single hidden layer with 5 neurons, and an output layer with 3 neurons.

Thus, the network layers are linear transformations interspersed with non-linear functions. When the inputs correlate positively with a vector of weights, the corresponding hidden layer neuron tends to become active via the non-linearity. In this way, each neuron of a hidden layer amounts to a feature detector for the prior layer. The hidden layer activations are then recombined in the next layer, and so on, allowing for ever more complex aggregations of features. This is the essence of deep learning. However, this tends not to work very well without certain inductive biases, (and, indeed, an amenable dataset) which we will encounter in Section 2.2.

Despite their inherent non-linearity, and non-convexity, neural networks can be trained with standard gradient descent,

$$\theta_{t+1} \leftarrow \theta_t - \alpha \cdot \nabla_{\theta} \mathcal{L}, \quad (2.4)$$

for the full set of model parameters θ and *learning rate* α . More sophisticated update rules than vanilla gradient descent exist such as RMSprop (Tieleman and Hinton [2012]), and Adam (Kingma and Ba [2014]). These methods improve over Equation 2.4 by adapting a learning rate for each parameter (rather than one-size-fits-all). Whatever the chosen method, the gradients are always computed using the backpropagation algorithm, which we examine presently.

2.1.1 Backpropagation

The procedure for computing the gradients at each iteration of gradient descent is called backpropagation. Backpropagation is an application of the

chain rule to the graphical structure of the neural network. The crucial formula for backpropagation, as presented in [Rumelhart et al. \[1985\]](#) is,

$$\frac{\partial \mathcal{L}}{\partial x_i} = \sum_j \frac{\partial \mathcal{L}}{\partial y_{ij}} \cdot \frac{\partial y_{ij}}{\partial x_i}, \quad (2.5)$$

that is, the gradient of neuron x_i at a given layer l is the sum product of the gradients of the neurons of the following layer $(l + 1)$ emanating from it y_{ij} and the gradient of the connections between them. From a practical standpoint, each layer l over which we perform backpropagation requires three computations:

1. $\frac{\partial \mathcal{L}}{\partial \mathbf{s}^{(l)}}$, the gradient of the *scores*, $\mathbf{s}^{(l)} = \mathbf{W}^{(l)}\mathbf{x}^{(l)} + \mathbf{b}^{(l)}$. Note this is a “pseudo-layer” between the linear transformation and the activation.
2. $\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(l)}}$, the gradient of the weights. These are recorded to ultimately make the descent step (Equation 2.4).
3. $\frac{\partial \mathcal{L}}{\partial \mathbf{x}^{(l)}} = \frac{\partial \mathcal{L}}{\partial \sigma^{(l-1)}}$, the gradient of the input (previous activation). This is the *error signal* that is passed back to the layer below.

Let us first consider (1), the gradient of the scores. In general, we compute, $\frac{\partial \mathcal{L}}{\partial \mathbf{s}^{(l)}} = \frac{\partial \mathcal{L}}{\partial \sigma^{(l)}} \cdot \frac{\partial \sigma^{(l)}}{\partial \mathbf{s}^{(l)}}$ where $\frac{\partial \mathcal{L}}{\partial \sigma^{(l)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{x}^{(l+1)}}$, since the activation of the present layer is the input to the following layer. The activation function σ is applied element-wise. Consequently, in the pseudo-layer between linear transform and activation, each neuron has a single connection. Therefore, with respect to Equation 2.5, the sum reduces to a single element per gradient giving,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{s}^{(l)}} = \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial \sigma_1^{(l)}} \cdot \frac{\partial \sigma_1^{(l)}}{\partial s_1^{(l)}} \\ \vdots \\ \frac{\partial \mathcal{L}}{\partial \sigma_k^{(l)}} \cdot \frac{\partial \sigma_k^{(l)}}{\partial s_k^{(l)}} \end{bmatrix}, \quad (2.6)$$

where for sigmoid, $\frac{\partial \sigma_i^{(l)}}{\partial s_i^{(l)}} = \partial \sigma_i^{(l)}(1 - \sigma_i^{(l)})$, and for ReLU, $\frac{\partial \sigma_i^{(l)}}{\partial s_i^{(l)}} = 1_{\{\sigma_i^{(l)} > 0\}}$.

For computation (2), the weights gradient, consider, $\frac{\partial \mathcal{L}}{\partial w_{kj}^{(l)}} = \frac{\partial \mathcal{L}}{\partial s_j} \cdot \frac{\partial s_j}{\partial w_{kj}^{(l)}} = \frac{\partial \mathcal{L}}{\partial s_j} \cdot x_j$. In vector form this gives us, $\frac{\partial \mathcal{L}}{\partial \mathbf{s}^{(l)}} \mathbf{x}^T$, that is, the outer product of the score gradient and the input vector. Considering that the loss over a batch is the sum of losses for each sample, in general we have for batch $\mathbf{X} \in \mathbb{R}^{D \times M}$,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(l)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{s}^{(l)}} \cdot \mathbf{X}^T. \quad (2.7)$$

For the bias terms, consider that with the bias trick, the gradients $\frac{\partial s_j}{\partial b_k^{(l)}} = 1, \forall k$. We can therefore simply sum the score gradients over the size m batch, $\sum_m \frac{\partial s_{m,j}}{\partial b_k^{(l)}}$.

Finally, for computation (3), simply consider the chain rule, $\frac{\partial \mathcal{L}}{\partial \mathbf{x}^{(l)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{s}^{(l)}} \cdot \frac{\partial \mathbf{s}^{(l)}}{\partial \mathbf{x}^{(l)}}$. It is easy to show by forming the Jacobian matrix that $\frac{\partial \mathbf{s}^{(l)}}{\partial \mathbf{x}^{(l)}} = \mathbf{W}^{(l)}$. Hence,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}^{(l)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{s}^{(l)}} \cdot \mathbf{W}^T. \quad (2.8)$$

This final gradient is passed to the previous layer as $\frac{\partial \mathcal{L}}{\partial \sigma^{(l-1)}}$ to continue the backpropagation. Upon traversing the network layers, the weight gradients are used to update the weights as in Equation 2.4.

2.2 Convolutional neural networks

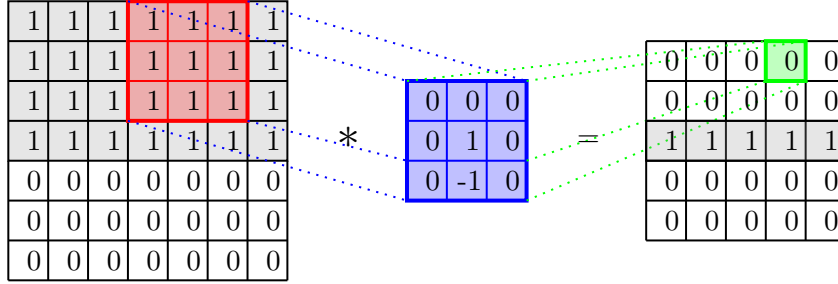


Figure 2.2: Convolution operation consisting of (from left to right) an input image, convolutional kernel, and convolved output image. The *valid* convolution results in the loss of the border pixels. Notice how the output represents the gradient image, obtained by setting the kernel to the finite difference between vertically adjacent pixels.

Convolutional neural networks (CNNs) are a family of neural network architectures having at least one *convolutional layer*. In image processing, a convolution between an image $\mathbf{I}$ and *kernel* $\mathbf{K}$ of size $d \times d$ and centered at a given pixel (x, y) is defined as,

$$(\mathbf{I} * \mathbf{K})(x, y) = \sum_{i=1}^d \sum_{j=1}^d \mathbf{I}(x + i - d/2, y + j - d/2) \times \mathbf{K}(i, j), \quad (2.9)$$

and is illustrated in Figure 2.2. Convolutions are useful for operations such as feature extraction and edge detection. A CNN is a neural network explicitly wired to perform convolutions³. In a convolutional layer, the elements of kernel $\mathbf{K}$ become learnable weights, replacing the large, fully-connected weight matrices from Section 2.1. In effect, the weights are *shared* across the input surface, greatly reducing the model dimensionality.

The earliest recognisable CNN is LeNet (LeCun et al. [1998]), bearing the name of its principal author, Yann LeCun. The destiny of LeNet is forever entwined with MNIST, the dataset it was developed to classify. MNIST is a 10-class image dataset of small (28×28 px) handwritten digits (0 – 9) in greyscale, crowd-sourced from US school students. The LeNet architecture can be written as,

$$\begin{aligned} \mathbf{H}_1 &= \sigma(\mathbf{X} * \mathbf{K}^{(1)}) && \text{(first convolutional layer)} \\ \mathbf{P}_1 &= \text{maxpool}(\mathbf{H}_1) && \text{(first pooling layer)} \\ \mathbf{H}_2 &= \sigma(\mathbf{P}_1 * \mathbf{K}^{(2)}) && \text{(second convolutional layer)} \\ \mathbf{P}_2 &= \text{maxpool}(\mathbf{H}_2) && \text{(second pooling layer)} \\ \mathbf{F}_1 &= \sigma(\mathbf{W}^{(1)}\mathbf{P}_2 + \mathbf{b}^{(1)}) && \text{(first fully-connected layer)} \\ \mathbf{F}_2 &= \sigma(\mathbf{W}^{(2)}\mathbf{F}_1 + \mathbf{b}^{(2)}) && \text{(second fully-connected layer)} \\ \mathbf{f}(\mathbf{X}) &= \mathcal{S}(\mathbf{W}^{(3)}\mathbf{F}_2 + \mathbf{b}^{(3)}) && \text{(output layer)} \end{aligned} \quad (2.10)$$

where for MNIST, the $\mathbf{X} \in \mathbb{R}^{28 \times 28}$ denotes an input image (two-dimensional array), $*$ denotes the convolution operation, and σ and $\mathcal{S}$ denote activation functions, as described previously.

The first convolutional kernels were originally 6 (5×5) kernels. The six convolutions are performed on the same grayscale input image, producing six *activation maps* that are concatenated across the channel dimension. As a result, the following convolutional layer originally had 16 ($5 \times 5 \times 6$) kernels. Notice the kernels are now three-dimensional tensors rather than arrays, so as to account for the multi-channel inputs. This represents a generalisation of Equation 2.9 in which each channel is convolved separately and the outputs are summed channel-wise. A depiction is given in Figure 2.3.

³A convolutional layer can nevertheless be represented as a sparse linear layer.

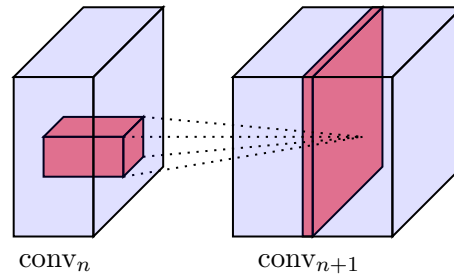


Figure 2.3: Each kernel of a convolutional layer (in red) performs a convolution as a tensor-product at each spatial location of an incoming tensor, to produce a single activation map (channel) in the output tensor.

The maxpool are pooling layers (in effect max filter operations), replacing each 2×2 grid of pixels with its maximum. The maxpool operations are usually performed at a *stride* of 2, resulting in an output halved in each spatial dimension. This ensures the most salient features are routed to the proceeding layers.

After the final pooling operation, the input tensor is implicitly flattened and traverses a series of fully-connected layers. The fully-connected layers were originally of size 120, 84, and finally 10 neurons, for the 10-way classification of MNIST digits. Note the numbers are fairly arbitrary and reflect the computational budget of the day. The architecture can be understood as having a feature extraction component in the convolutional and pooling layers, and a classification component in the fully-connected layers, with all layers trained end-to-end in unison using backpropagation and gradient descent. At each convolutional layer, activations arise from input features coinciding with the motifs embedded in the kernels. These are promoted by the max pooling layers, which simultaneously coarsen the representation. Thus, salient motifs discovered at lower levels, such as edges and corners, can be recombined and aggregated into shapes, and, later, semantic objects. [Zeiler and Fergus \[2014\]](#) confirmed empirically that such feature hierarchies are learned by deep CNNs.

2.2.1 AlexNet and the ConvNet revolution

The emphatic victory of AlexNet ([Krizhevsky et al. \[2012\]](#)) in the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC)⁴ heralded the inexorable rise of CNNs. While emulating the same basic design as LeNet, the AlexNet architecture was deeper (more layers) and wider (more neurons/kernels), and processed larger images ($224 \times 224 \times 3\text{px}$), sporting in

⁴1000 classes, over 1 million images

excess of 60 million parameters over 8 learnable layers. The breakthrough of AlexNet is often attributed to the conjunction of three technological developments:

1. The rise of graphical processing units (GPUs)
2. The sourcing of large image datasets such as ImageNet
3. The streamlining of neural network training e.g. more stable activation functions (ReLU), adaptive optimisers, regularisers such as dropout, and better weight initialisation or pre-training strategies.

With academic and industrial vision research in full swing, VGGNet⁵ (Simonyan and Zisserman [2014]) and GoogLeNet (Szegedy et al. [2015]) followed as ILSVRC 2014 frontrunners, pushing the state-of-the-art ever further, while achieving lasting fame. The competition winner, GoogLeNet, pioneered a more complex neural network design by engineering the *inception module*, a concatenation of differently-sized convolutions. Laying several inception blocks end-to-end constitutes the 22-learnable-layer GoogLeNet. VGGNet likewise strived for unprecedented network depth (up to 19 learnable layers) while advocating some simple design principles. For example, (padded) 3×3 convolutions are used everywhere, given that a stack of three 3×3 convolutions with C channels has a receptive field of size 7×7 on its inputs, despite having $27C^2$ weights, fewer than a single 7×7 kernel at $49C^2$.

A multitude of network architectures have since been proposed. Among the most influential is ResNet (He et al. [2016]), which introduced the residual block: a skip connection bypassing a stack convolutions. The motivation is to prevent the gradients of deeper layers from overwhelming all others early during training. The residual block thus facilitated the training of extremely deep networks of up to 1202 layers⁶. More recently, EfficientNet (Tan and Le [2019]) proposed a “compound scaling” design strategy to increase network depth, width, and resolution in unison to maximise the performance benefit. As of February 2020, the state-of-the-art in ImageNet is an EfficientNet variant. Table 2.1 displays results from the history of the ILSVRC challenge illustrating the advancement of minimisation of test error over time. Note that AlexNet was evaluated on the ILSVRC 2012 dataset, whereas all others were evaluated on a similar dataset from the 2014 competition. Also, while GoogLeNet was the ILSVRC 2014 challenge winner, it prevailed only with a complex ensemble approach; we provide single-model results only.

⁵Visual Geometry Group at Oxford University.

⁶However, the standard depths are 50, 101, and 152 layers.

Model	Year	Top-1	Top-5	No. Parameters
AlexNet	2012	40.7%	18.2%	60M
GoogLeNet	2014	-	10.1%	6.8M
VGG-19	2014	25.5%	8.0%	144M
ResNet-152	2015	22.2%	6.2%	60M
EfficientNet	2019	15.6%	2.9%	66M

Table 2.1: Single-model test errors on ImageNet for five groundbreaking CNNs. Note AlexNet was evaluated on the ILSVRC 2012 dataset, the others on ILSVRC 2014. Human performance has been estimated to be 5% Top-5 error (Karpthy [2014]).

2.3 Neural object detection

An object detection system f is a model or automated pipeline that both localises and classifies ontologised objects in images. Localisation is usually expressed as a bounding box, which the detector emits along with the class prediction as a tuple,

$$f : \mathbf{x} \rightarrow \{(x, y, w, h, c)\}, \quad (2.11)$$

for image $\mathbf{x}$, where x, y are the coordinates of the bounding box center, and w, h are its width and height. The value c denotes the class of the detected object. Figure 2.4 illustrates a set of object detections made on a custom image by a state-of-the-art system.

Whereas the ILSVRC datasets came to set the gold standard benchmarks in image classification, similarly rich datasets predominate for object detection. The Microsoft Common Objects in COntext (COCO) (Lin et al. [2014]) (80 classes + background) and the PASCAL⁷ Visual Object Classes (VOC) challenge (20 classes + background) datasets (Everingham et al. [2010]) are the two leading datasets against which state-of-the-art object detection systems are tested. Following the explosion of interest in CNNs as image classifiers in 2012, rapid progress was made virtually in parallel for neural object detectors. An early example is *Overfeat* (Sermanet et al. [2013]), appearing in 2013. The basic architecture of Overfeat was an AlexNet, refurbished as a fully-convolutional model⁸.

No family of object detectors is more celebrated, however, than the R-CNN series, the study of which is to be recommended to the student of deep learning.

⁷Pattern Analysis, Statistical Modelling and Computational Learning

⁸Replacing fully-connected with convolutional layers frees the network from the constraint of a fixed input size.

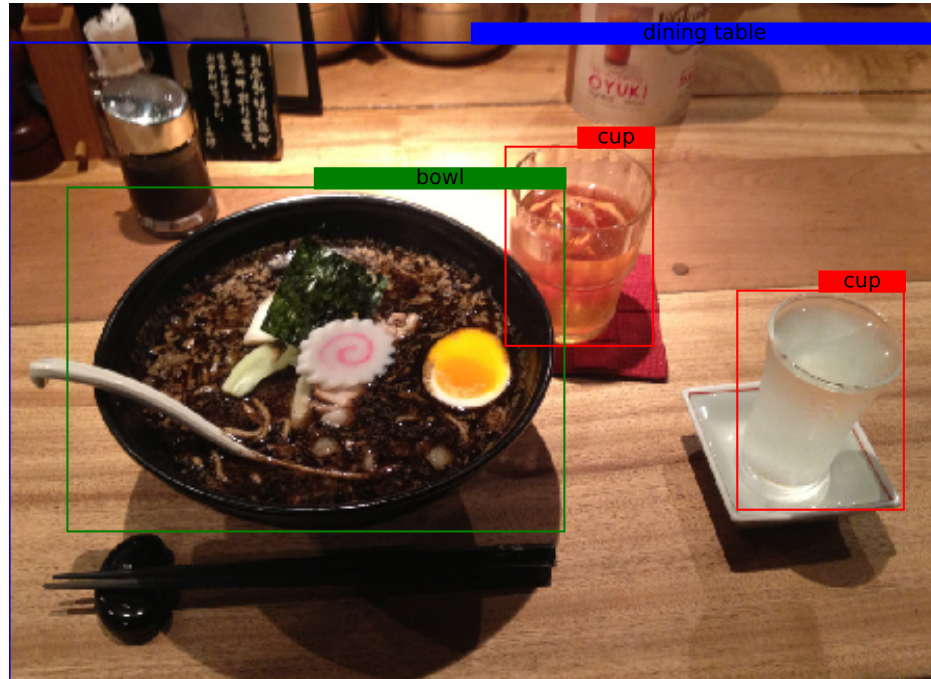


Figure 2.4: Object detections made by pre-trained R-CNN available in PyTorch (Paszke et al. [2017]). The image was taken in the Kyoto Gogyo ramen restaurant in Kyoto.

2.3.1 Regions with CNN features

Regions with CNN features (R-CNN) (Girshick et al. [2014]) began as a disjoint pipeline consisting of four ordered stages:

- (1) region proposal
- (2) feature extraction
- (3a) object classification
- (3b) bounding box correction

In the earliest iterations of R-CNN, the “selective search” algorithm played the role of stage (1)⁹. Feature extraction (2) was then performed by a pre-trained CNN, producing a “CNN code” feature vector for each region. Stage (3a) consisted of a set of linear models (typically SVMs) that perform one-vs-the-rest classification for each object class. These models were trained offline on the outputs of stage (2), “warped” to a standard size. Likewise, in

⁹A greedy algorithm that progressively merges homogeneous regions of an input image, from which derive a large number of bounding box proposals.

stage (3b), a set of per-class (ridge) regression models are trained to correct the region proposal coordinates (x, y, w, h) .

In Fast R-CNN (Girshick [2015]) stages (3a) and (3b) were absorbed into the CNN “backbone” of stage (2) as appendages to the neural feature extractor trained end-to-end. Now, entire images were processed in a single forward pass, with the region proposals cropped from the activation maps. The encoded regions were quantised by a generalisation of the max pooling layer, RoIPool. Faster R-CNN Ren et al. [2015] completed this work by integrating the region proposal algorithm (1) into the network. To this end, a separate region proposal network (RPN) was trained to predict both object presence and bounding box coordinates. A multi-stage training procedure was used to synchronise the weights between the two networks. Consequently, at prediction time, the feature extraction can be performed for both networks in a single forward pass. Then, the RPN heads produce the region proposals, which are passed to the heads of the R-CNN for detection. The overhead of region proposal is therefore negligible, and Faster R-CNN produces state-of-the-art object detection results more or less at real time. An even more recent variant, Mask R-CNN (He et al. [2017]) adds another stage (3c) to Faster R-CNN for segmentation, resulting in a system capable of the impressive task of *instance segmentation*.

2.4 Fighting overfitting in deep learning

In machine learning, overfitting is the effect of *fitting the noise instead of the signal*. In practice, all data contains noise that obscures the ground signal, and when a dataset is sufficiently small, a modestly powerful model may interpolate it perfectly, only to then be useless on independent test data. Much of machine learning is ultimately concerned with striking a balance between overfitting and underfitting. In supervised learning, this balance is evaluated with *generalisation error*, a measure of the ability of a model to generalise the data. This is estimated by evaluating a trained model over a test set of independent, unseen data. The *bias-variance decomposition* illustrates the tradeoff between over- and underfitting. Suppose we have $Y = f(x) + \epsilon$ generating training data with $f(x)$ the true signal for data point x , and noise, $\epsilon = \mathcal{N}(0, \sigma^2)$. Let our model estimate be denoted by $\hat{f}(x)$. Then, the MSE (here an arbitrary measure of goodness of fit) between an estimate and the true parameters, averaged over all possible data,

$$\mathbb{E}[(Y - \hat{f}(x))^2] = \mathbb{E}[((f(x) + \epsilon - \mathbb{E}[\hat{f}(x)]) + (\mathbb{E}[\hat{f}(x)] - \hat{f}(x)))^2] \quad (2.12)$$

$$= \sigma^2 + \mathbb{E}[f(x) - \mathbb{E}[\hat{f}(x)]]^2 + \mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2] \quad (2.13)$$

$$= \text{noise} + \text{bias}^2 + \text{variance} \quad (2.14)$$

where in step (1.2) some terms are eliminated with zero mean. Note the positive terms means this represents a lower bound on error. The bias term represents underfitting as a simplified model fails to follow a more complex trend (for example a linear model undershooting a higher-order function). This bias will be observed in training error. The variance term represents overfitting as the model varies around its mean greatly to interpolate noisy data. This variance will be observed in test error.

Neural networks, being such powerful “universal approximators” (Hornik et al. [1989]), are especially prone to overfitting. Many strategies to attenuate it have therefore been proposed. A common approach used elsewhere in machine learning is *weight regularisation*, often called *weight decay* in the neural net literature. This usually takes the form of a square error penalty term in the loss function. Parametric models that overfit usually require large parameter values (in the extreme, a degree n polynomial with up to $n - 1$ turning points can interpolate $n + 1$ data points), so regularisation curtails this tendency.

Deep learning succeeds because it embodies a different type of bias. Between layers of a vanilla fully-connected network, the number of weights is quadratic in the number of neurons per layer. Deep learning models reduce this dimensionality by spatial weight sharing, in the case of convolutional networks, and weight sharing in time, in the case of recurrent neural networks. These may be referred to as *structural biases*.

Other tricks to avoid overfitting include dropout (Srivastava et al. [2014]), whereby neurons are randomly zeroed-out during training so as to curtail complex co-dependencies; batch normalisation (Ioffe and Szegedy [2015]), where a normalisation operation is performed on the inputs of each layer so as to standardise their variance (and avoid “internal covariate shift”); and data augmentation, which aims to artificially increase the data supply. It is data augmentation that has particular relevance to this dissertation, and which is detail in the following.

2.4.1 Data augmentation

In the long run, as more data is added, a model with sufficient capacity improves its predictive power (generalisation error). In an ideal world, the data

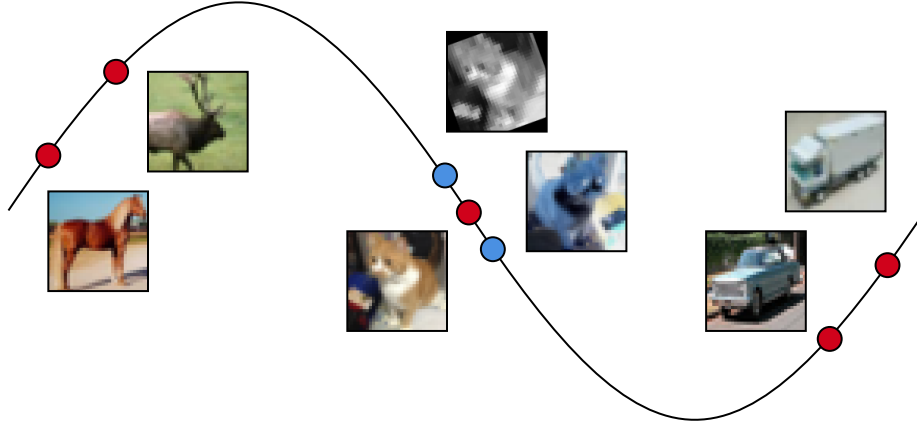


Figure 2.5: Data augmentation can make modest but useful interpolations of the space surrounding real images. CIFAR-10 images marked by red circle; augmented images marked by blue circle; black line represents natural image manifold. Upper augmentation based on conversion to grayscale (an element-wise weighted average of the RGB pixels) and rotation; the lower on colour inversion and horizontal flipping.

is unlimited. However, annotated data for supervised training is expensive, requiring manual effort, often by domain experts. Data augmentation is a technique for obtaining new data for fighting overfitting in machine learning models, thereby improving generalisation (test time) error ([Shorten and Khoshgoftaar \[2019\]](#)). The idea is to extract additional data *gratis* from the available dataset, by means of bootstrapping or interpolation. A classic algorithm is SMOTE (Synthetic Minority Over-sampling Technique) ([Chawla et al. \[2002\]](#)), which interpolates lines between training samples and their nearest neighbours in feature space, to create synthetic points, and is an alternative to over-sampling by replacement.

Data augmentation is especially vital in deep learning, where powerful neural networks require large datasets to train. Due to the geometry of natural images (pixels are ordered and neighbouring pixels are highly correlated), image data is uniquely receptive to data augmentation, and humans are well-equipped for the engineering of it. Images support a multitude of *label-preserving transformations*, that is, transformations of lower-level image properties that nevertheless maintain the high-level semantic content. Examples are cropping, flipping, rotating, noise injection, convolution, and colour space transformations (if colour is available). These may be referred

to as basic image manipulations (Shorten and Khoshgoftaar [2019]) and can be “stacked” (combined) endlessly, especially on the fly during training. Classic examples include image warping (LeCun et al. [1998]), elastic deformations (Simard et al. [2003]), and PCA-based colour space transformation (Krizhevsky et al. [2012]). However, data augmentation must not be mistaken as equivalent to adding authentic data and may reinforce existing biases in a dataset (Shorten and Khoshgoftaar [2019]). The augmented images are highly interdependent and will not in general amount to substantially increasing the coverage of space of all images (see Figure 2.5). Additionally, certain transformations will be domain-specific. For example, biomedical imagery can usually be rotated arbitrarily without losing meaning, unlike images of everyday objects or handwritten digits.

Deep learning itself can be used as a powerful, albeit expensive, mode of data augmentation. One good example is *adversarial training*. Goodfellow et al. [2014a] show how with a precise perturbation to a model input, one may “fool” a linear model or neural network alike. Consider perturbing model input $\mathbf{x}$,

$$\hat{\mathbf{x}} = \mathbf{x} + \alpha \cdot \boldsymbol{\eta}, \quad (2.15)$$

where $\boldsymbol{\eta} = \text{sign}(\mathbf{w})$ for $\mathbf{w}$ are the model weights of a softmax regression model and α a tuning parameter. Now, the model will predict $\sigma(\mathbf{w}^T \hat{\mathbf{x}}) = \sigma(\mathbf{w}^T \mathbf{x} + \alpha \cdot \|\mathbf{w}\|_1)$. If $\mathbf{x}$ is sufficiently high-dimensional (as image data often is) the perturbations will accumulate even for $\alpha \ll 1$ and can change the decision of the model. This produces a paradox especially apparent for image data: one may, for example, add a small, carefully chosen value to each pixel of an image, and the model may suddenly predict the wrong object class with high confidence, even though the perturbation remains imperceptible to the human eye. This paradox defies intuitions about how neural networks represent images and “adversarial attacks” remain problematic for deep learning. Goodfellow et al. [2014a] nevertheless showed how adversarial attacks could be harnessed during training to strengthen neural networks against such attacks, in effect a sort of data augmentation.

Generative models, in particular generative adversarial models (GANs) (Goodfellow et al. [2014b]) (loosely related to the above) are a more recent and exciting prospect for data augmentation. Generative models are trained to learn (perhaps implicitly) the data-generating distribution, and thereupon may be used as a sampling system for synthetic data. Ideally, a generative model based on deep neural networks will make for a more powerful interpolation system than any of the above.

One may also consider transfer learning as a form of data augmentation commuted via pretrained model. The information content of a rich source

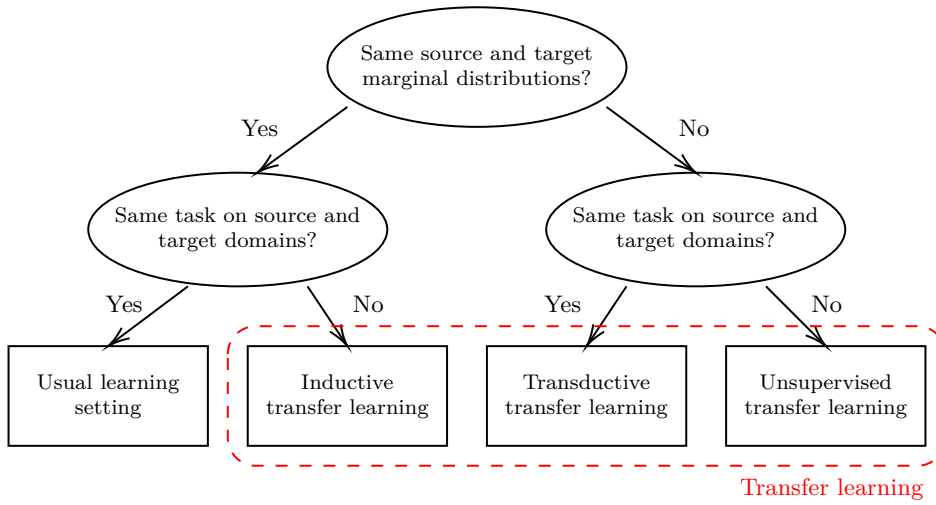


Figure 2.6: Positioning transfer learning with respect to the typical learning setting. Categories of transfer learning differ either with respect to the marginal distribution. Reproduced from https://en.wikipedia.org/wiki/Domain_adaptation

dataset $\mathcal{D}_S$ may be distilled by a powerful neural network, and transferred to augment the available data for learning some task on a target dataset $\mathcal{D}_T$. Indeed, style transfer involving deep, pretrained networks (Gatys et al. [2016]) is a powerful mode of data augmentation.

2.5 Transfer learning

Transfer learning is a set of solutions for learning problems in which training and test data differ in their marginal distribution, or else that the learning tasks differ between model training and testing (Pan and Yang [2009]). As such, these problems violate the basic assumption for training supervised models. Transfer learning bridges *domains* of learning. A domain $\mathcal{D}$ consists of a feature space $\mathcal{X}$ and marginal probability distribution on that domain $P(X)$. A learning *task* $\mathcal{T}$ is performed on a domain, itself consisting of a label space $\mathcal{Y}$ and some (usually unobserved) objective function $f(\cdot)$. In transfer learning problems one denotes source domain and tasks $\mathcal{D}_S$ and $\mathcal{T}_S$, and likewise for target domain, $\mathcal{D}_T$ and $\mathcal{T}_T$. The source is the data on which the model is initially trained, and the target is that for which it must be adapted.

Neural networks, with their endless flexibility, are well-equipped for transfer learning. Shortly after the deep learning revolution of 2012, the potential for the learning transfer of powerful pretrained neural networks was discovered

(see Zeiler and Fergus [2014] and Sharif Razavian et al. [2014]). Pretrained networks are often used in either of two ways:

1. feature extraction: a pretrained network is used to extract features for a new problem by passing images forward through its hidden layers and recording a given layer (“CNN code”) as a vector of (obscure) image features.
2. fine-tuning: a pretrained network recommences training on a new problem from a saved point of convergence, usually at a significantly lower learning rate, and often only on the uppermost layers.

The pretraining itself is usually done on a large standard image corpus like ImageNet. The success of these approaches illustrates the universality of the filters learned by deep networks, as general feature extractors for images. Neither approach fits easily into the schema of Figure 2.6. However, if for example the target images are similar in resolution and content then the low-level neural activity of earlier layers may be identical as for the source dataset, placing it near inductive transfer learning.

An interesting way to perform transductive transfer with deep learning is with *domain-adversarial neural networks*, which we examine presently.

2.5.1 Domain-adversarial neural networks

The inspiration for domain-adversarial neural networks (DANNs) lies in work published in Ben-David et al. [2010], which defines the $\mathcal{H}$ -divergence,

$$d_{\mathcal{H}}(D_S^X, D_T^X) = 2 \sup_{h \in \mathcal{H}} \left| P_{\mathbf{x} \sim D_S^X}(h(\mathbf{x}) = 1) - P_{\mathbf{x} \sim D_T^X}(h(\mathbf{x}) = 1) \right|, \quad (2.16)$$

that is, given a source domain (distribution), D_S^X (marginalised by the input variable), and a target domain D_T^X , and given a hypothesis class¹⁰ $\mathcal{H}$, the divergence between the source and target domains with respect to $\mathcal{H}$ is the classifier (here binary—for simplicity) that, proportionally, most classifies the domains into separate classes.

To illustrate, suppose we choose our hypothesis class to be a linear SVM. This class is capable (through choice of the SVM weights) of creating any linear hyperplane. Suppose the two domains were linearly separate in the feature space (see Figure 2.7). It would then be possible to train an SVM to perfectly separate the two, putting one domain fully into the positive class,

¹⁰Category of classifiers, e.g. linear SVMs.

and the other into the negative class. This would maximise the inner expression of the divergence formula, thus describing a high divergence. Should we have, on the other hand, two highly overlapping domains, any hyperplane cut through the middle of the point cloud would give us a divergence close to 0. Should we cut so as to isolate a few outlier points of a particular class, these would represent only a small proportion of the data, and the probability would also be close to 0. Intuitively, the $\mathcal{H}$ -divergence captures the effect nicely.

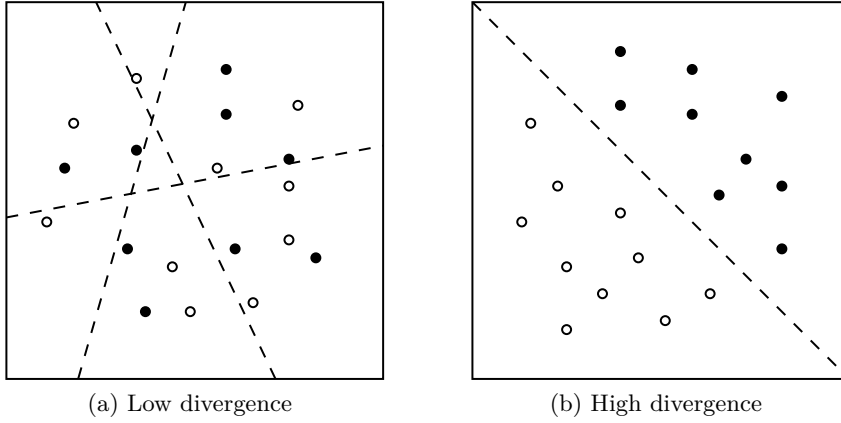


Figure 2.7: Overlapping domains (a) and divergent domains separated by a linear decision boundary (b).

Conveniently, it is possible to compute a consistent estimate of the $\mathcal{H}$ -divergence using finite data with the *empirical* $\mathcal{H}$ -divergence,

$$\hat{d}_{\mathcal{H}}(S, T) = 2 \left(1 - \min_{h \in \mathcal{H}} \left[\frac{1}{n} \sum_{i=1}^n \mathbf{1}[h(\mathbf{x}_i) = 0] + \frac{1}{n'} \sum_{i=n+1}^N \mathbf{1}[h(\mathbf{x}_i) = 1] \right] \right), \quad (2.17)$$

for samples, $S \sim D_S^X$ of size n and $T \sim D_T^X$ of size n' . For convenience, these are indexed from $1 \rightarrow n$ and $(n+1) \rightarrow N$, where $N = n + n'$. Though this may be hard to compute precisely in general, we can approximate this simply by training a classifier of the class $\mathcal{H}$ on the constructed dataset, $U = \{(\mathbf{x}, 0) : \mathbf{x} \in S\} \cup \{(\mathbf{x}, 1) : \mathbf{x} \in T\}$, that is, a classifier to differentiate between the domains. The estimated generalisation error, ϵ of this classifier could then be used to approximate the empirical domain divergence as $2(1 - 2\epsilon)$.

Ajakan et al. [2014] first formulate a DANN as a learning model of the form,

$$E(\theta_f, \theta_d, \theta_y) = \frac{1}{n} \sum_{i=1}^n \mathcal{L}_y^i(G_y(G_f(\mathbf{x}; \theta_f); \theta_y), y_i) + \lambda R(\theta_f, \theta_d), \quad (2.18)$$

where the G_f component acts as a feature extractor, and G_y is a classifier. These may have any neural architecture, shallow, deep, or convolutional. Recalling that the target data is unlabelled, they propose to define the regulariser of this model as,

$$R(\theta_f, \theta_d) = -\frac{1}{n} \sum_{i=1}^n \mathcal{L}_d^i(\theta_d, d_i) - \frac{1}{n'} \sum_{i=n+1}^N \mathcal{L}_d^i(\theta_d, d_i), \quad (2.19)$$

where $\mathcal{L}_d^i(\theta_d, d^i) = \mathcal{L}(G_d(G_f(\theta_f); \theta_d), d_i)$, $d_i \in \{0, 1\}$. Thus formulated, one has a loss maximising the negative of the minimisation term in the divergence formula. Therefore, adding this regulariser to the objective function, and maximising the objective with respect to its parameters, θ_d , gives a worse overall loss. At the same time, however, the feature parameters, θ_f , which are trained to minimise the objective, are chosen to minimise this maximisation of the regulariser by the divergence parameters, θ_d , thus maximising the minimising component of the divergence, thereby minimising the divergence. The feature parameters therefore play a dual role in both minimising classification loss, and minimising divergence (equivalently, maximising discrimination error), *adversarially* to the domain classifier. This promotes domain invariant features. Formally, we have,

$$\begin{aligned} (\theta_f, \theta_y) &= \arg \min_{\theta_f, \theta_y} E(\theta_f, \theta_y, \hat{\theta}_d) \\ \theta_d &= \arg \max_{\theta_d} E(\hat{\theta}_f, \hat{\theta}_y, \theta_d) \end{aligned} \quad (2.20)$$

which can be found as a saddle point of the update steps,

$$\begin{aligned} \theta_f &\leftarrow \theta_f - \mu \left(\frac{\partial \mathcal{L}_y^i}{\partial \theta_f} - \lambda \frac{\partial \mathcal{L}_d^i}{\partial \theta_f} \right) \\ \theta_y &\leftarrow \theta_y - \mu \frac{\partial \mathcal{L}_y^i}{\partial \theta_y} \\ \theta_d &\leftarrow \theta_d - \mu \frac{\partial \mathcal{L}_d^i}{\partial \theta_d} \end{aligned} \quad (2.21)$$

Note that θ_f is updated to minimise the classification loss, but maximise the domain classification loss. This is called the *adversarial step*. The main

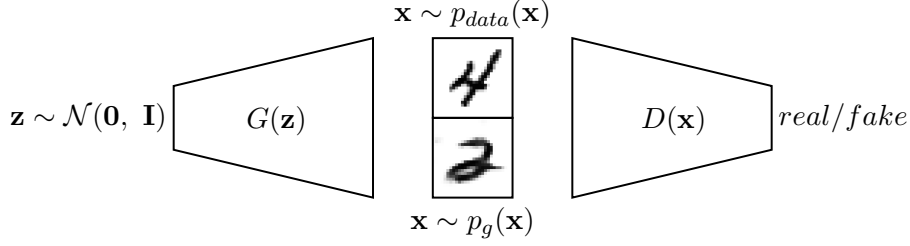


Figure 2.8: A GAN trains a generator network G to fool a discriminator network D in emitting counterfeit images $\mathbf{x} \sim p_g(\mathbf{x})$ by transforming noise input $\mathbf{z}$. In order to do so, G must (implicitly) learn the data-generating distribution, p_{data} .

contribution of [Ganin and Lempitsky \[2014\]](#) is to apply this to deeper architectures (in particular, convolutional), and to introduce a “gradient reversal layer” to enable a smooth implementation without altering the internal implementations of gradient descent (this still assumes that the local gradients can be specified arbitrarily) of deep learning frameworks. This is achieved with the pseudo-function,

$$R_\lambda(\mathbf{x}) = \mathbf{x}, \quad (2.22)$$

whose gradient is hard-coded as,

$$\frac{\partial R_\lambda}{\partial \mathbf{x}} = -\lambda \mathbf{I}. \quad (2.23)$$

This pseudo-layer is inserted between the feature extractor and the domain classifier in the model formulation. It should be emphasised that this is not a mathematical trick, but rather an implementation hack. The gradient reversal layer is easy to implement in all the major deep learning frameworks.

2.6 Generative adversarial networks

Generative adversarial networks (GANs) ([Goodfellow et al. \[2014b\]](#)) is a learning framework that trains a generator model to be capable of sampling new data from a data distribution (in particular, images), learnt (implicitly) from a training set. GANs compete with variational autoencoders ([Kingma and Welling \[2013\]](#)) and, more recently, autoregressive models ([Oord et al. \[2016\]](#)) for image generation. GANs are arguably a more mature and versatile learning paradigm, however.

The GAN framework (Figure 2.8) is *adversarial* in that it pits a pair of neural networks in a two-player minimax *game*: a *generator*,

$$G : Z \rightarrow X, \quad (2.24)$$

where in the simplest case $\mathbf{z} \sim Z$ is a vector of (uniform or Gaussian) noise and $\mathbf{x} \sim X$ is a synthetic data point (in particular, an image); and a *discriminator*,

$$D : X \rightarrow \{0, 1\}, \quad (2.25)$$

that is, a mapping from data point to binary value. D aims to minimise its rate of failure in discerning true data from “fake” data sampled from the the generator, which aims to maximise the error made by the discriminator,

$$\max_G \min_D \mathcal{L}_{GAN}(D, G) = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \quad (2.26)$$

The stationary point of Equation 2.26 is a *Nash equilibrium* (a saddle point) between the two objectives. Goodfellow et al. [2014b] guarantee that the distribution implicitly defined¹¹ by G , p_g is equal to p_{data} for G^* at (global) optimality and,

$$D^*(\mathbf{x}) = \frac{p_{\text{data}}(\mathbf{x})}{p_{\text{data}}(\mathbf{x}) + p_g(\mathbf{x})}, \quad (2.27)$$

where p_{data} is the data-generating distribution. In practice, GANs are trained by backpropagation with alternating gradient descent between D and G . The weights of D are frozen while propagating the errors back through D to G .

GAN architecture design must strike a balance between a generator expressive enough to approximate the data distribution, and a discriminator powerful enough to hold it to the task. GANs are afflicted with a phenomenon known as *mode collapse*. Mode collapse is the condition in which the generator “collapses” to generating few or even a unique output, regardless of the noise input $\mathbf{z}$. Metz et al. [2016] differentiate two varieties of mode collapse: *discrete mode collapse*, where the generator collapses to a subset of data

¹¹GANs are implicit generative models, in contrast to explicit models such as VAEs, which model the probabilities directly. GANs serve as an sample-emitting oracle emulating the true distribution.

modes (easily detected); and the more insidious *manifold collapse*, where the generator collapses to a subspace of the data distribution. They reason that mode collapse originates from the fact that in each iteration of training, the generator is impelled towards a delta function at the mode considered most “real” by the discriminator. The discriminator responds by lowering the probability of this mode, leading to oscillations in training. The “un-rolled” GAN training approach that they advocate mitigates this tendency by informing the generator in advance of the discriminator’s response to its prospective weight updates.

Another promising attempt to attenuate the mode collapse problem is to replace the Jensen-Shannon divergence loss function with the Earth Mover (EM) or Wasserstein-1 distance. Such is the strategy of Wasserstein GANs (WGANs). Intuitively, the EM distance measures a distance between two distributions. One can write the EM distance as the quantity $\mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim p(z)}[f_w(g_\theta(z))]$ maximised by choice of f_w (discriminator). The goal is then to minimise this distance through optimisation of the generator. The training algorithm is very similar to GANs. WGANs seem to be more stable, as the substituted loss function allows the discriminator to be trained to optimality. This greatly mitigates the mode collapse problem. Nevertheless, more subtle forms of partial mode collapse remain a recurrent problem for GANs. As a bonus, the discriminator loss (estimated EM distance) becomes meaningful during training, interpretable as image sample quality, rendering the implementation and fine-tuning of WGANs far easier.

2.6.1 Deep convolutional GANs

Radford et al. [2015] first succeeded in training more powerful, high-resolution GANs, by identifying a set of design principles: replacing pooling layers with strided convolutions; removing fully-connected layers (apart from the first layer of the generator); using batch normalisation after all but the last layer; using a tanh activation at the end of the generator (and ReLU everywhere else); and using LeakyReLU activations everywhere in the discriminator. For image data, deep convolutional GANs (DCGANs) represent a vast improvement over the original fully-connected GANs.

2.6.2 Conditional GANs

Mirza and Osindero [2014] present a simple extension to GANs allowing for control over the data generating process. One simply includes an additional input $\mathbf{y} \sim Y$ (for example, class information), to obtain conditional GANs (CGANs). That is, now the generator is conditional,

$$G : Z \times Y \rightarrow X, \quad (2.28)$$

as is the discriminator,

$$G : X \times Y \rightarrow \{0, 1\}, \quad (2.29)$$

and the CGAN is trained against the objective,

$$\mathcal{L}_{CGAN}(D, G) = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})}[\log D(\mathbf{x}|\mathbf{y})] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z}|\mathbf{y})))] \quad (2.30)$$

Mirza and Osindero [2014] do not offer so much as an intuition as to why conditional GANs works, but it can be an instructive thing to consider. As D learns to associate conditional information such as object class with the ground truth images, it will declare “fake” any image emitted by G not conforming to the condition, no matter the quality of the image. If G is to succeed, it will be forced to generate images in adherence to the condition. The outcome of successful training is therefore a generator that can, for example, synthesise an authentic-looking image of a desired class on demand.

The implementation of CGANs is a straightforward modification to GANs. The conditional information is concatenated to the standard inputs of both D and G and fed through to their hidden layers. A DCGAN discriminator may, however, concatenate the conditional information after its convolutional layers.

2.6.3 Assorted GANs

The success of GANs have led to many interesting results in recent years. Image-to-image translation has been achieved with `pix2pix` (Isola et al. [2017]), a GAN conditioned on a full image serving as a blueprint for the generated image. Such models are capable of an endless variety of applications, including segmentation and image colourisation. CycleGANs (Zhu et al. [2017]) achieve *unpaired* image-to-image translation, learning to map between image domains in an unsupervised way. Applications include style transfer. RecycleGANs (Bansal et al. [2018]) achieve the same for video-to-video retargeting.

Elsewhere, progressive growing of GANs (Karras et al. [2017]) produce dazzling high resolution outputs (up to 1024×1024 pixels). Here, training is

done in stages, beginning at low resolutions (4×4), and doubling the resolution at intervals. This is done by dynamically adding new layers to the generator and discriminator, which maintain symmetry throughout training. Thus, the GAN learns global structures first at low resolution and progressively refining its output.

Part I

Computational phenotyping for multiple cell lines

Chapter 3

High content analysis in drug and wild type screens

Summary: *This chapter puts the high content analysis pipeline into practice on two image datasets: a drug screen and a wild type screen of triple negative breast cancer cell lines. The implementation of a conventional pipeline for computational phenotyping is described in phases of cell segmentation and feature extraction. A collection of use cases are then given on each of the screens, covering the full range of univariate, multivariate, and machine learning-based analyses identified in previously, with the aim of understanding the morphological properties of the cell lines, and the effects of drug perturbation thereupon. This lays the groundwork for the phenotypic profiling of the following chapter.*

Résumé: *Dans ce chapitre, le pipeline d'analyse à haut contenu est mis en pratique sur deux ensembles de données d'images : un criblage de drogues et un criblage de type sauvage de lignées cellulaires de cancer du sein triple négatif. La mise en œuvre d'un pipeline classique pour le phénotypage par ordinateur est décrite dans les phases de segmentation cellulaire et d'extraction de caractéristiques. Un ensemble de cas d'utilisation est ensuite décrit sur chacun des cribles, couvrant la gamme complète des analyses univariées, multivariées et basées sur l'apprentissage machine identifiées au chapitre précédent, dans le but de comprendre les propriétés morphologiques des lignées cellulaires et les effets de la perturbation par la drogue sur celles-ci. Ceci pose les bases du chapitre suivant.*

3.1 Overview

Triple-negative breast cancer (TNBC) is a variety of breast cancer whose cells do not express proteins *estrogen receptor* (ER), nor *progesterone receptor* (PR), nor do they amplify *Human epidermal growth factor receptor 2* (HER2/neu). As these genetic markers are common targets for cancer therapies, TNBC is more difficult to treat than other breast cancers, being, in particular, unresponsive to hormone therapies (Hudis and Gianni [2011]). As a result, there are limited treatment options for TNBC, and chemotherapy remains the main treatment. TNBC is responsible for 10%-15% of breast cancers (Chavez et al. [2010]) and has a poorer 5-year survival rate than other forms of breast cancer (Gonçalves Jr et al. [2018]). TNBC exhibits molecular heterogeneity at the level of multiple subtypes Hatem et al. [2016].

Two datasets are considered in the following analysis on TNBC cell lines. The first is a pilot drug screen study on two TNBC cell lines (MDA231 and MDA468) (Filmus et al. [1985]). The second dataset is a wild type screen of 12 cell lines (Chavez et al. [2010]) (11 TNBC cell lines and a negative control), that is, without perturbation. Part I of this dissertation focusses primarily on the drug screen data, but both datasets feature in the following analyses.

3.2 Datasets

The drug screen pilot data set (Table 3.1) is an assay of two microplates each housing a grid of 384 (18×24) wells, labeled A01-P24, (alphabetic character denoting the row of the well on the plate, numeric denoting the column). One plate tests TNBC cell line MDA231, the other, MDA468. Wells were seeded to confluence with a controlled 1000 and 1250 cells for MDA231 and MDA468 cell lines respectively. The drugs are allocated according to a *plate map* (given in full in Figure A.2), which is applied for both plates. 36 wells are treated with the neutral agent dimethyl sulfoxide DMSO as a negative control, 2 with positive controls (Olaparib, Cisplatin), 166 with test compounds (concentration $10\mu M$), and 184 untreated (denoted empty). Following hibernation, the cells were fixed, washed, and stained with four fluorescent markers. A final washing procedure is performed, where extraneous dye content is aspirated from the wells, finally preserving only the chemically-bound dye compounds excited during the microscopy, and further evacuating all cells unfastened to the well, as a consequence of perturbation or otherwise. This step has consequences for the analysis, as it may remove cells dead prior to fixation, ostensibly leaving an absence of cells as a proxy

Plate/cell line	Well, field	Channel	Acquisition Mode
MDA231, MDA468	Plate row (A-P), column (1-24), well field (1-4) e.g. A01_01	DAPI, Cy3, Cy5, FITC	2D, 2D-Decon, Adv2D-Decon, 2.5D

Table 3.1: The drug screen pilot data, consisting of two cell lines on separate 384-well plates, with four fields per well. Four fluorescence channels are captured, under four acquisition modes. The full data set of 49152 images is the outer product of each of the table fields.

Stain	Marker
DAPI	A-T regions of DNA
Cyanine 3 (Cy3)	DNA double-strand bbreaks
Cyanine 5 (Cy5)	β -tubulin of microtubules
FITC	Phospholipids in cell membrane

Table 3.2: The stains used in the fluorescence microscopy of the screen and their corresponding biological markers.

for high levels of apoptosis, indicating a cytotoxic compound. The array of perturbations show a range of effects on cell mortality, from no apparent visual effect to a near or complete elimination of cells. The drugs comprise of a set of panels of kinase, protease and phosphatase inhibitors and can be categorised into 70 mechanism of action (MOA) classes of varying sizes, according to their targets.

Fluorescence microscopy ($20\times$ magnification widefield with deconvolution image restoration) was performed at four non-overlapping fields of view per well, producing two experimental datasets of 1536 multiplexed images apiece. The stains used in the screen are listed in Table 3.2. *DAPI* (4',6-diamidino-2-phenylindole) is a bright blue stain that permeates a cell's nuclear membrane and binds to regions of DNA rich in A-T base pairs. *DAPI* is effective in highlighting the cell nucleus, where the DNA is housed. Cyanine 3 (Cy3) and cyanine 5 (Cy5) are widely used dyes belonging to a common family. Here, Cy3 is conjugated to a γ -H2AX antibody. γ -H2AX is a protein that naturally binds to double-strand breaks as a marker for the repair of DNA damage. Furthermore, Cy5 is conjugated to a β -tubulin antibody, thus highlighting this sub-component of the tubulin polymer that constitutes microtubules, which are structure-providing components of cells. Finally, FITC (fluorescein isothiocyanate) highlights phospholipids, molecules in the lipid cellular membrane. Figure 3.1 shows a composite of *DAPI*, Cy5, and Cy3 fluorescence channels for an indicative field from the drug screen.

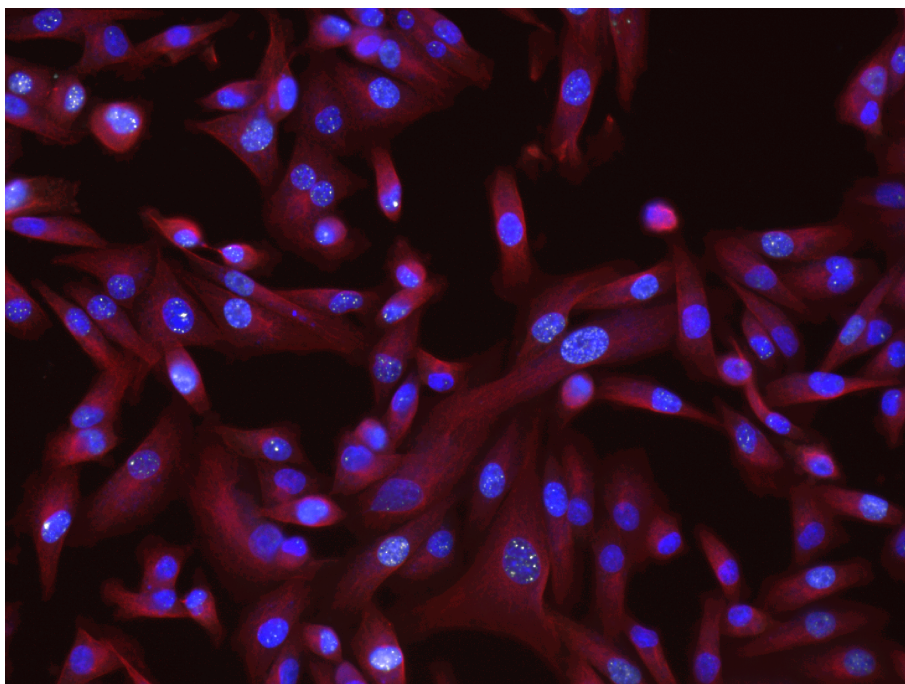


Figure 3.1: Fluorescence image of MDA231 cells. DAPI highlighting the nuclei in blue, cyanine 5 highlighting the microtubules in red, and cyanine 3 highlighting the double-strand breaks as white spots on the cell nuclei.

The microscopy was performed in various *acquisition modes*, whereby the imagery is captured in different ways. These are: 2D (raw camera output), 2D + deconvolution (2D with blur reduction), Advanced 2D + deconvolution (combination of image 3-stack), and 2.5D acquisition and deconvolution (aggregation over 3D image stack). Each mode captured the same cells at virtually the same time. The 2.5D mode was chosen for all analysis contained in Part I as it exhibited the sharpest detail from manual visual inspection.

3.2.1 Wild type screen dataset

In the wild type screen, 12 cell lines were imaged, including 11 TNBC cell lines (see Table 3.3), distributed evenly over a 96-well plate (8 rows, 12 columns). In contrast to the drug screen, the objective was to monitor cell wild type morphologies, therefore no chemical compounds were included in the wells for the screen. The cell line MCF10A is immortalised but non-tumorigenic, and serves as a negative control. The dataset is significantly smaller than that of the drug screen, albeit with five fields of view imaged per well. The same fluorescent markers used in the drug screen are used except

Cell line	Well, field	Channel
MDAMB231, MCF10A (negative), Hs578T, MDAMB157, HCC1143, HCC38, MDAMB468, HCC1937, MDAMB436, BT20, BT549, HCC70	Plate row (A-H), column (1-12), well field (1-5). Each cell line occupies one column.	DAPI, Cy3, Cy5, FITC rows A-D, <i>rhodamine</i> rows E-H

Table 3.3: The wild type screen data, consisting of 12 cell lines evenly distributed over a 96-well plate (ordered by column). Four fluorescence channels are captured as in the drug screen, albeit with FITC replaced by rhodamine in the final four rows.

for the fourth channel, which uses the FITC marker for half the plate, and rhodamine for the other half to monitor expression of tumour-suppressor protein p53.

Table 3.4 collates properties of the 12 cell lines, from which one can begin to appreciate the heterogeneity at play. The MDA family derive from pleural effusions, that is, from a fluid buildup in the patient, while all others derive from primary tumours. The molecular classification Basal A characterises cells that are more luminal while Basal B are more basal (Dai et al. [2017]). These imply different functions of the cells. Not listed are relevant information on common tumour suppressor genes: all cell lines (aside from the negative) exhibit a p53 mutation; contrarily, most exhibit wild type BRCA1¹, aside from cell lines HCC-1937 and MDA-MB-436. For the curious reader, the cell lines are prefixed generally according to the research group behind their discovery: BT (Breast Tumour) (Lasfargues and Ozzello [1958]); HCC (Hamon Center) (Gazdar et al. [1998]); HS (Hackett + Smith) (Hackett et al. [1977]); MCF (Michigan Cancer Foundation) (Soule et al. [1990]); MDA (MD Anderson Cancer Center) (Brinkley et al. [1980]).

3.3 Cell measurement pipeline

As mentioned in Section 1.1.3, the basis of high content analysis are the *features* measured on individual cells. The conventional path to measuring cellular features entails an initial segmentation of individual cells. This section describes the segmentation strategy as well as the features extracted for both the drug and wild type screen datasets. These procedures underpin standard preliminary analyses of the data (Section 3.4), as well as the developments described in later chapters.

¹Breast cancer type 1 susceptibility protein

Cell line	Site ¹	Pathology ²	Molecular classification
BT-20	PT	IDC	Basal A
BT-549	PT	IDC	Basal B
HCC-38	PT	IDC	Basal A
HCC-70	PT	IDC	Basal A
HCC-1143	PT	IDC	Basal A
HCC-1937	PT	IDC	Basal A
HS-578T	PT	IDC	Basal B
MCF-10A	NB	Fibrocystic	Basal B
MDA-MB-157	PE	IMC	Basal B
MDA-MB-231	PE	AC	Basal B
MDA-MB-436	PE	IDC	Basal B
MDA-MB-468	PE	AC	Basal A

Table 3.4: TNBC cell lines and negative control MCF-10A properties referenced from [Chavez et al. \[2010\]](#). Drug screen cell lines indicated in bold. ¹Site: NB, normal breast; PE, pleural effusion; PT, primary tumour. ²Pathology: AC, adenocarcinoma; IDC, infiltrating ductal carcinoma; IMC, infiltrating medullary carcinoma.

3.3.1 Nuclei segmentation

The cell nucleus is the logical starting point for cell segmentation (see Figure 3.2). Roughly speaking, most nuclei are visible, uniformly-sized, and ellipsoidal, and, crucially, seldom overlap, as the surrounding cell membrane acts as a buffer to other cells, and as the cell density has been chosen such that cells do not grow on top of each other. As a result, they are typically the easiest thing to accurately segment in the fluorescent stack, and may then be used as the starting point for segmenting other cytological components. The segmentation typically follows three stages: pre-filtering, background subtraction, and refinement.

Pre-filtering

A common approach to pre-filtering is median filtering ([Huang et al. \[1979\]](#)), which can be very effective in eliminating noise. The approach works as a sliding window over the image. Thus, for image $f : (X \times Y) \rightarrow V$ and window $W(x, y)$ centered on (x, y) of size $m \times n$, one creates a pre-filtered image p with,

$$p(x, y) = \text{median}\{f(x', y') | (x', y') \in W(x, y)\}. \quad (3.1)$$

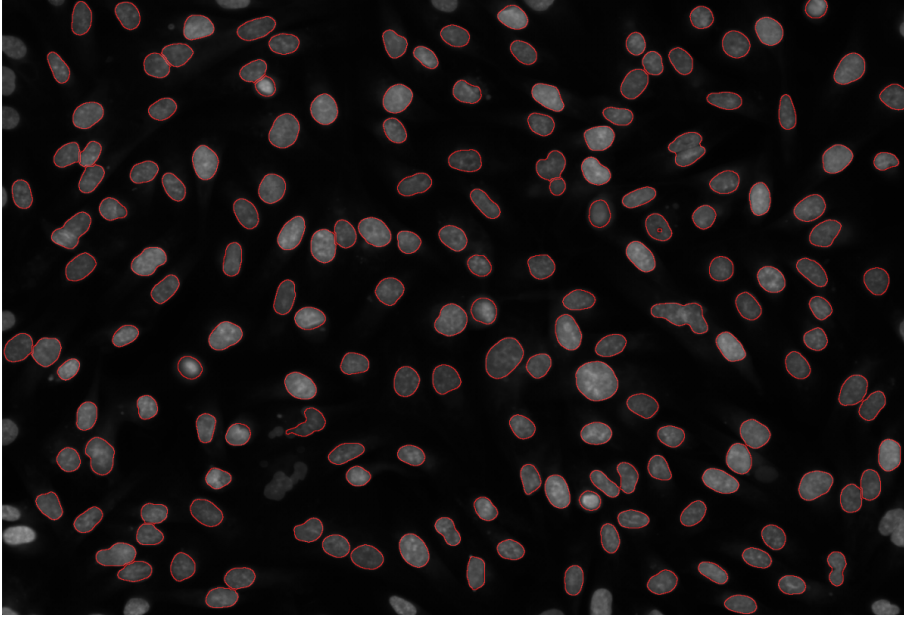


Figure 3.2: Nuclei segmentation on the DAPI channel, with segmentation contours indicated by red bands.

Note the median is an appropriate operator to use as it is robust to outliers, unlike the mean. Note also, however, that the elimination of noise may come at the cost of linking separated objects as, depending on the size of the filter, the pixels between nearby objects can be interpreted as outliers.

Background subtraction

In order to detect the contours of the cell nuclei, one takes the pre-filtered image p and performs a background subtraction. This again works with a sliding window, W , of size $m \times n$, which calculates the local mean intensity for a pixel $b(x, y)$, giving,

$$b(x, y) = \frac{1}{mn} \sum_{i=-\lfloor m/2 \rfloor}^{+\lfloor m/2 \rfloor} \sum_{j=-\lfloor n/2 \rfloor}^{+\lfloor n/2 \rfloor} p(x+i, y+j), \quad (3.2)$$

with which one can perform the subtraction yielding,

$$r(x, y) = \max(0, p(x, y) - b(x, y)). \quad (3.3)$$

Note that values in r will now be close to zero, with non-zeros corresponding

to regions of high intensity gradient. One can therefore identify the contours of the cells, $c(x, y)$ by applying a threshold t giving,

$$c(x, y) = \mathbb{1}\{r(x, y) \geq t\}. \quad (3.4)$$

The nuclei contours may be filled to produce a final segmentation mask.

Refinement

In the ideal case of well-separated cells, the aforementioned procedures might be sufficient for perfect nuclei segmentation. However, in practice, some additional steps are required to segment overlapping cells. One of the most enduring refinement approaches is the *watershed* algorithm (Beucher and Lantuéjoul [1979]) applied to the inverse distance map of the segmentation result². Informally, the watershed algorithm simulates a flooding of the topography of the intensity levels in an image. The meeting points of the growing regions build the *watershed line*, which amounts to a separating line between objects. This method partitions the initially identified object into as many regions as there are local minima of the inverse distance map. In order to avoid over-segmentation due to small irregularities in the object boundary potentially leading to local minima, one selects local minima according to their morphological dynamic (Grimaud [1992]).

3.3.2 Cell membrane segmentation

With the cell nuclei segmented on the DAPI channel, one may then segment the cell cytoskeleton, approximated by the microtubules on the Cy5 channel. Such is the strategy of Jones et al. [2005], who use the cell nuclei as seeds for the more difficult cell membrane segmentation. The approach consists of defining a specialised distance metric that accounts for changes in intensity (high gradients), with the matrix,

$$\mathbf{G} = \frac{1}{1 + \lambda} \cdot \begin{bmatrix} (\frac{\partial g}{\partial x})^2 + \lambda & \frac{\partial g}{\partial y} \cdot \frac{\partial g}{\partial x} \\ \frac{\partial g}{\partial x} \cdot \frac{\partial g}{\partial y} & (\frac{\partial g}{\partial y})^2 + \lambda \end{bmatrix}, \quad (3.5)$$

where $\mathbf{g}(\mathcal{I})$ is Gaussian blur, whose gradients are simply the finite central difference of the adjacent pixels (these too can be defined by convolutions) and λ is a regularisation parameter. Under this metric, infinitesimal distances are calculated as,

²The distance map is the image formed by the distances of the shortest path to the background for every foreground pixel in a segmentation mask.

$$||d\mathbf{x}||_{\mathbf{G}}^2 = d\mathbf{x}^T \mathbf{G} d\mathbf{x}. \quad (3.6)$$

The distance metric is defined for neighbouring pixels. Distances between non-adjacent pixels are computed as the shortest path, for example with Dijkstra’s algorithm. Thus, the metric behaves like a Euclidean distance metric on flat, uniform regions, yet takes intensity into account where necessary. The segmentation is performed by assigning pixels to the nearest seed, that is, cell nucleus, under the metric. One may see from Equation 3.5 that the metric converges to Euclidean distance as $\lambda \rightarrow \infty$. Hence, the segmentation becomes a Voronoi tessellation in the limit. The tunable parameters of the algorithm are: the size (receptive field) of $g\mathcal{I}$; an initial global thresholding step to eliminate the background, for example with an Otsu threshold (Otsu [1979]); and the regularisation λ .

3.3.3 Feature extraction

In order to quantify cellular phenotypes, features are extracted from the segmented cells. These features can be interpretable or general shape, intensity and texture descriptors. In this base line approach, the features used were provided by standard open-source software tools, here CellCognition³. The software can export many hundreds of features for each fluorescent channel at the user’s discretion. These features generally fall into the following categories: basic intensity features, basic shape features (e.g. RoI size, perimeter, circularity), convex hull features, distance map features, granulometry features, Haralick features, moments, and statistical geometric features (see Held et al. [2010], Walter et al. [2010a] for a detailed description).

In these experiments, 238 features were extracted for each of the DAPI and Cy5 channels, and 38 for the Cy3 channel, giving 516 features in total for each cell. These features are engineered to be plausibly discriminative, yet rarely have an immediate biological sense. Nevertheless, a minority of features can inform about some basic biological phenomena: the size of each segmented object (RoI size) on the DAPI channel directly quantifies nuclear size, from which one may infer about cell growth and aberrant phenomena such as chromosome segregation problems and consequent micronucleation; average DAPI intensity can indicate an interphase cell prior to or after DNA replication; and spot features directly measure double-strand breaks (DSBs) on the Cy3 channel (see Section 3.4.3).

³<http://cellcognition-project.org>

3.4 Use cases in high content analysis

The following section details a series of use cases of explorative analysis in the drug and wild type screens. Section 3.4.1 first investigates the possible presence of spatial biases in the drug screen microplates. Sections 3.4.2 and 3.4.3 in turn describe a univariate and multivariate analysis of the drug screen. Finally, machine learning techniques are used in Section 3.4.4 to make morphological comparisons of TNBC cell lines in their wild type.

3.4.1 Controlling for spatial effects in the drug screen dataset

One concern in HCS is the presence of systematic bias in the experimental setup. Controlling for this bias is the objective of a series of quality control techniques. In the pilot drug screen dataset, with a single plate per cell line, the most relevant concern is the presence of *spatial bias*, that is, a dependency of the phenotypic readout on the position of the well on the plate. A common cause of such a bias is a temperature gradient inside the incubator during hibernation. This can result in an uneven evaporation of well solvents, in particular around the microplate borders, leading to an alteration of the drug concentration. Where spatial biases exist, their effects might be corrected for (Caraus et al., Ljosa and Carpenter). However, the best solution is without question to reconduct the experiments, as statistical correction only addresses effect strength, not phenotypic alterations (for example, if more cells are dead, one cannot infer what would have happened had the concentration been lower).

Spatial biases are investigated in Figure 3.3. These figures visualise, for both cell lines, firstly the number of cells in each of the 384 wells of the plate (indexed first by the alphabetic vertical axis, then by the horizontal numeric axis). The cell count relates directly to viability, a key readout for drug potency. Also visualised are, in turn, the first and second principal components of the matrix formed from the vectors of mean feature readouts for the cells of each well. Note that these vectors amount to a simple phenotypic profile of the well condition (a construct that will be explored in great detail in Chapter 4). The principal components of the profile matrix (taken separately) therefore provide a univariate summary of phenotypic information for each well. Small clusters appear in each heat map reflecting the clustering of similar negative controls, however there is no apparent border effect or directional tendency that would indicate systematic bias for any of the three readouts. The validity of the screen data may therefore be concluded.

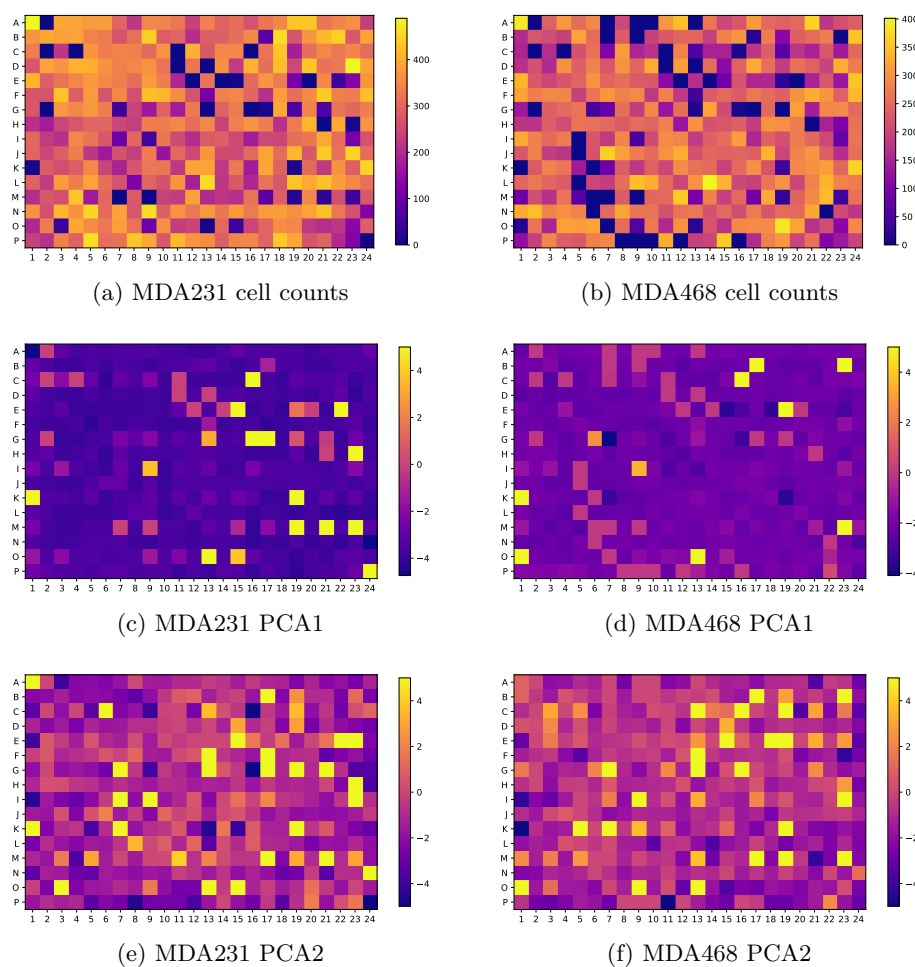


Figure 3.3: Searching for spatial biases: comparison of cell counts (a), (b); comparison of first principal component of morphological profiles (c), (d); comparison of second principal component (e), (f), arranged according to the plate map of the microplate. Left column (a), (c), and (e) pertain to cell line MDA231; right column (b), (d), (f) to cell line MDA468.

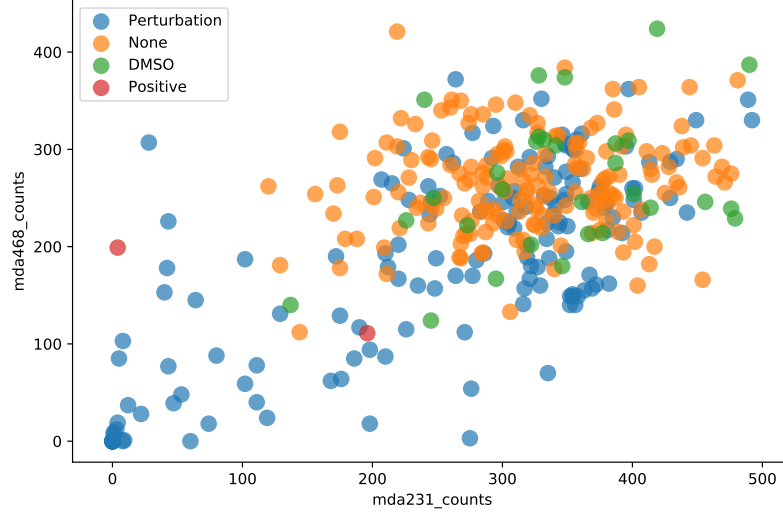


Figure 3.4: Viability comparison between cell lines for a variety of drugs and negative controls. The neutral DMSO and untreated wells strongly overlap, while showing considerable variation in both cell lines. Viability correlates well between cell lines with Pearson correlation coefficient $\rho = 0.6556$

3.4.2 Viabilities of TNBC cell lines correlate

One of the initial aims of this TNBC pilot drug screen is simply to quantify cell *viability*. In a viability assay, viability is conventionally measured in the range $[0, 1]$ (Pegg [1989]). Given that cells are seeded at a fixed density, the change in the number of dead cells with respect to the negative control is an indication of the viability. The viability,

$$v^d = \frac{n^d}{n^-}, \quad (3.7)$$

of cells in response to a drug d refers to the change in the population size n^d relative to that of the negative control n^- . Note that this can in principle exceed a value of 1. Due to the effects of lysis (cellular breakdown) and the washing procedure, the cells remaining at the point of imaging are the “survivors” of the drug perturbation. When cells are segmented, the viability of a drug is given by the cell sample size. When cells are not explicitly segmented, the viability is represented indirectly by the foreground density of the field. Viability is an important phenotypic characteristic of a drug effect. In Figure 3.4 we may compare the number of segmented cells (therefore a proxy viability) between well populations of cell lines MDA231 and MDA468 subjected to the same perturbations. The majority of viabilities

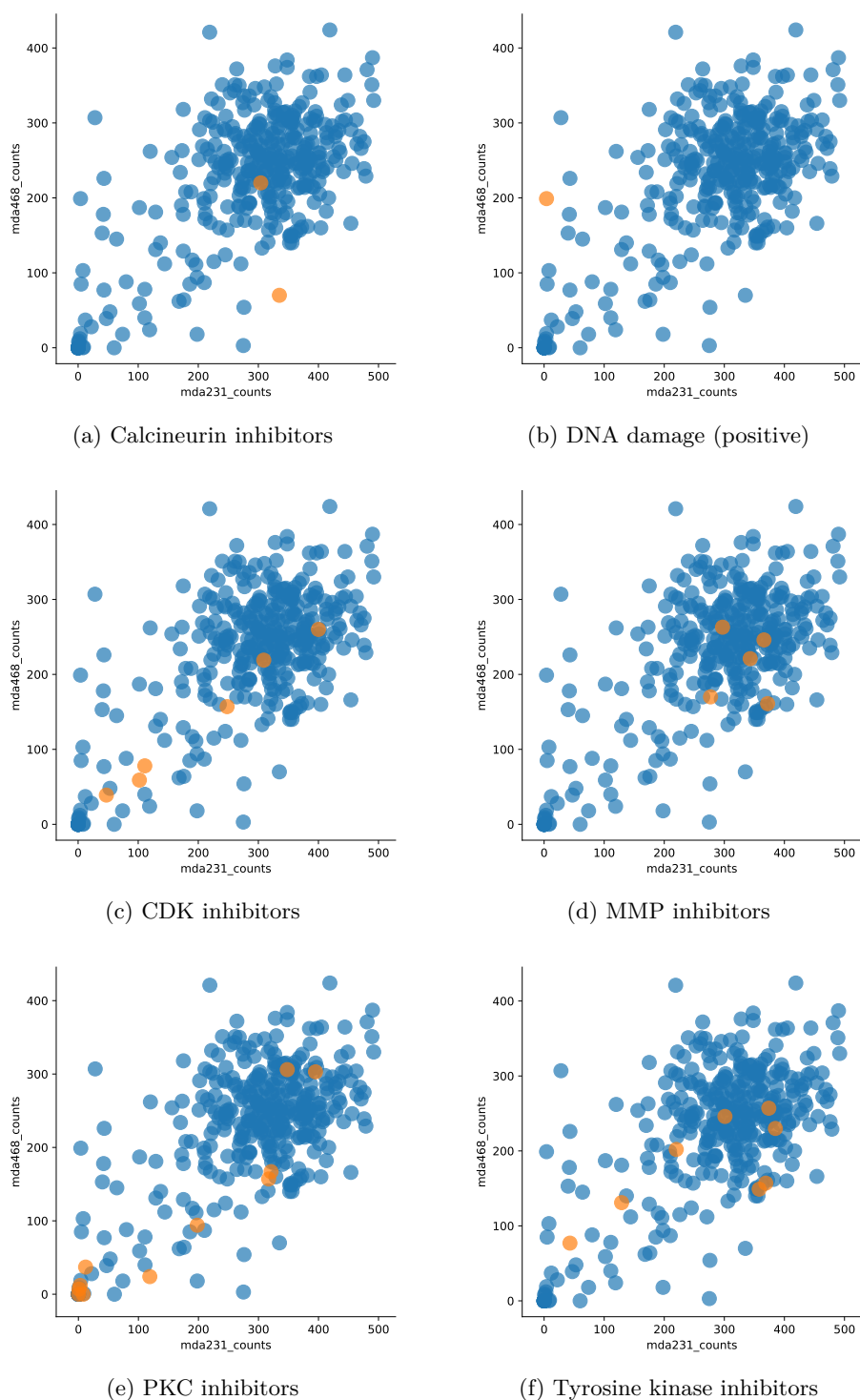


Figure 3.5: Comparison of viability by drug mechanism of action. One may observe differential effects by cell line: (a), (b) show differential effects on viability; (c) shows no clear divergence from the control cluster; (d), (e), (f) show a range of correlated effects.

belong to a dense cluster of negative controls (DMSO and None) with an average of 289 cells for cell line MDA231 and 227 cells for MDA468. We additionally observe a small cluster of highly potent cytotoxic drugs near the origin. There is a positive correlation of $\rho = 0.6556$ for the viability of the two cell lines, suggesting that in many cases, one can actually expect a similar viability phenotype for both cell lines. However, a closer look at Figure 3.4 shows that this is by no means true for all drugs. Some drugs act differentially on both cell lines, demonstrating the heterogeneous effect on distinct molecular TNBC subtypes discussed in section Section 1.2.1. These *differential drug effects* correspond to the “L”-structure traced by the data points skirting the horizontal and vertical axes of the plotting area.

This comparison furthermore enables a categorisation of drugs as having no effect, joint effect, or separate effects on the two cell lines. Figure 3.5 visualises the same population data, this time coloured by selected drug mechanism of action (MOA) categories. We see, in particular, the success of protein kinase C (PKC) inhibitors in regularly killing most—if not all—cells in both cell lines, with a small cluster nearby the origin. Note that PKC inhibitors induce interesting phenotypes under other modes of analysis, as will be seen in both Section 3.4.3 and Section 4.3.2.

3.4.3 Cell cycle modulates double-strand break rate

This section contains an excerpt from a paper published in the proceedings of the International Symposium for Biomedical Imaging in 2018. The paper is reproduced in full in Appendix C.

An example of multivariate analysis in the assay arises in the analysis of double strand breaks (DSBs). Dysfunction of the DNA repair mechanisms is a major hallmark of cancer, also providing therapeutic opportunities. Monitoring DNA damage by the fluorescent labeling of DSBs in cells is therefore an important readout in drug screening of cancer cell lines. DSBs occur when both strands of the DNA double helix are broken. DSBs have various causes, for example, cytotoxic radiation, or DNA replication over an existing single-strand break. Despite the natural DNA repair mechanisms of the cell, DSBs can be irreparable, leading ultimately to apoptosis, or hazardous DNA rearrangements. As such, DSBs are an interesting property to assess when analysing the effects of small compounds upon cancer cells. Three approaches to quantifying DSBs on the Cy3 fluorescence channel are compared, based on detecting and counting spots, granulometric features, and average intensity, which are mutually strongly correlated readouts.

Spot density (spot count normalised by nuclear size) is used as the basis of a DSB comparison between perturbations and negative control. The

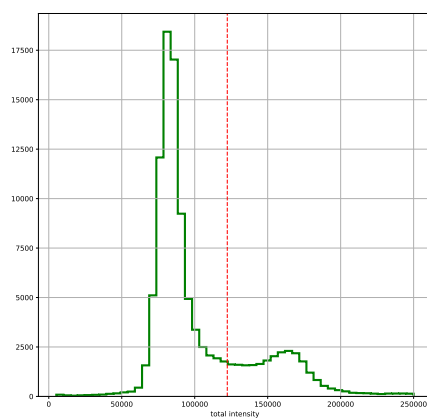
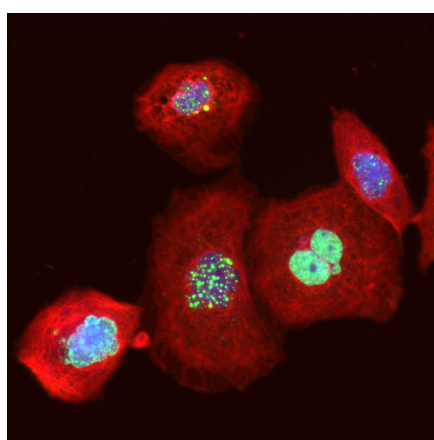
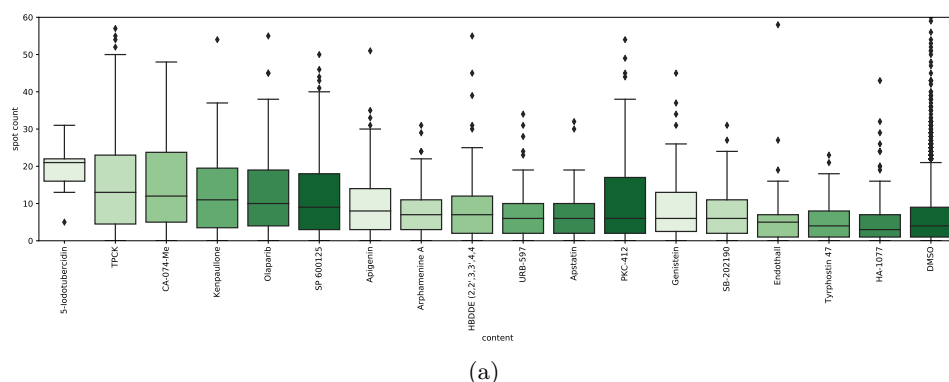


Figure 3.6: Drug perturbations inducing significant changes to DSB distribution (measured by spot density) on cell line MDA231 at $p = 0.01$ with multiple testing correction, ordered by median spot density. Sample of cells perturbed with significant drug PKC-412 (b), DSBs visible as green spots above nucleus. A bimodal distribution is seen on the DAPI channel revealing a growth of mean intensity post DNA replication (c). An Otsu threshold (red vertical line) can be used to stratify the cell population, further revealing distributional differences in DSB rates.

distribution of spot density from a perturbation is compared with that of the negative control in a Kolmogorov-Smirnov (KS) test. Hits are declared at a p -value of 0.01, with multiple testing controlled for with a Benjamini-Hochberg correction. Significant results are displayed in Figure 3.6a, ranked by median spot density and a sample of cells from a significant drug PKC-412 are shown in 3.6b.

Although some drugs have a clear effect on DSBs, they might also cause intermediate effects, which themselves modulate DSB rates. We see in Fig-

ure 3.6c a bimodal distribution in fluorescent density on the DAPI channel, apparently reflecting subpopulations of cells in the G₁ and G₂ phases of the cell cycle. The higher frequency of low intensities corresponds to the longer G₁ phase meaning more cells will be fixed in this phase after hibernation. Because S phase (DNA replication) occurs between the G₁ and G₂ phases, there is more DNA available on which DSBs can occur. Controlling for these phases produces different distributions of DSBs in the G₁ and G₂ subpopulations. This multivariate analysis thus reveals an *effect modifier* of cell cycle phase on DSBs. Stratification is used to establish whether increased rates of DSBs are a direct effect of perturbation or an indirect effect of a modified cell cycle. See Appendix C for further detail.

PKC inhibitors are found to feature prominently among the drugs inducing highest spot densities per cell. However, we know from Section 3.4.2 that drugs in this MOA category often cause low viability. Populations with low sample size tend not to meet the significance criteria. It is therefore drugs such as PKC-412 (Figure 3.6b), which kill fewer cells, that are ultimately declared as hits. This therefore presents a potential selection bias: DSBs are a cytotoxic effect that, when induced at a high rate by perturbation, should increase the rate of cell death. Given the experimental protocol, dead cells are not fixed and are lost during the fluorescence washing step. Hence, the more severe the drug effect on DSBs, the fewer the cells available for measurement. It is concluded in Appendix D that this systematic bias cannot be resolved from the available data, and can only be avoided in future assays with an adjusted imaging protocol, and it also points to potential problems in the interpretation of the derived hit list.

In this section, it has been shown that even in the case that one is interested only in a single feature (here the number of DSBs or a surrogate feature), it is still beneficial to use a multivariate readout in order to extract a better understanding and a cleaner analysis. In addition, the interactions of variables have been identified, revealing the necessity to correct for confounding variables. Given that the analysis of DSBs is so widely used in the field of cancer drug screening, it is important to alert the scientific community to potential misinterpretations of their screening results.

3.4.4 TNBC cell lines assume distinct wild type morphologies

At a glance, one can see that TNBC cell lines manifest distinct morphologies, even in the absence of perturbation. To better understand the degree of this difference, we turn to the wild type screen on 12 cell lines (11 TNBC + negative control). The segmentation and feature extraction pipeline detailed in Section 3.3 is used. With each cell described by a vector of features, one

may proceed to aggregate the readouts so as to describe the cell population in a similar way. Such a phenotypic profile can, for example, summarily characterise the response of the cell population to a given drug perturbation. Profiles, comprising of vectors of aggregated features, can then serve as the basis of comparison between drug effects on a cell line. While this methodology is explored in detail in Chapter 4, let us here consider the special case where individual cell phenotypes are identifiable by an expert annotator. The strategy is thus to classify each cell into one of several meaningful biological classes and to describe the population as a profile of proportions of cells in each of the phenotypic categories. This strategy is based on [Neumann et al. \[2010\]](#).

	Phenotype	Description
	Interphase	Longest phase, cell nucleus is typically small and convex
	Large interphase	Large interphase nucleus, possible replication defect
	Bright interphase	DAPI at higher intensity, due to cell cycle deregulation
	Prometaphase	Mitotic phase, prior to division (short, therefore infrequent)
	Metaphase	Chromosome alignment
	Polylobed	Abnormal shape—mitosis aberration
	Apoptosis	Cell death

Table 3.5: Morphological classes manually annotated on wild type screen data to create a ground truth for training cell classifier.

In order to analyse the morphological landscape of these cell lines, seven nuclear morphology classes have been defined. Table 3.5 lists these classes for the wild type screen: *interphase*, the longest part of the cell cycle, where the cell nucleus is typically small and convex; *large interphase*, corresponding to an abnormally large interphase nucleus, potentially the result of a defect in DNA replication or nuclear membrane control; *bright interphase*, where DAPI exhibits higher fluorescence, presumably due to cell cycle deregulation; *prometaphase*, the first observable mitotic state (condensed chromosomes, broken nuclear envelope), which is relatively rare due to its short

duration; *metaphase*, the mitotic phase in which the cell's chromosomes are aligned in the metaphase plate prior to chromosome segregation; *apoptosis*, cell death; and *polylobed*, in which the cell nucleus takes on an abnormal shape, due to problems in the division process.

Phenotypic profiles are not only informative about drug perturbations, but also about cell lines themselves. Indeed, a cancer cell line is supposed to have acquired properties that differentiate them from normal cells. Depending on the markers used, such differences can be measured by imaging approaches. One may hypothesise it to be interesting to identify the phenotypic profiles of the 12 cancer cell lines from the wild type screen (see Section 3.2.1), corresponding to different molecular subtypes of TNBC, in order to understand to what extent the molecular subtypes coincide with phenotypic differences. This would allow us to infer the biological processes which are perturbed in these different cell lines.

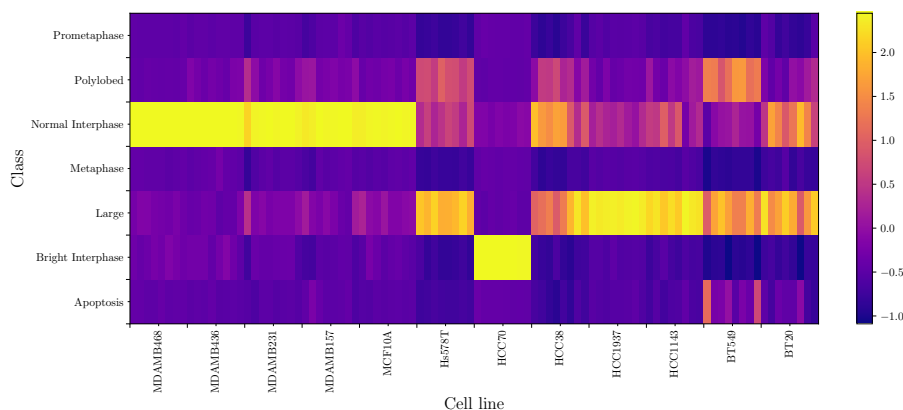


Figure 3.7: Morphological profiles of 12 TNBC cell lines with 8 replicates apiece, based on 7 biologically meaningful classes, derived from manual annotation. One observes the phenotypic similarity between cell line families.

926 cells were annotated in the wild type screen dataset⁴, according to seven morphological classes specified in Table 3.5, from which were extracted the 239 DAPI-channel nuclei features. A SVM with RBF kernel was then trained, with hyperparameters optimised by grid search and 10-fold cross-validation). The trained SVM was used to classify the remaining cells in the assay (some 36338 cells in total). The number of cells in each phenotypic class are counted, and divided by the total to create phenotypic profiles for each well, that is, a summary of seven proportions for each of the 96 wells. The profiles were normalised by subtracting the average from the negative control cell line, MCF10A. The 96 profiles (8 per cell line) are visualised

⁴Later, a larger set of 6530 cells were annotated and released for pedagogical purposes at <https://github.com/jcboyd/deep-learning-workshop/tree/cell-data>

in Figure 3.7. We can see how families of TNBC cell lines cluster in their phenotypes. Of note are the cell lines of the drug screen, MDA231 and MDA468, which manifest predominantly normal interphase cells.

Figure 3.8 plots a (two-dimensional) UMAP dimensionality reduction (McInnes and Healy [2018]) of a balanced sample of 150 cells from 4 of the 12 cell lines. We see a clear separation of cell lines in the UMAP embedding, mirroring their distinctive appearance in the microscopy. Note that a subset of cell lines was chosen to avoid visual clutter, but the similar degrees of separation were observed across the 12. The negative control cell line (MCF10A) is included, as is one of the cell lines studied in the drug screen (MDA231).

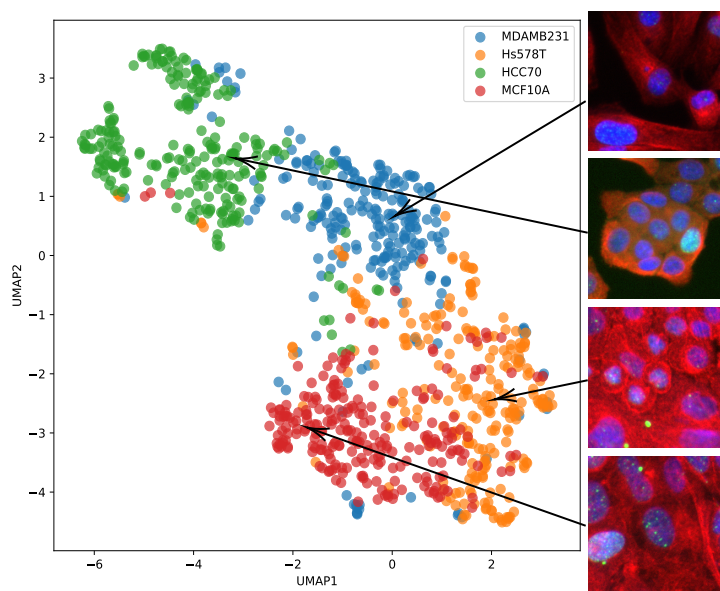


Figure 3.8: UMAP projection from feature space of sample cells from four TNBC cell lines (selected among 12) in a wild type screen. On the right we show a fixed-size (128×128 px) crop centered on an indicative cell from each cell line.

A random forest classifier with 500 trees is also trained to classify cell lines in the feature space. The data set is built from a balanced sampling of 150 cells per cell line, totaling 1800 cells, with 25% of the data reserved for testing. The random forest achieves in excess of 70% accuracy on the 450 test samples. A confusion matrix is provided in Appendix A. Repeated experiments reveal that the greatest confusion occurs between two cell lines of the same family: MDA231 and MDA436. Thus, even with a small fraction of the total data available for training (and little to no tuning), an off-the-shelf classifier achieves high accuracy in discerning cells of different cell lines. This demonstrates that the cells of distinct cell lines occupy different

regions of the feature space and that their basic wild type morphologies are different.

3.5 Discussion

This chapter has presented the HCA analysis pipeline and applied it to a set of use cases, providing first results on TNBC cell lines. In particular in Section 3.4.4, two levels of phenotypic distinction between TNBC cell line wild types have been demonstrated. The first concerns the regulation of cell states, as in Figure 3.7, where we saw how cell populations over different cell lines could be differentiated in a morphological profile, which appeared to cluster by pathology. Through annotation (first manual, then by classifier), these morphological classes abstracted away from the second distinction, that of the morphology of individual cells within each cell line, as in Figure 3.8. It is overcoming this second distinction that will motivate the methods of Chapter 4, albeit in an unsupervised setting where manual annotation is no longer feasible.

Chapter 4

Domain-invariant features for mechanism of action prediction in a multi-cell-line drug screen

This chapter consists of an extended version of a paper published in the journal Bioinformatics in 2019. Modifications have been made where appropriate to cohere with this dissertation.

Summary: We saw previously that high content screening is an important tool for the discovery and characterisation of drugs. Often, high content drug screens are performed on a single cell line. Yet, a single cell line cannot be thought of as a perfect disease model. Many diseases feature an important molecular heterogeneity. Consequently, a drug may be effective against one molecular subtype of a disease, but less so against another. To characterise drugs with respect to their effect not only on one cell line but on a panel of cell lines is therefore a promising strategy to streamline the drug discovery process. The contribution of this chapter is twofold. First, we investigate whether we can predict drug mechanism of action (MOA) at the molecular level without optimisation of the MOA classes to the screen specificities. To this end, we benchmark a set of algorithms within a conventional pipeline, and evaluate their MOA prediction performance according to a statistically rigorous framework. Second, we extend this conventional pipeline to the simultaneous analysis of multiple cell lines, each manifesting potentially different morphological baselines. For this, we propose multitask autoencoders, including a domain-adaptive model used to construct domain-invariant feature representations across cell lines. We apply these methods to a pilot screen of two triple negative breast cancer cell lines as models for two dif-

ferent molecular subtypes of the disease.

Résumé: *Nous avons vu précédemment que le criblage à haut contenu est un outil important pour la découverte et la caractérisation des médicaments. Souvent, les criblages de médicaments à haute contenu sont effectués sur une seule lignée cellulaire. Pourtant, une seule lignée cellulaire ne peut être considérée comme un modèle de maladie parfait. De nombreuses maladies se caractérisent par une importante hétérogénéité moléculaire. Par conséquent, un médicament peut être efficace contre un sous-type moléculaire d’une maladie, mais moins contre un autre. La caractérisation des médicaments en fonction de leur effet non seulement sur une lignée cellulaire mais aussi sur un ensemble de lignées cellulaires est donc une stratégie prometteuse pour rationaliser le processus de découverte de médicaments. La contribution de ce chapitre est double. Premièrement, nous cherchons à savoir si nous pouvons prédire le mécanisme d’action des médicaments (MOA) au niveau moléculaire sans optimiser les classes de MOA en fonction des spécificités du crible. À cette fin, nous comparons un ensemble d’algorithmes dans un pipeline conventionnel et évaluons leurs performances de prédiction du MOA selon un cadre statistiquement rigoureux. Ensuite, nous étendons ce pipeline conventionnel à l’analyse simultanée de plusieurs lignées cellulaires, chacune manifestant des lignes de base morphologiques potentiellement différentes. Pour cela, nous proposons des auto-encodeurs multitâches, y compris un modèle adaptatif au domaine utilisé pour construire des représentations de caractéristiques invariantes au domaine sur les lignées cellulaires. Nous appliquons ces méthodes à un écran pilote de deux lignées cellulaires triple négatif de cancer du sein comme modèles pour deux sous-types moléculaires différents de la maladie.*

4.1 Overview

As discussed in Chapters 1 and 3 high content screening (HCS) is a powerful tool for identifying potential drugs effective against a particular disease. The idea is to expose a cell line representative of the disease to a large panel of drugs. For each drug, one obtains a set of images informative of its phenotypic effect and hence on the biological pathways undergoing perturbation. Various advances in microscopy automation and image analysis have pushed HCS to the early *hit-to-lead* stages of the drug discovery process (Haney et al. [2006]).

The discovery of new drugs may be guided by a reference set of drugs of known mechanism of action (MOA). The MOA of a drug is the particular cellular pathway it perturbs to achieve its effect. Through application of image analysis, one may attempt to infer the MOA of an unknown drug from

HCS image data. Note that MOA can be defined at different levels and with different degrees of specificity: MOA might concern the exact protein that is targeted (e.g. AURKA inhibition), or a specific effect on cellular components (e.g. stabilisation of microtubuli) or perturbation of a more general cellular pathway (e.g. DNA repair). HCS is usually optimised with respect to particular pathways by the choice of the fluorescent markers and readouts (Pepperkok and Ellenberg [2006b]). Consequently, MOA prediction might be reasonably straightforward if MOA classes are chosen in accordance with the phenotypic readout (Ljosa et al. [2013]), but it is challenging in general, in particular if we aim at predicting specific MOAs the assay has not been optimised for.

A second difficulty concerns the cellular model that is used. As a proxy for diseased cells, a cell line cannot be thought of as a perfect model. Many diseases feature a significant molecular heterogeneity. Consequently, a drug may be effective against one molecular subtype of a disease, but less so against another. Furthermore, immortalised cell lines may diverge over time due to genetic drift. For example, HeLa, the quintessential cell line, is famously the cause of great scientific confusion due to difficulties in cell line identification (Horbach and Halfman [2017]) and significant molecular and phenotypic variability (Liu et al. [2019]). To characterise drugs with respect to their effect not only on one cell line but on a consensus of several is therefore a promising strategy to streamline the drug discovery process. Nevertheless, this is not an easy task in morphological screening, as different cell lines usually have distinct archetypal morphologies even prior to perturbation, as shown in Section 3.4.4. It is therefore conceptually difficult to characterise and compare drug effects across cell lines.

In this chapter, we investigate whether we can predict MOA at the molecular level without optimisation of the MOA classes to the screen specificities. To this end, we benchmark a set of algorithms within a conventional pipeline, and evaluate their MOA prediction performance according to a statistically rigorous framework.

Second, we extend this conventional pipeline to the simultaneous analysis of multiple cell lines, each with potentially different morphological baselines. For this, we propose multitask autoencoders, including an adaptive model used to construct domain-invariant feature representations across cell lines. We apply these methods to a pilot screen of two triple negative breast cancer (TNBC) cell lines as models for two different molecular subtypes of the disease.

In Section 4.2 we formalise a range of profiling approaches from the literature according to four key properties, and extend this to a multi-cell-line analysis. In Section 4.3 we illustrate the benefit of multi-task models for our dataset through extensive cross-validation and provide an exploratory analysis of

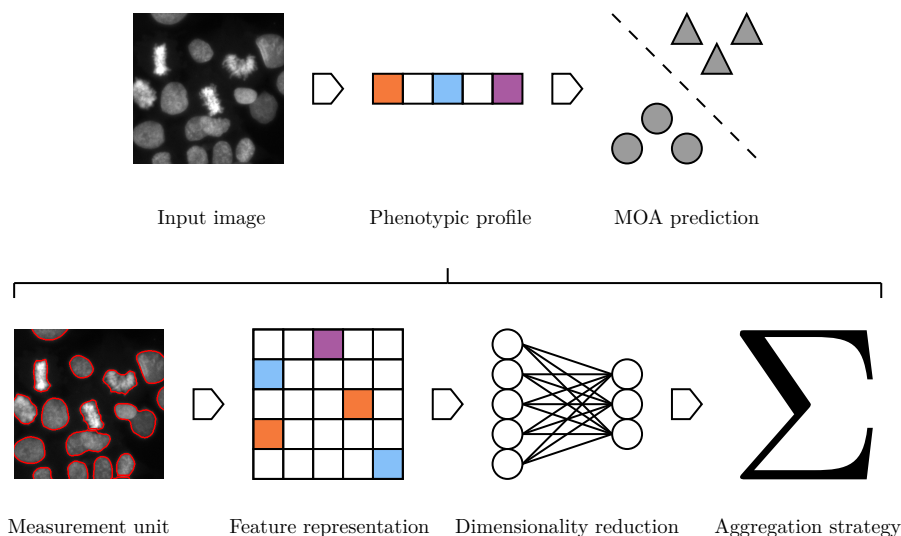


Figure 4.1: MOA prediction is performed on an image via a phenotypic profile. The development of such a profile spans four ordered stages. Each stage may be accomplished by a variety of algorithms, the combination of which define a unique pipeline. Some stages may be omitted in certain pipelines, or subsumed to a common framework.

differential drug effects between the two cell lines. In Section 4.4 we discuss our methods and the obtained results.

4.2 Phenotypic profiling for mechanism of action prediction

This section describes the approaches for phenotypic profiling we have benchmarked. We embed these descriptions in a formalised overview of phenotypic profiling strategies to motivate the different setups. In Section 4.2.3 we describe methods for a joint analysis of multiple cell lines.

4.2.1 MOA prediction

Drugs are assigned a class based on their *mechanism of action* (MOA), the cellular pathway perturbed by the drug, as depicted in Figure 4.1. Given a set of drug profiles annotated with MOA classes, we can simulate reference and discovery drug sets in a *leave-one-compound-out* cross-validation (LOCOCV) scheme. At each fold of the cross-validation, we hold out a drug and predict its MOA class using a classifier trained on the remaining

4.2. PHENOTYPIC PROFILING FOR MECHANISM OF ACTION PREDICTION 81

“reference” drugs. The prediction is made as the nearest neighbour (1-NN) in cosine distance between drug profiles, $d(\mathbf{p}, \mathbf{p}') = 1 - \cos \theta_{\mathbf{p}, \mathbf{p}'}$. This was proposed in [Ljosa et al. \[2013\]](#) as an equitable way of comparing profiling algorithms. We settle for this lightweight approach as our focus here is on the discriminative power of the profiles.

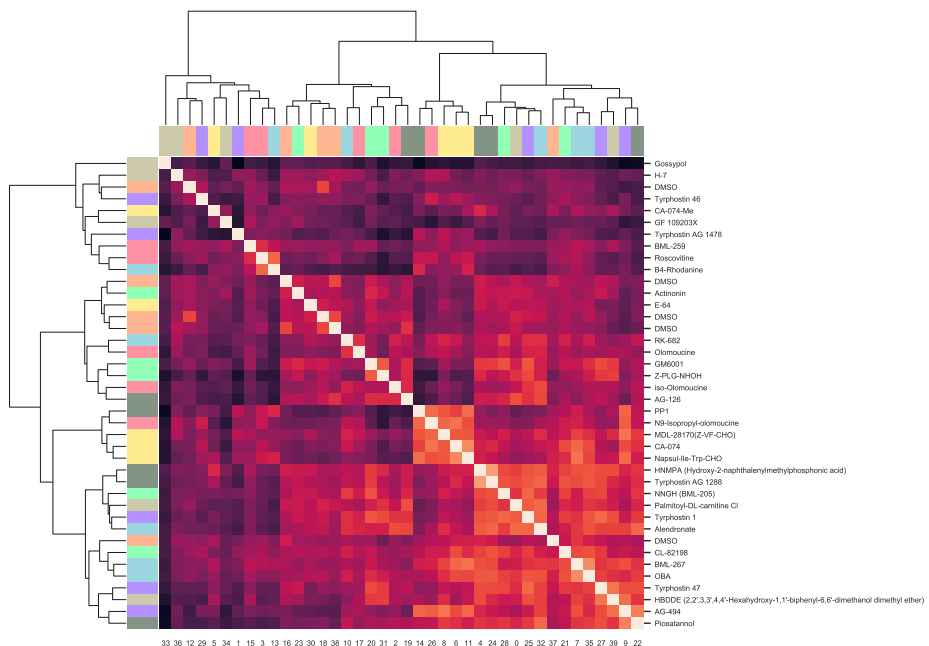


Figure 4.2: Example of how phenotypic profiles may cluster with a hierarchical model and Ward linkage for 40 drugs in 8 mechanism of action classes (including negative control)s from our drug screen data set. Heat map colour indicates distance between profiles, and dendrogram leaf colours indicate mechanism of action class.

In our drug screen, the drugs comprise of a set of panels of kinase, protease and phosphatase inhibitors and can be categorised into 70 mechanism of action (MOA) classes of varying sizes, according to their targets. For our experiments, we take the 8 MOA classes having at least five member drugs. These are CDK inhibitors, cysteine protease inhibitors, EGF receptor kinase inhibitors, MMP inhibitors, DMSO (negative control), PKC inhibitors, protein tyrosine phosphatase inhibitors, and tyrosine kinase inhibitors. In comparison with other datasets, [Adams et al. \[2006\]](#) used 51 drugs in 13 MOA categories, [Slack et al. \[2008\]](#) used 35 drugs in six MOA categories, and the widely studied Broad Institute Benchmark Collection 21 (BBBC21v2, [Ljosa et al. \[2012\]](#)) – used, for example, in [Kandaswamy et al. \[2016\]](#) and [Godinez et al. \[2017\]](#) – consists of 39 drugs in 13 categories. The

key difference is that our own MOA classes were not selected *a posteriori* to reflect visually different phenotypes, mounting a greater bioinformatic challenge than the standard benchmark datasets, where even a simple model can be extremely effective. For example, Singh et al. [2014] achieved 90% accuracy with element-wise averaging of hand-crafted features after a simple luminosity correction.

4.2.2 Phenotypic profiling

As stated in Section 1.1.3, the conventional approach to HCS analysis is a multi-stage pipeline, consisting of a sequence of modules of image and statistical analysis, including cell segmentation and hand-crafted feature extraction (Caicedo et al. [2017]). The aim is to ascribe a phenotypic profile to each cell population to serve as the basis of comparison between drugs. Each profile will take the form of a vector $\mathbf{p} \in \mathbb{R}^D$ of some dimensionality D and is constructed according to four ordered methodological stages: measurement unit, feature representation, dimensionality reduction, and aggregation strategy (Figure 4.1). Certain properties may be omitted by some approaches, or subsumed to a common framework, such as a neural network, that may perform each task simultaneously (Kraus et al. [2016], Godinez et al. [2017]). In the following sections we detail each property in turn, providing references to the relevant literature and describing the concrete setup that was retained for the benchmarking.

Measurement unit

The most common measurement unit is the cell itself, constituting a *per-cell* analysis. This entails an initial segmentation of the cells (their nuclei and other organelles). We segmented cell nuclei on the DAPI channel by subtracting a background image formed with a mean filter, before clipping to zero. Touching nuclei were further separated by applying the watershed transform on the inverse distance map of the foreground image. The cytoplasm was segmented from the microtubule channel (Cy5) following Jones et al. [2005].

Alternatively, one might analyse the image field directly in a *per-image* analysis, such as in Orlov et al. [2008], Uhlmann et al. [2016], or Godinez et al. [2017]. Such approaches are referred to as *segmentation-free*, as they obviate the segmentation phase of the conventional pipeline. In this chapter, we choose to focus on the cell as unit of measurement.

Feature representation

For a given choice of measurement unit, one further chooses a feature representation. This yields a matrix $\mathbf{X} \in \mathbb{R}^{N \times D}$ for each well where N is the number of samples for that well and D is the number of features measured. For each segmented cell we extracted a previously published set of features (Walter et al. [2010b]) across the three fluorescent channels, as well as spot features informative on DNA double-strand breaks (Boyd et al. [2018]). These features, hereafter referred to as *handcrafted features*, thus retain a degree of biological interpretability. In contrast, in Orlov et al. [2008] and Uhlmann et al. [2016] a large number of handcrafted features are extracted over each image as a whole.

More recently, features are extracted within the layers of a convolutional neural network (CNN) trained directly on image pixels. We benchmarked a convolutional autoencoder (CAE) following the design of Sommer et al. [2017], trained on $40 \times 40 \times 3$ inputs, formed by extracting 100×100 px padded bounding boxes of segmented cells, rescaling, and stacking the fluorescent channels. The central hidden layer of the trained CAE is then used as a feature representation.

Dimensionality reduction

Dimensionality reduction requires some function $enc : \mathbf{X} \rightarrow \mathbf{Z}$ where $\mathbf{Z} \in \mathbb{R}^{N \times M}$, with reduced dimensionality $M < D$. The objective is to capture the essential information in lower dimension or to cast the high-dimensional feature vector to an interpretable representation. Supervised classification of individual cells is one way of achieving this (Neumann et al. [2010]), as each cell is represented either by a one-hot binary vector $\mathbf{z}_i \in \{0, 1\}^M$ or by a vector of probabilities $\mathbf{z}_i \in [0, 1]^M$ where $\sum_j z_{ij} = 1$ and M is the number of classes, in effect, the new dimensionality. With multiple-instance learning (MIL) (Kraus et al. [2016]) one can circumvent the manual effort involved in creating a phenotypic ontology and a manually curated training set. Here, one labels each cell with the MOA of the drug of the population, thus creating a weakly supervised ground truth. As individual cells may respond differentially to perturbation, not all regions of an image will bear the hallmarks of a particular drug, but the cellular landscape can be viewed as a multiple instance bag of objects. Godinez et al. [2017] make this assumption implicitly. We benchmarked a random forest tuned to 500 trees, trained on cells weakly labeled by MOA class of their well ($M = 8$). Necessarily, we partition wells into separate train and test sets, where the test data alone is used to build profiles for the MOA prediction downstream.

Another popular option is to use unsupervised learning. We benchmarked

hard clustering methods k-means and hierarchical clustering in Euclidean space with Ward linkage. These were tuned to $M = 80$ and $M = 100$ clusters respectively (by cross-validation, on the training set). K-means is fast to fit approximately, in particular using mini-batch training. On the other hand, even using optimised software (Müllner et al. [2013]), hierarchical clustering is not scalable. We also performed soft clustering with Gaussian mixture models (GMM) (Slack et al. [2008]), tuned to $M = 100$ Gaussians.

Feature selection and principal components analysis (PCA) are other popular options (Loo et al. [2007]). Here, we applied PCA on the handcrafted features, selecting 40 of the 516 components, retaining $\sim 90\%$ of the energy on average. We further whitened the latent features.

Autoencoders, as used by Kandaswamy et al. [2016], formulate a function $f(\mathbf{x}) = \text{dec}(\text{enc}(\mathbf{x}))$, where $\text{enc}(\cdot)$ and $\text{dec}(\cdot)$ correspond to the encoder and decoder parts of the neural network. This model can be trained with a mean square error (MSE) loss function,

$$\mathcal{L}(\mathbf{X}; \boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^N \|\mathbf{x}_i - f(\mathbf{x}_i)\|_2^2 + \lambda \|\boldsymbol{\theta}\|_F^2, \quad (4.1)$$

for the N samples in the dataset and where λ is a tunable hyperparameter for the regulariser. The hidden representation corresponds to the output of the encoder, the central layer of the neural network, i.e. our reduced sample is $\mathbf{z}_i = \text{enc}(\mathbf{x}_i)$. We train shallow affine autoencoders—with a single hidden layer (tuned to $M = 100$ neurons)—on handcrafted features. We also train deep convolutional autoencoders directly on image pixels, as described in Section 4.2.2. Note that such models perform both feature extraction and dimensionality reduction simultaneously. Here, the encoder consists of 5×5 and 3×3 convolutional layers, with 16 and 8 kernels respectively, and a fully connected layer ($M = 128$), each alternating with max pooling layers. The decoder mirrors this, albeit replacing pooling with upsampling.

Aggregation strategy

Once all cells are endowed with a representation, one needs some means of reducing the population to a single profile, $\mathbf{p}$. A variable number of cells per well requires an aggregation strategy yielding a profile of fixed size. The most straightforward approach is an element-wise averaging as in Adams et al. [2006] where $\mathbf{p} = \frac{1}{N} \sum_{i=1}^N \mathbf{z}_i$. This amounts to replacing the cell population cluster with its own centroid, and for classification or clustering approaches (Section 4.2.2), this simply corresponds to the percentage of cells that fall into each category.

Alternatively, [Perlman et al. \[2004\]](#) apply an element-wise Kolmogorov-Smirnov test and [Loo et al. \[2007\]](#) use the vector normal to the SVM decision boundary between perturbed and control populations. As we are modeling the negative control as one of our ground truth classes, we aggregate exclusively with element-wise averaging in our analysis.

4.2.3 Multi-cell-line analysis

One can extend the above MOA prediction framework for multiple cell lines either by pooling data or by ensembling models. In a pooling analysis such as [Warchal et al. \[2016\]](#), the cells of the respective cell lines are first normalised and then grouped across drugs to increase the amount of available data. An ensemble approach such as in [Rose et al. \[2018\]](#) creates models for each cell line and aggregates their individual predictions. This approach has the additional advantage of allowing different imaging modalities of fluorescent markers.

We adopted a pooling approach to predict MOA from multiple cell lines. The challenge of this approach is to reconcile the inherent differences between the cell lines in feature space, which derives from the fundamental morphological differences of the cell lines. For this purpose, we tested multi-task autoencoders (Figure 4.3), extensions of both our affine and convolutional autoencoders.

Multitask autoencoders for multi-cell-line analysis

Multi-task models learn to predict multiple targets simultaneously and multitask neural nets often build more generalised internal representations ([Caruana \[1997\]](#)). We propose multitask autoencoders as an approach to reconcile the divergent nature of our multi-cell-line data.

One obvious design is to have separate decoders for each cell line with a shared encoder. During training, minibatches can be split after the shared layers with samples routed to the decoder corresponding to their cell line. We thus minimise,

$$\mathcal{L}_{MTA}(\mathbf{X}; \boldsymbol{\theta}) = \sum_{i:d_i=0} \|\mathbf{x}_i - dec_s(enc(\mathbf{x}_i))\|_2^2 + \sum_{i:d_i=1} \|\mathbf{x}_i - dec_t(enc(\mathbf{x}_i))\|_2^2, \quad (4.2)$$

where d_i identifies the cell line of $\mathbf{x}_i$. We test multitask variants of both our affine and convolutional autoencoders described in Section 4.2.2.

The fundamental morphological differences between the cell lines can be quantified in feature space by a $\mathcal{H}$ -divergence, first proposed by Ben-David et al. [2010], where $\mathcal{H}$ is some hypothesis class (such as the space of linear classifiers). This is expressed as $d_{\mathcal{H}}(D_S^X, D_T^X) = 2 \sup_{h \in \mathcal{H}} |P_{\mathbf{x} \sim D_S^X}(h(\mathbf{x}) = 1) - P_{\mathbf{x} \sim D_T^X}(h(\mathbf{x}) = 1)|$, where the domains D_S^X and D_T^X are marginal probability distributions on $\mathbf{x}$. That is, given source and target domains, and given a hypothesis class $\mathcal{H}$, the divergence between the source and target domains is the best performance among that class of classifiers trained to distinguish them. In practice, we can approximate this by training a classifier of the class $\mathcal{H}$ on the constructed dataset, $U = \{(\mathbf{x}, 0) : \mathbf{x} \in S\} \cup \{(\mathbf{x}, 1) : \mathbf{x} \in T\}$, that is, a classifier trained to distinguish between the domains. Ajakan et al. [2014] proposed multi-task classifiers involving a domain discriminator trained against a classifier adversarially. As the classifier was trained to minimise one loss, the competing domain discriminator was trained to maximise another loss, such that data from either domain could not be distinguished, promoting *domain-invariant features* in the earlier, shared layers of the network.

Thus, we propose domain-adversarial autoencoders (DAA), to promote domain-invariant representations between the cell lines. This consists of attaching a domain discriminator $g(\mathbf{x})$ to the encoding layer. This can be thought of as a dynamic regularisation function. In a bias-variance tradeoff, we expect this to on average *increase* the reconstruction error of the autoencoder. However, we hypothesise that the domain-invariant features learned will be more useful to the downstream MOA prediction when combining heterogeneous cell line data. For example, with a single additional affine layer, $g(\mathbf{x}) = \mathcal{S}(\mathbf{W}_d \text{enc}(\mathbf{x}) + \mathbf{b}_d)$, where $\mathbf{W}_d$ and $\mathbf{b}_d$ are the weights and biases of the layer, and $\mathcal{S}$ is the softmax function producing posterior probabilities $p(d = 0|\mathbf{x})$ and $p(d = 1|\mathbf{x})$. The loss function then becomes,

$$\begin{aligned} \mathcal{L}_{DAA}(\mathbf{X}, \mathbf{d}; \boldsymbol{\theta}) = & \frac{1}{N} \sum_{i=1}^N \|\mathbf{x}_i - f(\mathbf{x}_i)\|_2^2 - \\ & \frac{\omega}{N} \sum_{i=1}^N d_i g(\mathbf{x}_i) - \log[1 + \exp(g(\mathbf{x}_i))], \end{aligned} \quad (4.3)$$

that is, the difference of a mean square error (MSE) loss and a log loss, where $f(\mathbf{x})$ is defined as before, and ω is a modulating hyperparameter. However, now the parameters of $g(\mathbf{x})$ are updated to *maximise* $\mathcal{L}_{DAA}$, so as to improve domain discrimination. At the same time, the parameters of $f(\mathbf{x})$ are updated to *minimise* $\mathcal{L}_{DAA}$. This has the dual effect of minimising the MSE (as usual) but also maximising the log loss. This is known as an

adversarial step, and aims at converging to a saddle point between the two objectives. In practice, this is implemented with a *gradient reversal* pseudo-layer (Ganin et al. [2016]), which is readily programmable in standard deep learning frameworks.

We test multitask versions of both our affine and convolutional autoencoders, and compare them directly in Section 4.3.2. The domain discriminator of our DAAs are linear in terms of the encoding (domain invariant features) and the weight of the log loss was tuned to $\omega = 1.5$. For each affine model we tried the same range of hidden units in a grid search $M \in \{100, 125, 150, 175, 200\}$, and trained for 20 epochs using the RMSprop gradient descent algorithm (Tieleman and Hinton [2012]). For the convolutional autoencoders we kept the architecture defined in Section 4.2.2. We further used weight decay ($\lambda = 10^{-3}$) for all models as well as batch normalisation (Ioffe and Szegedy [2015]), which we found stabilised the training, in particular the adversarial training.

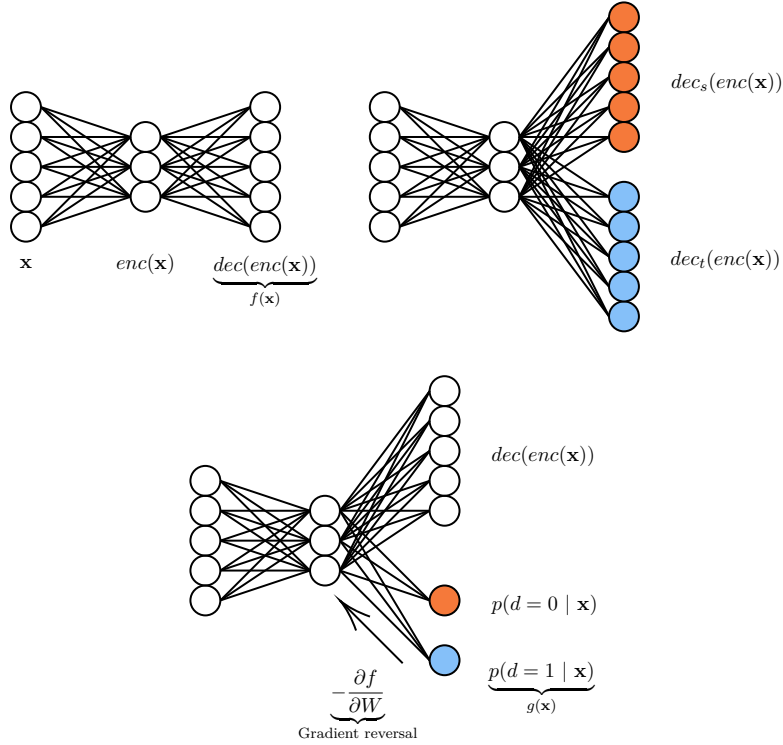


Figure 4.3: Multitask autoencoders used for dimensionality reduction over multi-cell-line data. Clockwise from top left: vanilla autoencoder, multitask autoencoder, and domain-adversarial autoencoder. Colouring indicates separate treatment of each domain (cell line).

4.2.4 Model evaluation

We compared different profiling settings by evaluating performance on a MOA prediction task. For this, we balanced our datasets by randomly sampling five drugs from each of the 8 classes specified in Section 4.2.1, analysing 40 drugs at a time. Applying the LOCOCV scheme described in Section 4.2.1, we note that random accuracy is 12.5%. To account for random variability, we repeated LOCOCV 60 times with different sets of randomly sampled drugs and in Section 4.3 report average top-1 accuracy and standard deviation as the percentage of MOAs correctly predicted by the 1-NN classifier. We consider this to be a more rigorous approach in a comparative study, as while a given method often fit one drug set well, it was harder to find hyperparameter choices that worked well across all sets. We used a Wilcoxon signed-rank test to establish significance against baselines over the 60 rounds.

4.2.5 Software

We use Cell Cognition (Held et al. [2010]) to perform the first stages of the classical analysis pipeline, namely, image preprocessing, cell segmentation and feature extraction.

All models were coded using the `scikit-learn` (Pedregosa et al. [2011]) and `Keras` (Chollet et al. [2015]) frameworks for Python, unless otherwise noted¹. Basic image processing was performed with `scikit-image` (van der Walt et al. [2014]).

4.3 Results

In Section 4.3.1 we evaluate a range of approaches to dimensionality reduction—as described in Section 4.2.2—on their utility in creating cell representations that aggregate into discriminative phenotypic profiles for MOA prediction. This we do in separate single-cell-line experiments. In Section 4.3.2 we show how our best performing model on single-cell-line data—the autoencoder—may be extended for multi-cell-line analysis, providing comparisons for learning on handcrafted features, as well as raw pixels. We then illustrate how our optimised phenotypic profile design can be used to identify differential drug effects between cell lines across our entire drug panel in Section 4.3.2. In Section 4.3.2 we explore how the effect of adding cell lines to an analysis effects MOA predictability.

¹Worked examples of code and feature data available at <https://github.com/jcboyd/multi-cell-line>

Approach	MDA231 accuracy (μ, σ)	MDA468 accuracy (μ, σ)
Handcrafted features	18.58, 5.62	20.08, 4.49
PCA + whitening	21.33, 6.54*	19.58, 6.43
Hierarchical clustering	17.83, 6.46	20.13, 6.46
K-means	19.38, 7.56	19.50, 5.86
GMM	20.21, 6.88	21.29, 7.37
Autoencoder	22.13, 6.48**	23.92, 6.23**
Random forest (MIL)	19.51, 9.95	16.81, 8.16
Conv. autoencoder	19.96, 6.23	13.79, 5.51

Table 4.1: Comparison of dimensionality reduction approaches against unreduced baseline for cell lines treated separately. We show mean and standard deviation of accuracies over 60 runs with (*) indicating significant results at the $p = 0.05$ level; (**) at the $p = 0.01$ level.

4.3.1 Single cell line analysis

In Table 4.1 we evaluate a range of approaches to dimensionality reduction on cell lines taken separately. The baseline for this comparison are the hand-crafted features averaged element-wise from segmented cells in each well. Note that even such a simple baseline proved to be highly competitive in earlier comparative studies such as Ljosa et al. [2013]. The models are used to create a reduced representation of cells prior to aggregation by element-wise averaging (Section 4.2.2). The one exception is the convolutional autoencoder, which learns cell representations directly from image pixels.

We observe dimensionality reduction techniques register broad improvement over the baseline, with PCA ($W = 385.0, p < 0.05$) and autoencoders ($W = 281.5, p < 0.01$) significant for the MDA231 cell line. Autoencoders also registered significant improvement ($W = 356.0, p < 0.01$) for the MDA468 cell line. This further motivates autoencoders as the benchmark in our multi-cell-line analysis (Section 4.3.2).

The deep convolutional autoencoder fails to stand out from the group. However, this may rather testify to the effectiveness of handcrafted features on cell line data—at least at this resolution—over learning representations from scratch.

The sole weakly supervised method, multiple instance learning (MIL) with random forests, shows promise on cell line MDA321, but falls short on MDA468. This may stem from the necessary splitting of data into train and test sets prior to LOCOCV, reducing the available training data. Approaches based on weakly supervised MIL are popular, particularly for deep learning approaches, but we do not see any benefit for them on our

Approach	Pooled cell line accuracy (μ, σ)
Autoencoder	31.67, 6.43
Multitask autoencoder	32.04, 6.88
Domain-adversarial autoencoder	35.67, 6.94**

Table 4.2: MOA prediction on multiple cell lines (pooled) with autoencoders trained on handcrafted features. From top to bottom: vanilla autoencoders (baseline), multitask autoencoders and domain-adversarial autoencoders. We compare with the vanilla autoencoder (top row) ((**): $p < 0.01$)

dataset.

4.3.2 Analysis on multiple cell lines

So far, we have considered the analysis of several cell lines as independent problems to inform model selection. We now turn to a joint analysis on multiple cell lines.

Prediction of MOA from multiple cell lines

With their different transcriptional programs multiple cell lines potentially bear complementary information on the mechanism of action of a drug. We pool cells in corresponding wells across our two cell lines, thus enlarging the available data for each drug. In each case, the data from each cell line were standardised separately to have zero mean and unit variance for all features. Our multitask autoencoders are compared with their single-task counterparts, the best performing models from Section 4.3.1.

We observe in both Tables 4.2 and 4.3 that we obtain a higher degree of accuracy in MOA prediction for our multitask autoencoders compared with their baselines, particularly the domain-adversarial autoencoders, which achieve a statistically superior average accuracy ($W = 283.5, p < 0.01$) for the shallow variant, based on handcrafted features, as well as for the deep learning variant ($W = 438.5, p < 0.01$). The former constitutes our best overall accuracy in MOA prediction on this dataset. This supports our hypothesis that promoting domain invariant features facilitates the mixing of heterogeneous data from multiple cell lines. As anticipated in Section 4.2.3, adversarial training did not improve the reconstruction error of our autoencoders, but the resultant features performed better downstream in the MOA prediction pipeline.

Inspired by Ganin et al. [2016], we use t-SNE (Maaten and Hinton [2008]) to project a sample of learned cell features into two dimensions. We typi-

Approach	Pooled cell line accuracy (μ, σ)
Conv. autoencoder	19.58, 6.98
Multitask conv. autoencoder	20.42, 6.14
Domain-adversarial conv. autoencoder	22.38, 5.91^{**}

Table 4.3: MOA prediction on multiple cell lines (pooled) with convolutional autoencoders. From top to bottom: vanilla convolutional autoencoders (baseline), multitask convolutional autoencoders and domain-adversarial convolutional autoencoders. We compare with the vanilla convolutional autoencoder (top row) (^{**}) : $p < 0.01$.

cally observe a higher degree of alignment between the feature distributions of the two domains as produced by the domain-adversarial model, as illustrated in Figure 4.4. To quantitatively confirm this domain overlap, we compute the mean silhouette score over all points where the cluster identity of each point is simply its domain class. The scores given in Figure 4.4 of 0.11 (lower overlap) and 0.01 (higher overlap) for unadapted and adapted features respectively are typical. Altschuler and Wu [2010] wrote that multiple modalities render aggregation over a cell population problematic, as a centroid may be a bad representative of the overall population. Computing domain invariant features appears to be a partial remedy to this when pooling heterogeneous data in a multi-cell-line analysis.

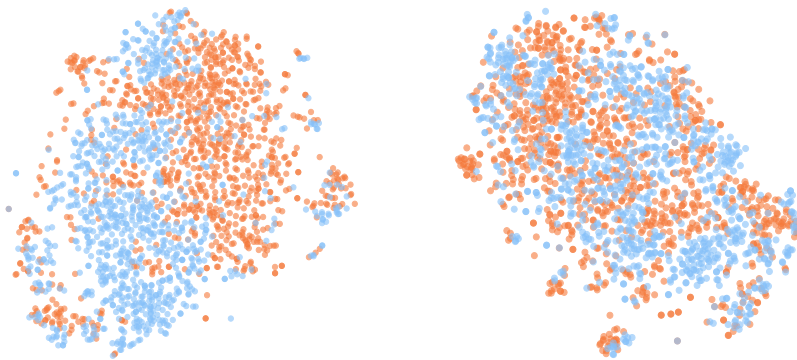


Figure 4.4: t-SNE embeddings of encodings from autoencoder (left) and domain-adversarial autoencoder (right), with cell lines distinguished by colour, and mean silhouette scores of 0.11 and 0.01 respectively.

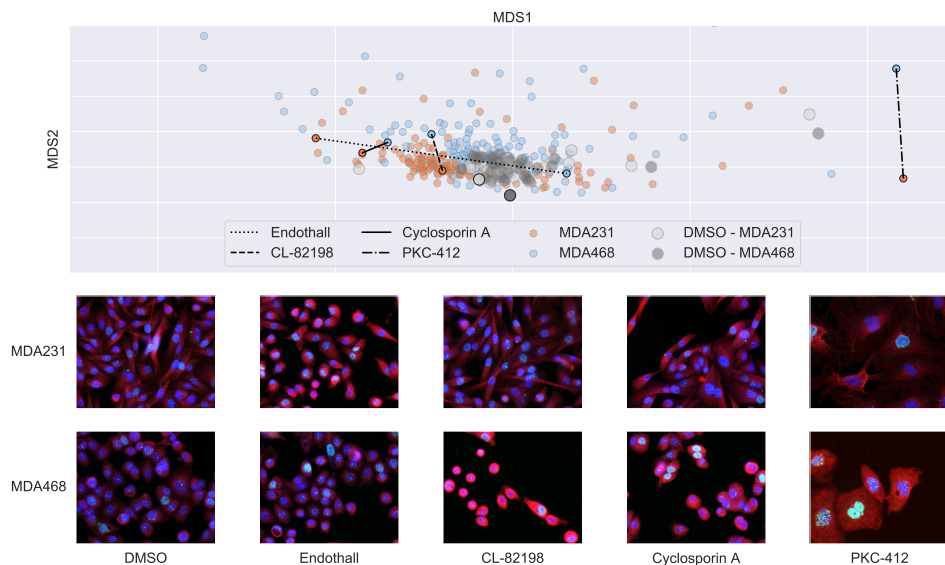


Figure 4.5: MDS embedding of drug effect profiles for MDA231 and MDA468 cell lines with DMSO centroid centered on origin. Detection of differential drug effects between cell lines with examples for each category below (MDA231 top, MDA468 bottom). From left to right: no drug effect in either cell line (negative control); drug effect in MDA231 cell line only; drug effect in MDA468 cell line only; similar drug effects in both cell lines; differentiated drug effects in both cell lines. Shown are example images, blue: DAPI, red: microtubules, green: DSB.

Differential drug effects across cell lines

Our DAA approach provides us with a representation that is optimised with respect to both MOA prediction accuracy and domain invariance between the cell two lines. This can assist us in producing profiles for all drugs in our pilot screen and investigate the differential effects of drugs across cell lines. For this, we trained our network on all data, producing phenotypic profiles for all drugs in the screen. We zero-centered each cell line by subtraction of their respective DMSO centroids and compared distances of drug profiles both from the DMSO centroid and between cell lines. By ranking these distances, we can identify four drug effect cases:

- no drug effect in either cell line;
- drug effect in one cell line only;
- differentiated drug effects in both cell lines;
- similar drug effects in both cell lines.

We visualise the relative distances between drug profiles using multi-dimensional scaling (MDS) on Euclidean distance in Figure 4.5 and identify examples of each of these cases. We include a comparison of DMSO populations that illustrate the unperturbed morphological differences between the two cell lines. We first show an indicative sample of DMSO cells from each cell line. Among the drugs, Endothall has a phenotypic effect on MDA231, but no visible effect on MDA468 (MDA231 cells are rounded up and smaller than in DMSO). Conversely, CL-82198 has an effect on MDA468 cells (cells are smaller and display cytoskeletal changes) and no visual effect on MDA231 cells. Cyclosporin A has a similar effect on both cell lines; the cell lines actually preserve many of their morphological baseline differences, but have a higher fraction of binucleated cells. PKC-412 has a differential effect on both cell lines. While the cell size is increased, the morphological properties as well as the number of DSBs seem to be very different between cell lines.

Effects of accumulating cell lines

Rose et al. [2018] demonstrated an increasing accuracy in MOA prediction as data from cell lines are added to create a growing ensemble of predictive models. This illustrates the value of drawing upon several biological sources to guide a drug discovery process. Nevertheless, predictive models will tend to perform better when supplied with greater volumes of data anyway. Any attribution of a model’s success to a richer biological foundation must first correct for the confounding effect of an increasing sample size.

We ran a separate experiment controlling for the aforementioned bias to attempt to measure the effective power of heterogeneous cell line data. To do this we created equally sized samples: 10000 randomly subsampled cells from the MDA231 cell line; 10000 randomly subsampled cells from the MDA468 cell line; and 5000 cells sampled from each cell line and pooled into a multi-cell-line dataset. We did this for the handcrafted features of segmented cells, averaged element-wise, again repeated over the 60 experimental folds. We found the pooled samples yielded an average accuracy of 20.89, significantly improving over the pure MDA231 sample at 14.94 ($W = 240.0, p < 0.01$) and the pure MDA468 sample at 19.42 ($W = 516.0, p < 0.1$). This therefore supports the hypothesis that a multi-cell-line analysis can be advantageous in and of itself, even before accounting for any increased sample size.

Generalising to further cell lines

To test how our model behaves with increasing numbers of cell lines, we acquired image data without drug perturbation of a third TNBC cell line

(MDA-MB-157) under the same protocol as the pilot screen. In order to apply domain adaptation to K cell lines with $K > 2$, the log loss in equation 4.3 was replaced with a cross entropy loss,

$$\mathcal{L}_{DAA}(\mathbf{X}, \mathbf{d}; \theta) = \frac{1}{N} \sum_{i=1}^N \|\mathbf{x}_i - f(\mathbf{x}_i)\|_2^2 + \frac{\omega}{N} \sum_{i=1}^N \log p_{d_i}, \quad (4.4)$$

where p_{d_i} is the softmax probability produced by the domain discriminator, indexed by the domain of i th sample, and all other terms are defined as before. We found that this simple modification sufficed to train effectively on the new dataset, provided ω was reduced as the number of cell lines increased. We produce a t-SNE plot in Figure 4.6 and observe a similar tendency of distributional overlap for three cell lines. Interestingly, there is only a moderate number of additional parameters when we add a new cell line, which contrasts to the multi-task autoencoders where a whole new decoder is required for each new cell line.

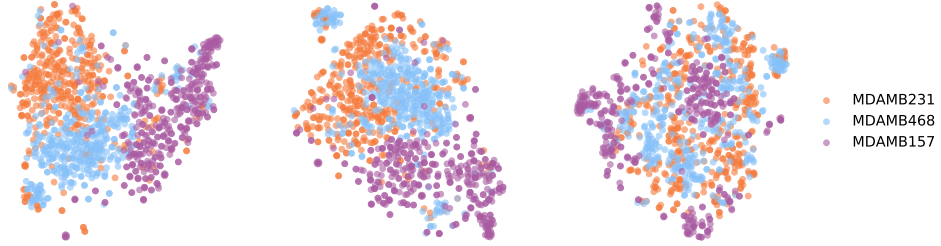


Figure 4.6: t-SNE embeddings of encodings from handcrafted features (left), autoencoder (center) and domain-adversarial autoencoder (right), with cell lines distinguished by colour. Respective silhouette scores of 0.22 and 0.14 and -0.02 confirm the reduced divergence in the adapted domains.

As stated above, this new dataset was not a drug screen and therefore we could not evaluate our model in the same way as before. We were, however, able to pretrain our model on this morphological screen and transfer it to our drug screen to be used as a feature encoder directly. Compared with an equivalent vanilla autoencoder our model performed marginally better for three cell lines (32.08 compared with 31.17) ($W = 366, p < 0.1$), following our evaluation strategy. Though not the exact intended application of our model, we again see an improvement over baselines, suggesting an aptitude of our method for analysing multi-cell-line data. It will be the subject of future work to test our method on a full drug screen in greater numbers of cell lines.

4.4 Discussion

In this chapter we address prediction of mechanism of action (MOA) at a molecular level. Importantly, we have not optimised the MOA classes with respect to the readout of the screen, as is common in many benchmarking studies. We have studied a number of different approaches, including traditional approaches based on hand-crafted features, and deep learning approaches, allowing us to learn suitable representations.

A major gain can be achieved by using multiple cell lines, but the choice of algorithm is important to most benefit from the data heterogeneity. We investigated several approaches, and obtained the best results for an autoencoder with a domain discriminative component to promote domain-invariant features across multiple cell lines. This approach requires the same set of markers to be used and ideally the same set of drugs to be tested.

In addition of improving MOA prediction accuracy, this method further produces a representation that allows us to compare effects of drugs on different cell lines. We use the representation in order to make comparisons between (drug, cell line) pairs. This is one of the most important use cases if the cell lines represent different molecular subtypes of a disease. Importantly, it allows one to identify highly specific drugs that only act on one particular subtype—the paradigm of precision medicine—and to distinguish them from drugs that are generally effective across different subtypes, as well as from drugs that lead to different phenotypic effects, which in turn suggest a target of different pathways depending on the transcriptional program.

While these approaches have only been applied to a small-scale pilot study, they provide an interesting starting point for larger multi-cell-line screens. A larger screen would, of course, provide greater opportunities to explore the strengths and weaknesses of our proposed method. It may be, for example, that our method is again advantageous for higher numbers of cell lines, yet it may happen that in the limit, performance converges to that of simpler methods, such as the vanilla autoencoders, as the variety of cell lines forces the autoencoder to learn domain-invariant features as a matter of course. In this case, our method will have provided a useful regularisation in the use case of lower numbers of cell lines.

With the exception of batch normalisation, we have neglected to borrow tricks of the trade from the GAN literature on adversarial learning. It is possible that our reported results underestimate the quality of our method as result. A future study could include an assessment of the effects of varying activation function (for example, LeakyReLU), alternating gradient descent steps, and learning rate scheduling, to name a few.

Finally, the domain-invariant features themselves remain mysterious. It

would be fitting to analyse the properties of those arising from our experiments and understand how they could be improved. Consider Information Maximising GANs (InfoGANs) (Chen et al. [2016]), a GAN with an information maximising regularisation function. InfoGANs allocate a subset c of the generator input noise vector z to act as a latent code, which is optimised to have maximum mutual information with the generator output. That is, one maximises $I(c; G(z, c))$. In practice, the discriminator predicts the distribution on c used in generation, given the generated image. If the discriminator is easily able to discern c given generated image x , then the mutual information is high, and the content of c has been preserved during generation. The only way to maximise the mutual information is, for each latent variable c_i , for the generator and discriminator to “agree” on a salient and independently-varying property of image objects (such as size and rotation), such that the auxiliary network can infer the original, independently-varying latent variable. In principle, the outcome is a generator that will generate images that are to some extent controllable by semantically meaningful properties. Information maximising is therefore a powerful prior for learning disentangled representations. Some similar mechanism could be attempted in our autoencoders to address the interpretability of the latent codes.

Part II

Fluorescence-free phenotyping

Chapter 5

Experimentally-generated ground truth for detecting cell types in phase contrast time-lapses microscopy

Summary: *One of the major obstacles in the use of deep learning for computational phenotyping is the need for extensive image annotation, which in many biological projects is infeasible. This chapter proposes strategies for the experimental generation of ground truth, allowing the design of a full cell detection and classification workflow without a single manually annotated cell. These strategies are applied to the analysis of image-based immunotherapy assays. In particular, chimeric antigen receptor (CAR) is an immunotherapy whereby T lymphocytes are engineered to selectively attack cancer cells. Image-based screens of CAR-T cells, combining phase contrast and fluorescence microscopy, suffer from the gradual quenching of the fluorescent signal, making the reliable tracking of cell populations across time-lapse movies difficult. In this chapter, the available fluorescent markers are leveraged as an experimentally-generated ground truth for phenotyping the cell population in time. It then compares two learning strategies. The first, based on predicting fluorescent markers directly from phase contrast microscopy is, in the first instance, a powerful visualisation system. The second, an object detection system, learns from an automatically annotated training set, acquired with some simple image processing of the image set. Depending on the experimental objectives, either approach has scope for potentially eliminating the need for the cumbersome fluorescent markers. This approach will underpin the development of cheap and robust microscope-based protocols to quantify CAR-T activity against tumor cell in vitro.*

Résumé: *L'un des principaux obstacles à l'utilisation de l'apprentissage profond pour le phénotypage informatique est la nécessité d'une annotation extensive des images, ce qui est impossible dans de nombreux projets biologiques. Ce chapitre propose des stratégies pour la annotation expérimentale des données d'entraînement, permettant la conception d'un flux de travail complet de détection et de classification de cellules sans une seule cellule annotée manuellement. Ces stratégies sont appliquées à l'analyse des essais d'immunothérapie basés sur des images. En particulier, le récepteur antigénique chimérique (CAR) est une immunothérapie dans laquelle les lymphocytes T sont conçus pour attaquer sélectivement les cellules cancéreuses. Les criblages des cellules CAR-T à base d'image, combinant le contraste de phase et la microscopie à fluorescence, souffrent de l'extinction progressive du signal fluorescent, ce qui rend difficile le suivi fiable des populations cellulaires à travers les films en temps réel. Dans ce chapitre, les marqueurs fluorescents disponibles sont utilisés comme annotation générée expérimentalement pour phénotyper la population cellulaire dans le temps. Deux stratégies d'apprentissage sont comparées. La première, basée sur la prédiction des marqueurs fluorescents directement à partir de la microscopie à contraste de phase, est, dans un premier temps, un puissant système de visualisation. La seconde, un système de détection d'objets, apprend à partir d'un ensemble d'apprentissage automatiquement annoté, acquis avec un simple traitement de l'image de l'ensemble d'images. En fonction des objectifs expérimentaux, l'une ou l'autre de ces approches peut potentiellement éliminer le besoin de marqueurs fluorescents encombrants. Cette approche soutiendra le développement de protocoles peu coûteux et robustes basés sur le microscope pour quantifier l'activité CAR-T contre les cellules tumorales in vitro.*

5.1 Overview

5.1.1 Biological context

Chimeric antigen receptor T-cell (CAR-T) therapy is an immunotherapy whereby T lymphocytes (a subtype of white blood cells) are engineered to selectively attack cancer cells. Generally speaking, CARs are engineered or *recombinant* receptors designed to target a specific protein, in practice a tumour antigen. When a CAR allows the CAR-T cell to latch onto a cancer-specific antigen, and deliver cytotoxic chemicals to induce *lysis* (membrane breakdown) of the target cell (Benmeharek et al. [2019]). CAR technology has 30 years of development encompassing several design generations (Maude et al. [2015]). As a therapy, T cells are extracted from a patient or healthy donor, modified (such as with viral transduction) to express the

CAR in culture, and infused back into the patient. The *in vivo* CAR-T cells then target tumours as a “living drug” treatment. The main engineering challenge is ensuring safety and efficacy, in particular the target specificity of the T cells (Sadelain et al. [2013]).

In particular, CAR-T therapy can be applied to almost all B-cell cancers, such as acute lymphoblastic leukemia, one of the most common and fatal forms (primarily from relapse) of pediatric cancer worldwide (Hunger and Mullighan [2015]). In the latter disease, CAR-T has achieved up to 90% complete remission rates in clinical studies such as Maude et al. [2014]. The used antigen is CD19, a B-cell surface protein. Ideally, this antigen will be cancer-cell-specific, however CD19 CAR-T leads to B-cell *aplasia*, that is, the depletion of B cells both cancerous and normal. While there are studies showing that side effects can be managed, the pursuit of new and more specific forms of CAR is the subject of rapid and enthusiastic research (Wang et al. [2017]).

Microscopy-based assays are an excellent tool to study the effects of CAR-T therapy in vitro. B-cell populations are exposed to engineered T-cells, and the effects can be recorded by live cell imaging. For these assays, it is important to observe the long-term effects of the T-cells on the B-cell population, ideally over several days and for a large cell population. Even at low resolution, specific markers allow for a quick assessment of how many cells died and thus how effective the CAR-T cells were in attacking cancer cells. Quantitative analysis of these movies thus allow to compare different constructs, which is an important asset in the research for the most effective therapies.

5.1.2 Computational phenotyping for phase contrast images

Phase contrast microscopy is an optical, label-free microscopy technique, invented in the 1930s. It works by converting phase shifts of light passing through a transparent specimen to image contrast. Due to the selective amplification of scattered light and the suppression of background illumination, phase contrast microscopy significantly increases the contrast of the imaged objects. Phase contrast microscopy revolutionised microscopy in and has remained a standard technique used in the labs.

Fluorescence microscopy was invented in the 1980s. As described in Section 1.1.2, it relies on the fluorescent labeling of biomolecules. The resulting image shows the spatial distribution of the labeled molecule with high contrast and virtually no background illumination. The advantage of this imaging technique is that one may choose which biomolecule to highlight. In so doing, one can gather information such as the presence of a certain signaling

protein or the sub-cellular structure the protein localises to (for example, microtubules, nuclear or plasma membrane, Golgi). The downside is that fluorescence microscopy does not provide a full picture of the cell, rather only information on the selected marker proteins. The choice of the fluorescent labels is therefore a key parameter for this type of microscopy.

Despite its power, fluorescence microscopy has various drawbacks, with several experimental complications, in particular when imaging assays are performed over several days. For live cell imaging by fluorescence microscopy, there are essentially two main labeling options: either one genetically encodes the fluorescent label as a *fluorescent tag* or one uses live dyes. The first strategy requires genetic modification of the cell lines, a lengthy process that leads the cell line astray from a physiologically relevant model, while the live dyes have the problem that they fade as the cell population grows, as the total amount is constant (and therefore divided among more and more cells). In addition, the fluorescent marking of cells is expensive and time-consuming, which is a problem for large-scale imaging assays.

In a microscope setting where both transmitted light and fluorescence microscopy can be taken for the same cells simultaneously, interesting opportunities arise. Recently, successful attempts have been made ([Christiansen et al. \[2018\]](#), [Ounkomol et al. \[2018\]](#)) to predict fluorescent signals from transmitted light images, demonstrating that for certain biological experiments, the relevant information is wholly contained in the phase contrast signal. This is an attractive prospect because it means that one can in principle benefit from the ease and non-invasiveness of phase contrast microscopy, yet still benefit from the advantages and the specificity of fluorescence microscopy, whose reconstruction is learned in a calibration step prior to the imaging experiments. In this Chapter, fluorescent markers are leveraged to quantify the phenotype of cultured cells from the phase contrast alone. Two approaches to this quantification are studied: the first, based on fluorescent labeling, is described in [Section 5.3](#); the second, an object detection system, in [Section 5.4](#). In the last part of the Chapter, the two approaches are compared.

5.2 CAR-T dataset

The data behind this study consists in live-cell imaging experiments performed on an IncuCyte machine. In these experiments, the disease is modeled by Raji, an immortalised cell line of B lymphocytes from a 1963 Burkitt's lymphoma patient. Raji cell populations are studied in isolation, as well as cocultured with CAR-T cells. The setup is detailed in [Table 5.1](#). The row A wells study isolated Raji cells; the row B wells study the coculture. Each well

is imaged at $5\times$ magnification in four fields of view of size 1408×1040 pixels. Images were taken every two hours over a 220 hour period (110 frames apiece) producing 32 time lapse movies. Each row of the microplate studied consists of two groups of two replicates, where each group has uses different seeding densities. In this analysis, however, the groups are considered to be interchangeable and, where convenient, the data is pooled within each plate row. Each frame of each time lapse movie pairs a phase contrast image with green fluorescent protein (GFP) and mCherry fluorescent images, depicting the same scene. A sample is given in Figure 5.1. The mCherry fluorescence is present in all Raji cells while the GFP only appears in dead cells. The T cells (only active in the row B experiments) are only visible on the phase contrast channel. Thus, fluorescent markers combine for a quasi-annotation of the cells, which may be expressed with the (fuzzy) logic,

$$\text{IF } \neg \text{object THEN class IS background} \quad (5.1)$$

$$\text{IF object} \wedge \text{GFP THEN class IS dead Raji} \quad (5.2)$$

$$\text{IF object} \wedge \text{mCherry} \wedge \neg \text{GFP THEN class IS Raji} \quad (5.3)$$

$$\text{IF object} \wedge \neg \text{mCherry} \wedge \neg \text{GFP THEN class IS CAR-T} \quad (5.4)$$

	1	2	5	6
A	raji-target cells (1) 30K cells / well		raji-target cells (1) 30K cells / well	
B	CAR June 1:2 (1) 30K cells / well raji-target cells (1) 30K cells / well		CAR June 1:2 (1) 15K cells / well raji-target cells (1) 30K cells / well	

Table 5.1: Characteristics of CAR-T experiments studied. Row A studies RAJI cells in isolation; row B studies cocultured Raji and CAR-T cells.

Note the high volume of data: the cells are seeded to a total of up to 60000 per well, with four wells per experiment type, each with four fields and 110 time lapse frames. Coupled with the fluorescence annotations, this is, in principle, a veritable treasure trove of data for deep learning. Annotation by experiment is a promising strategy: a large ground truth can easily be collected and, in addition, the “experimental ground truth” is much more objective than a manual one. A similar strategy has already been applied to image segmentation (Sadanandan et al. [2017]). This potentially overcomes the main bottleneck in leveraging deep learning models in such experiments. On the other hand, most cells fit inside a 14×14 pixel window (contrast this

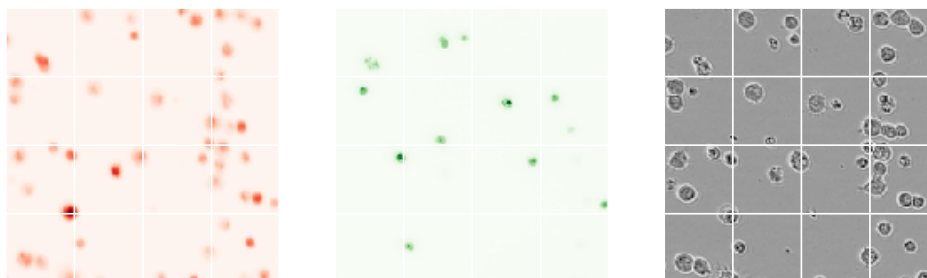


Figure 5.1: Aligned image channel crops (200×200 px) marking living Raji cells in mCherry (left), dead cells in GFP (center), and phase contrast (right).

with the much higher resolutions of cells imaged in Part I). This low resolution proves to be a constraint in the analyses to come, and an engineering challenge for the models.

5.2.1 Observations on the dataset

Even at the best of times it is highly advisable to perform an exploratory analysis of a dataset. This principle was put to get use in Chapter 3, as important insights were obtained that were carried through to Chapter 4. With the temporal dimension now thrown in, the CAR-T dataset proves to be highly dynamic, and becomes increasingly chaotic in time, in particular as the Raji B-cells undergo mitosis. Some notable phenomena observed in the experimental data are detailed presently. These prove to be influential factors in the methodological designs.

Apoptotic cells acquire GFP

Figure A.5 traces the death of a Raji cell in full fluorescence. One may observe some subtle morphological changes such as a reduction in size, as well as a change of texture, which remain visible on the phase contrast channel. The fluorescence, however, shows a clear accumulation of GFP fluorophores. Here, the average intensity of each fluorescent channel is measured in the cell region of interest and plotted over time in Figure 5.2. The GFP signal rises for at least 10 frames, corresponding to a 20 hour period. The mCherry signal declines in time, seemingly in accordance with the quenching effect.

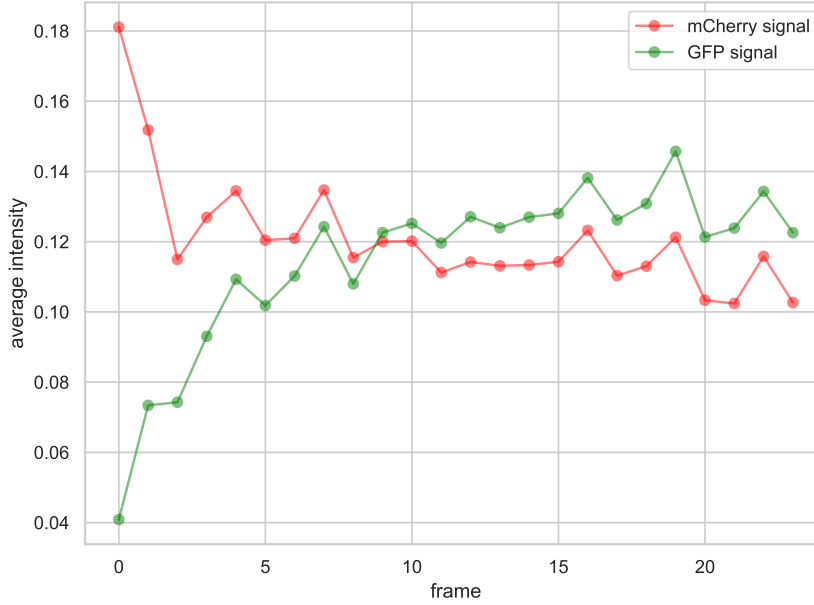


Figure 5.2: Tracking apoptosis of a Raji cell. GFP signal accumulates as a cell undergoes morphological changes (a)-(f).

Proliferation of Raji cells leads to cell clumps

Proliferation greatly complicates the task of counting cells, and influences the methods in the current chapter as well as the next. It is well known that Raji cells group to form clusters or “clumps” (Epstein et al. [1966]). Figure 5.3 shows a sequence of cropped frames centered on a proliferation of cells. From a handful of dispersed cells in the initial frame ($t = 50$), cell division occurs in time ($t = 70$), ($t = 80$), ($t = 90$), creating clusters of closely touching (or even overlapping) cells. Whatever methodological approach is taken, cell proliferation complexifies the detection and segmentation of individual cells as the image background can no longer be utilised to these ends. Whereas at first the cell contours are well-defined, the more the cells multiply, the less clear their separation, as cells begin to overlap.

CAR-T cells create clusters of lysed target cells

With the inclusion of T cells, the picture becomes even more complicated. Figure 5.4 shows a series of crops taken at 30 hour intervals from well B6. T cells can here be seen as non-fluorescent objects with an elongated morphol-

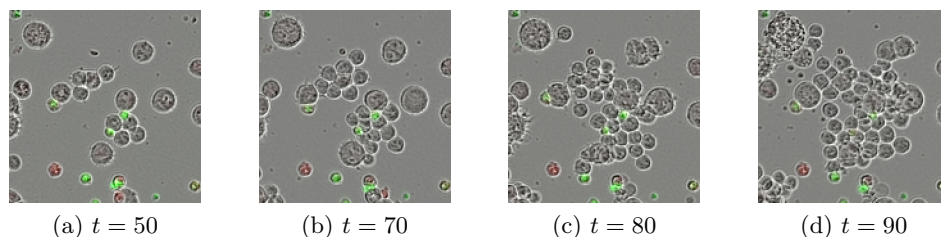


Figure 5.3: Raji cell population over time. Proliferation generates cell clusters.

ogy. Their activities involve the killing of B cells, the lysed (fragmented) remains of which are shepherded into tight clusters of cellular matter. The formation and ultimate merging of several such clusters is observable in Figure 5.4. An example of a CAR-T cell latching onto a Raji antigen is also given, with the subsequent death of the Raji cell in Appendix A.4. From a modeling perspective, individual cells are impossible to discern from such entropic clusters, as the membrane that provides each cell with its identity has been lost. Perspectives for handling these cases are offered in Section 5.5.

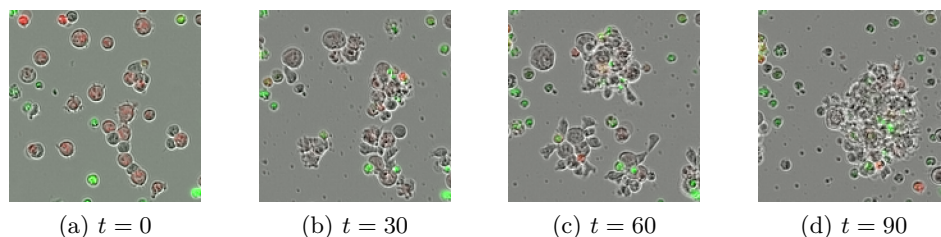


Figure 5.4: CAR-T cells (devoid of fluorescence) attack Raji B cells by latching onto Raji cell surface antigens and delivering cytotoxic chemicals. The induced lysis of the target Raji cells yields growing clusters of cellular matter.

5.2.2 Coping with fluorescent quenching

Image-based screens of CAR-T cells, combining phase contrast and fluorescence microscopy, suffer from the gradual quenching of the fluorescent signal, making the reliable tracking of cell populations across time-lapse imagery difficult. We saw across Figures 5.2, 5.3, and 5.4 evidence of the diminished mCherry (red) fluorescent signal in the later frames of the time lapse movies. In Figure 5.5 we see how the distribution of total image fluorescence diminishes across time, regardless of well. Already, the quenching effect obfuscates

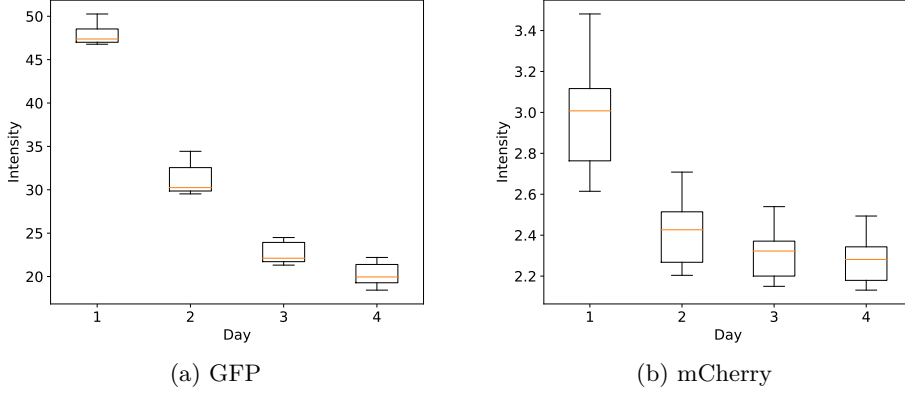


Figure 5.5: Comparison of average GFP (left) and mCherry (right) fluorescence measured across different fields of view at 24 hour intervals. One may observe the quenching effect of fluorescence over time, which occurs most rapidly in the first increment.

the interpretation of the experiments by lab technicians. It is furthermore a complication for learning algorithms, as fluorescence becomes inversely correlated with the various cellular phenomena occurring in the latter stages of the experiment, for example: the dissemination of certain large cells due to cell growth, and the appearance of cell clusters due to cell proliferation and clustering, as illustrated above. Learning only on earlier frames, where the fluorescence is strongest, is unlikely to suffice. To mitigate the quenching effect, each pixel x of image $\mathbf{x}$ is normalised to,

$$x' = \min\left(1, \frac{\bar{\mathbf{x}}_0}{\bar{\mathbf{x}}} \cdot \frac{x - \min(\mathbf{x})}{\text{perc}_{99}(\mathbf{x}) - \min(\mathbf{x})}\right), \quad (5.5)$$

where $\bar{\mathbf{x}}_0$ and $\bar{\mathbf{x}}$ are the means of the initial image and the image to be normalised, and perc_{99} returns the 99th percentile intensity. Intensities below some small threshold τ are set to zero to remove background noise. Note that the potentially naive assumption is made that the loss of fluorescence is entirely due to quenching, and not the phenotypic changes in the population.

5.3 Fluorescence prediction

In biological experiments pairing transmitted light and fluorescent signals, the fluorescence may often be largely inferred from the transmitted light image alone, as in [Christiansen et al. \[2018\]](#), [Ounkomol et al. \[2018\]](#), and

Lee et al. [2018]. That is, given a pair $\mathbf{x}, \mathbf{y}$ of images from the transmitted light X and fluorescence Y image domains, the fluorescent labeler F learns a mapping such that,

$$F(\mathbf{x}) \approx \mathbf{y}. \quad (5.6)$$

Note that the prospects for predicting fluorescent will always depend on, among other things, the available image resolution. While Ounkomol et al. [2018] predicted fluorescent images of nucleoli, nuclear envelope, and microtubules from transmitted light images with a median Pearson correlation coefficient $r \geq 0.8$, “painting” other organelles such as desmosomes and Golgi apparatus worked far less well $r \leq 0.2$. In the experiments of this chapter, the relatively low resolution of cells must therefore be compensated by high-level morphological cues associating phenotype with fluorescence.

In the following, a family of models for performing fluorescence prediction of phase contrast images automatically are presented. The motivations of the system are twofold: firstly, it can be used as a visualisation tool; secondly, it may serve as an intermediate step in a cell counting system. Naturally, the quality of the latter function depends on that of the former.

5.3.1 Image-to-image translation models

Though not the only fluorescence labeler in the literature, the “F-Net”, proposed by Ounkomol et al. [2018], is a suitable baseline for the problem at hand, on the grounds that it is essentially a repurposed U-Net (Ronneberger et al. [2015]), a widely studied architecture. This architecture has a deep encoder-decoder structure, such as the autoencoders studied in Section 4.2.2, albeit fully convolutional, and with *skip connections* concatenating each up-sampling layer with its opposite downsampling layers. The full architecture specification is provided in Figure B.3. The network is trained against the objective function,

$$\min_F \mathcal{L}_{L_2}(F) = \mathbb{E}_{\mathbf{x}, \mathbf{y}} [\|\mathbf{y} - F(\mathbf{x})\|_2^2], \quad (5.7)$$

that is, to minimise the expectation of mean square error over the distribution of input images $\mathbf{x}$ and target images $\mathbf{y}$, where $F : \mathbf{x} \rightarrow \mathbf{y}$ is the labeler network parameterised by a set of weights. The model is thus trained as a pixel-wise regressor. In the use cases to come, the input images are the phase contrast images, and the target images are the fluorescent channels. By default, the network is trained to predict a single fluorescent channel (as in Ounkomol et al. [2018]), producing one model per channel. Evidently, one

could train a multi-task labeler to predict a multiplexed fluorescent output. Experiments with this approach did not yield a significant improvement in this setting, however.

Roughly speaking, the F-Net model is a special case of the image-to-image translation models proposed in [Isola et al. \[2017\]](#). This family of `pix2pix` models may be formulated,

$$\min_G \max_D \mathcal{L}_{PIX}(D_{patch}, G) = \mathcal{L}_{CGAN}(D_{patch}, G) + \lambda \mathcal{L}_1(G), \quad (5.8)$$

where $\mathcal{L}_1(G) = \mathbb{E}[\|G(\mathbf{x}) - \mathbf{y}\|_1]$ is mean absolute error with tuning hyper-parameter λ and $\mathcal{L}_{CGAN}$ is the conditional GAN (CGAN) objective (see [Section 2.6.2](#)),

$$\mathcal{L}_{CGAN}(D, G) = \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})}[\log D(\mathbf{x}|\mathbf{y})] + \mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})}[\log(1 - D(G(\mathbf{z}|\mathbf{y})))] \quad (5.9)$$

where D is the discriminator and G is the generator, equivalent to F in all respects aside from it using a hyperbolic tangent ($\tanh$) function in the output layer (as is conventional for DCGANs, see [Section 2.6.1](#)) rather than a linear one. The variable $\mathbf{z}$ refers to the generator’s noise input. In the image-to-image setting, the discriminator network is a fully-convolutional network denoted “PatchGAN” (see [Appendix B.4](#) for an implementation),

$$d_{patch} : X, Y \rightarrow [0, 1]^{h \times w}, \quad (5.10)$$

producing an $h \times w$ grid of outputs rather than a single neuron, with each grid element critiquing an overlapping region of the input image. The grid of outputs are then averaged to give the discriminator,

$$D_{patch}(\mathbf{x}, \mathbf{y}) = \frac{1}{hw} \sum_{i=0}^h \sum_{j=0}^w d_{patch}(\mathbf{x}, \mathbf{y})_{ij}. \quad (5.11)$$

In practice, the conditioning is implemented as a channel-wise concatenation of condition image $\mathbf{x}$ and the target $\mathbf{y}$. [Isola et al. \[2017\]](#) make the insightful observation, however, that a GAN learns its own loss function via the discriminator, which penalises “unrealistic” images. As such, it is an infinitely flexible methodology and the discriminator D can be understood to act as an adaptive loss function, analogous to the way features are learned and optimised for the task of classification within the layers of a deep classifier. The choice of L_1 over an L_2 auxiliary loss is based on the empirical observation that L_2 leads to blurry results, something observe presently. The `pix2pix`

system has been demonstrated to be effective on a wide range of image-to-image translation problems. Just as with U-Net, they can achieve impressive results, while trained on a relatively small number of images.

In addition to the above, the F-Net F is also trained against a mean absolute error loss,

$$\min_F \mathcal{L}_{L_1}(F) = \lambda \mathcal{L}_1(F). \quad (5.12)$$

All models (Equations 5.7, 5.8, and 5.12) were trained on 1120 non-overlapping 256×256 px phase contrast crops paired with fluorescence crops selected from the first 14 fields of view from each experimental setting separately (rows A and B of the plate). Recall each fluorescent channel is trained on separately, producing separate models. The cropping strategy sampled from the first frame of each 24-hour window, for the first 96 hours of each time lapse movie. This strategy was adopted to maximise the available variation in the training data, without losing too much to the fluorescent quenching effect of later frames, while at the same time retaining the convenience of keeping all images in memory. Validation and testing was performed according to the same strategy on independent fields A06_03 and A06_04. Batches of a single image were sampled randomly and all models were trained with the Adam optimiser with $(\beta_0, \beta_1) = (0.5, 0.999)$ and maximum learning rate 2×10^{-4} .

5.3.2 Results

In practice, a fluorescence prediction ought to be considered good if the placement of its predictions are accurate, even if it does not match the ground truth intensities accurately. For this reason Ounkomol et al. [2018] propose an evaluation based on the Pearson correlation coefficient (PCC),

$$r = \frac{\sum(x - \bar{x})(y - \bar{y})}{\sqrt{\sum(x - \bar{x})^2 \sum(y - \bar{y})^2}}, \quad (5.13)$$

over all pixels x in the prediction and y in the test image. Figure 5.6 compares the average PCC correlation across all test images for each of the three models and each mode (GFP, mCherry) on the Raji-only experiments. The best overall performance comes from the L_2 model with mean 0.7617 and standard deviation 0.1049 for mCherry labeling, and mean 0.6163 and standard deviation 0.0931 for GFP. The pix2pix system performs at mean 0.7573 and standard deviation 0.0814 for mCherry, and mean 0.5848 and standard deviation 0.0972 for GFP. Finally, the L_1 model performs at mean 0.7720

and standard deviation 0.0727 for mCherry, and 0.5115 and standard deviation 0.1099 for GFP. Paradoxically, the L_2 model appears, by inspection, to generate worse results, with a significant amount of fluorescent “spillage” and blur effects. However, this noise can be removed with a simple thresholding operation. The pix2pix outputs look initially sharper, but ultimately fail more often to apply fluorescence where it counts, leading to a greater number of false negatives. On the other hand, it is also clear that the L_2 and L_1 models exhibit a greater degree of mode collapse than the pix2pix model, which necessarily varies its fluorescent prediction to continuously fool the discriminator.

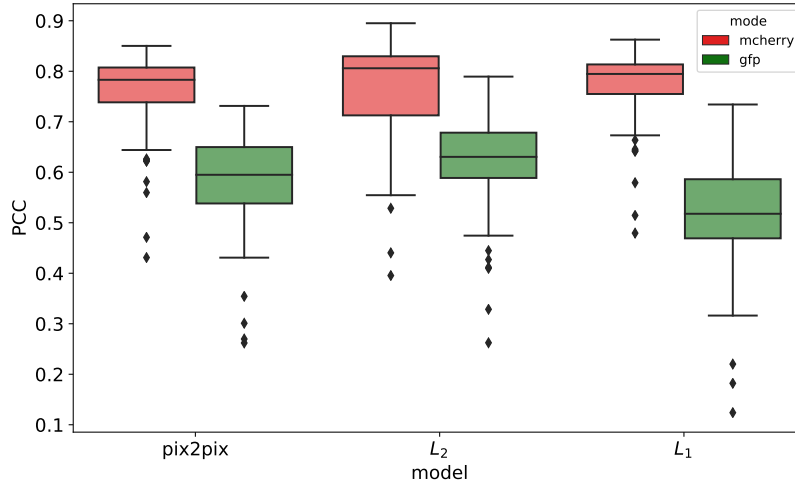


Figure 5.6: Pearson correlation coefficients for outputs of three fluorescent labelers, for both mCherry and GFP fluorescence prediction, measured over 80 test images.

Visualising full fluorescence outputs

An important qualitative test is to visualise the joint model outputs in full fluorescent colour, as well as in time lapse. To produce an appealing output, a “camera shake” effect of the plate moving inside the microscope must be corrected. Between successive frames, this is usually no more than $10\mu m$, corresponding to about 7px, in any direction. The necessary correction $(\delta x, \delta y)$ is found frame by frame by brute force over a suitable range of offsets. Fortunately, the Raji cells do not move much over time, and therefore the right offset can be found as the one that minimises the squared error between the frames. The images are each cropped slightly to account for the extents of the cumulative displacements in each direction caused by the shake. This tends to be a small amount, apparently with the camera

displacements occurring around a fixed center of mass. Lastly, the frames of the fluorescent channels are corrected by the same offsets discovered in the phase contrast.

The RGB pixel at position (i, j) of each full fluorescence image I is set as,

$$I_{ij} = [M_{ij} + P_{ij}, G_{ij} + P_{ij}, P_{ij}], \quad (5.14)$$

where M is the mCherry channel, G is the GFP channel, and P is the phase contrast image. The values are further clipped to the range $[0, 1]$. Figures 5.7 and 5.8 compare synthesised images combined from the respective `pix2pix` fluorescent labelers to the corresponding ground truth fluorescence. The errors are manifest, yet the models do a fine job of discerning cell types, implicit in their application of GFP over dead Raji cells in the Raji-only experiment (Figure 5.7) and its leaving T cells unlabelled in the full CAR-T experiment (Figure 5.8), as per the logic of Equation 5.1. The predicted fluorescence intensity levels also closely approximate those of the ground truth. Nevertheless, it should be noted that the mutual information between fluorescence intensity and phase contrast morphology may ultimately be too small to always guarantee an accurate prediction of intensity. For this reason, one must accept, for example, that certain dead cells appear more yellow than green, and vice versa. One possible extension of this work could be to incorporate time information into the prediction, which might alleviate this effect. Ultimately, however, it may be confirmed in the broader picture, that the fluorescent prediction is a success as a visualisation tool, at least for the earlier frames of the time lapse, and it ought now be considered how to incorporate it into a practicable tool for experimental use.

Full 110-frame time lapse movies comparing prediction (left) and ground truth full fluorescence (right) may be found online: Raji-only ($256 \times 256\text{px}$)¹, Raji-only ($512 \times 512\text{px}$)², full CAR-T ($256 \times 256\text{px}$)³, and full CAR-T ($512 \times 512\text{px}$)⁴.

5.3.3 Bridging the gap to cell detection

In the present problem context, it is of interest to derive a quantitative profile from live cell imaging data. While effective as a visualisation tool, fluorescent prediction only goes partway towards a full quantification of the phase contrast contents. Consider sample outputs for densely clustering

¹https://jcboyd.github.io/assets/car-t-videos/raji_target/256.mp4

²https://jcboyd.github.io/assets/car-t-videos/raji_target/512.mp4

³https://jcboyd.github.io/assets/car-t-videos/CAR_June/256.mp4

⁴https://jcboyd.github.io/assets/car-t-videos/CAR_June/512.mp4

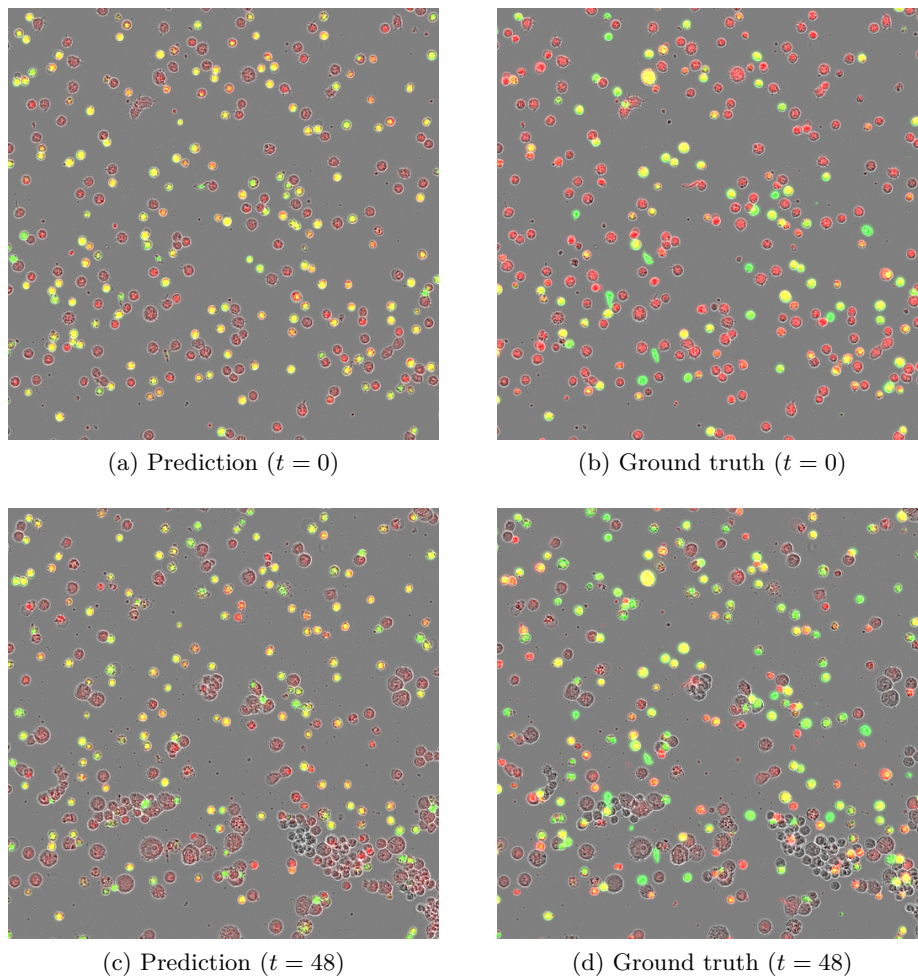


Figure 5.7: Full fluorescence for indicative ($512 \times 512\text{px}$) crop from an Raji-only experiment. Columns distinguish fluorescent labeler predictions (left) and ground truth (right); rows distinguish times an early frame ($t = 0$) (top) and a later one ($t = 48$) (bottom).

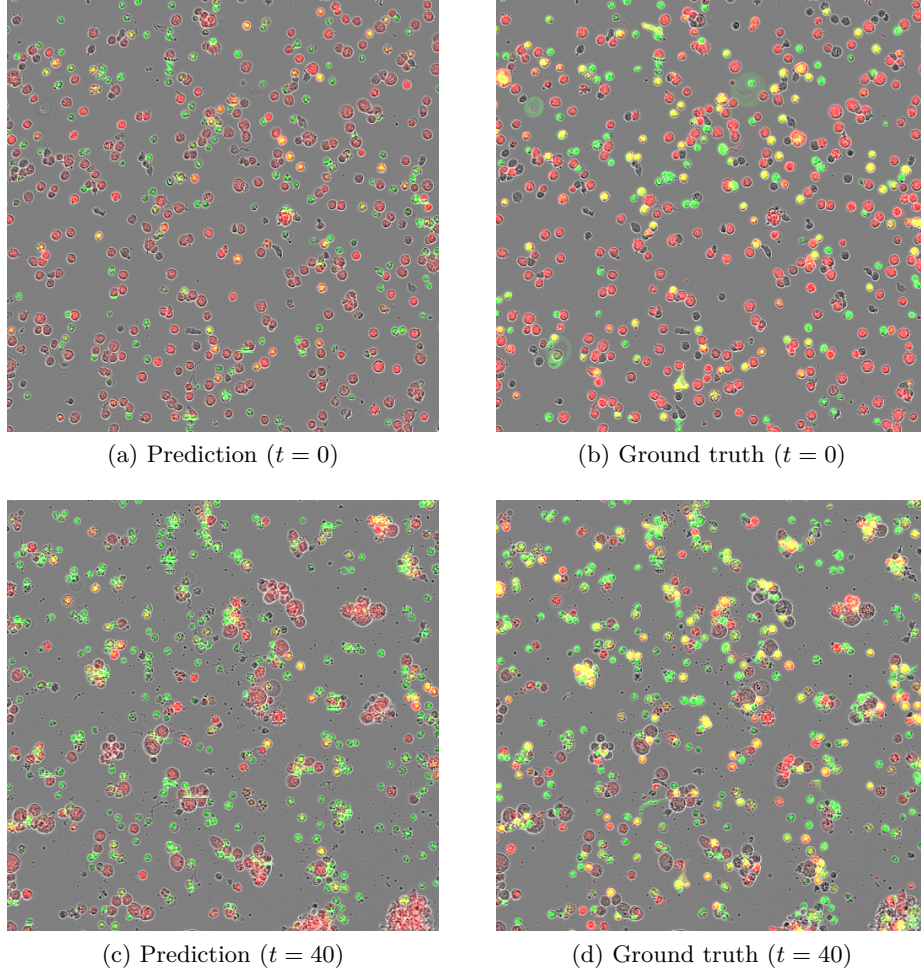


Figure 5.8: Full fluorescence for indicative ($512 \times 512\text{px}$) crop from a CAR-T experiment. Columns distinguish fluorescent labeler predictions (left) and ground truth (right); rows distinguish times an early frame ($t = 0$) (top) and a later one ($t = 40$) (bottom).

cells, given in Figure 5.9. Here we can see how the task of counting cells is far from complete following fluorescence prediction, with dense “clouds” of fluorescence corresponding to masses of cells. One can imagine all sorts of *post hoc* solutions for disambiguating these fluorescent clouds.

Blob detection for disambiguating fluorescent labels

As Raji cells have a consistent size and simply, circular geometry, one approach to counting individual cells from the fluorescence prediction outputs is with *blob detection*. A simple approach to blob detection is with a Laplacian of Gaussian (LoG) filter. That is,

$$\Delta G_\sigma(x, y) * I(x, y) = [\nabla_{xx} G_\sigma(x, y) + \nabla_{yy} G_\sigma(x, y)] * I(x, y), \quad (5.15)$$

where Δ is the Laplacian operator (sum of second partial derivatives), G_σ is a Gaussian filter with standard deviation σ and I is an image. Blobs distribute intensities in a Gaussian-like way, and the Gaussian pre-filter smooths the image. The second derivative of a Gaussian has a minimum in the center of the bell curve, thus marking the center of the blob. The maxima detected in the resultant image are the detected blobs. Usually the filter is performed over a range of scales, from which are chosen the maxima.

In practice, the tunable parameters for LoG blob detection are the range of standard deviations $\{\sigma_i\}$ to test for (corresponding to the anticipated size range) and a threshold θ on the minimum permissible intensity. These were tuned by hand in an example of this approach to blob detection of fluorescent outputs provided in Figure A.6.

This simple approach to blob detection, however, is found to not suffice for cases such as in Figure 5.9. While some system could surely be devised to infer the number of cells from a mass of fluorescence, in the following section we will see an alternative approach based on the fundamentally distinct methodology of object detection, which makes counting cells a primary objective.

5.4 Object detection system

This section contains an extended version of a paper published in the proceedings of the International Symposium for Biomedical Imaging in 2020.

We found in the previous section a limitation on the scope of fluorescent prediction for counting cells. This brings us back to a more conventional high content analysis (HCA). In Chapter 4 the concern was with generating

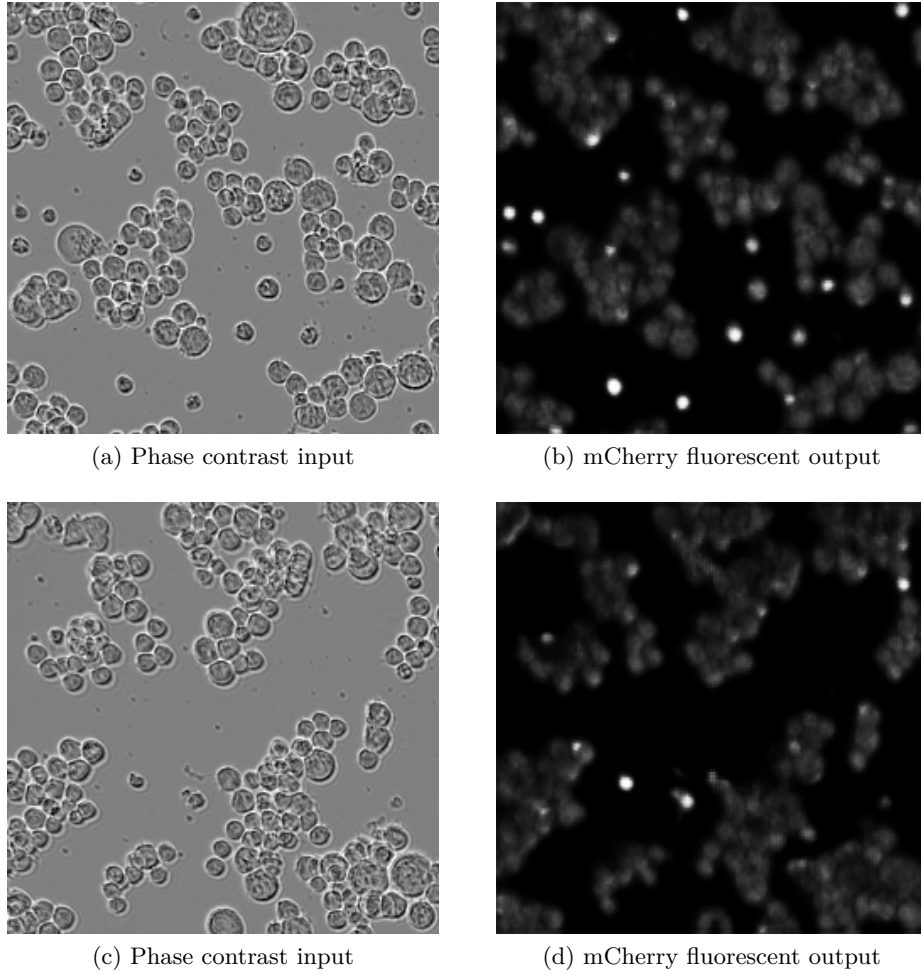


Figure 5.9: Fluorescent labeler outputs produce dense clouds of mCherry fluorescence for two manually selected phase contrast inputs. These outputs may be difficult to disambiguate.

a phenotypic profile that characterised a drug effect, potentially with uninterpretable intermediate features. In the present context, we are concerned with the more comprehensible task of counting cell types in time. Note that this task conforms to the conventional pipeline (see Section 1.1.3). In particular, the dimensionality reduction step will be fulfilled by the attribution of a cell type,

$$\mathbf{z} = f(\mathbf{x}), \quad (5.16)$$

where $\mathbf{z} \in \{0, 1\}^K$ and $\sum_{k=0}^K z_k = 1$ for K the number of cell classes, that is, $\mathbf{z}$ is one-hot, for extracted cell features $\mathbf{x}$, and cell classifier f . The cell population is then summarised by aggregation as a simple summation or average, giving the number or proportion of cells per cell class respectively in the phenotypic profile,

$$\mathbf{p} = \frac{1}{N} \sum_{i=1}^N f(\mathbf{x}_i). \quad (5.17)$$

As we shall see, in this section the first three stages of the HCA pipeline are achieved by a single neural network.

A phenotypic profile of this nature summarises a single image frame. However, we are interested in summarising a full time lapse movie. Therefore, our aim is to create a set of such count profiles,

$$\mathbb{P} = \{\mathbf{p}_i\}_{i=0}^T, \quad (5.18)$$

where $\mathbf{p}_i \in \mathbb{R}^K$ for K classes is the i th profile in a series of T frames. This ordered set amounts to a set of K time series, one per class, of length T , and is the final output in Section 5.4.3.

Section 5.4.1 describes an acquisition and preprocessing pipeline for an experimentally-generated ground truth with which to train the detection system. Section 5.4.2 specifies this system, and Section 5.4.3 we describe our evaluation methodology and report model performance on two manually annotated datasets. This section considers only the Raji-only experimental setting. Note, however, that the methodology should naturally extend to the full CAR-T setting also.

5.4.1 Experimentally-generated ground truth

Our pipeline (see Figure 5.10) begins by applying a Gaussian filter (tuned to $\sigma = 2$) to the phase contrast image. We then segment cells by subtracting

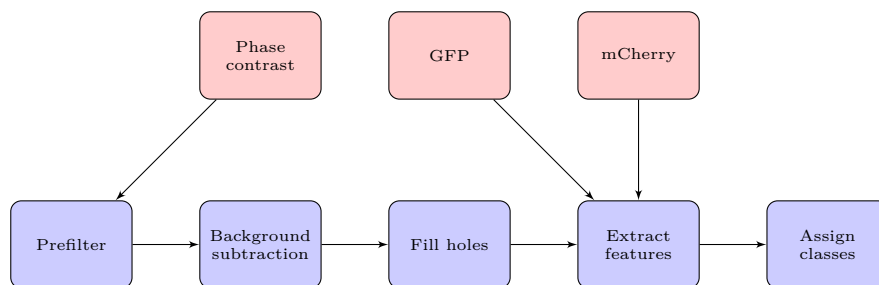


Figure 5.10: Pipeline for automatic construction of ground truth for training object detection system. Basic image processing steps indicated in blue, image inputs indicated in red.

a background image formed with a mean filter of diameter 19px, before clipping to zero as in [Walter et al. \[2010a\]](#). We fill object holes with a morphological reconstruction by erosion and use a morphological opening to remove small details. An example of this pipeline depicted in [Figure A.7](#). We further filter objects outside a reasonable size range ($< 6 \times 6\text{px} \approx 50\mu\text{m}^2$, determined by ranking cell areas), as these tend to be dust and other non-cellular particles on the well surface. An Otsu threshold on the distribution of averaged GFP signal per cell is then used to allocate a class (living/dead) for each connected component (see [Figure 5.11](#)). To train our object detector ([Section 5.4.2](#)), we also randomly sample background crops from the images, allowing for partial overlaps with cells.

5.4.2 Object detection system

In order to track cell phenotype populations over time, we require a robust object detection system to identify individual cells. The core of our system is a convolutional neural network and is detailed below.

Training as a classifier

Our preprocessing pipeline is imperfect and does not give a complete annotation of the cell populations as would be required by state-of-the-art detection systems such as [Redmon et al. \[2016\]](#). We therefore opt for crop-wise training, where the bounding boxes of successfully segmented cells are padded, to create $24 \times 24\text{px}$ crops, centered on the cells. Due to the low image resolution, we found this sizing provided sufficient contextual information to the network. Combined with background crops, this amounts to approximately 100,000 training examples in three classes. Samples are given in [Figure 5.12](#).

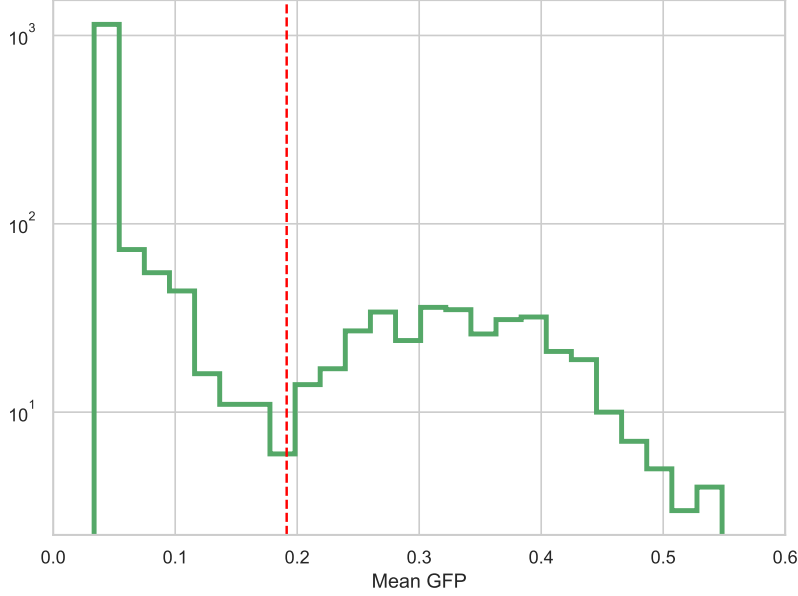


Figure 5.11: Living and dead Raji cells revealed as distinct modes of a bi-modal distribution on mean GFP fluorescence intensity per connected component of cell segmentation output.

Our network architecture is detailed in Table 5.2. All weighted layers have a ReLU activation, except Output_o and Output_c , which have softmax activations, and Output_b , which remains linear. The convolutions are all valid, and a 24×24 px input image is reduced to 1×1 px by the final layer. We implement this network in the `Keras` deep learning framework [Chollet et al. \[2015\]](#) and all code for our system is publicly available⁵. We train the network with stochastic gradient descent with learning rate 5×10^{-3} and Nesterov momentum ($\mu = 0.9$). Mini-batches of size 128 are sampled stochastically and simple data augmentation (horizontal and vertical flipping) is performed on the fly. We regularise the network with batch normalisation [Ioffe and Szegedy \[2015\]](#) and weight decay ($\lambda = 3 \times 10^{-5}$).

Inspired by [Redmon et al. \[2016\]](#), we formulate a multi-task prediction in which we predict $Pr(o)$, where o indicates the presence of an object in the center of the receptive field and, separately, $Pr(c|o)$, that is, the probability of cell phenotype class c given the presence of an object. These probabilities are combined at inference time (Section 5.4.2). In addition, our network performs regression on the height h and width w of the bounding box of the cell, measured as a fraction of the crop size from the crop centre. This information is readily available when generating the training set. Note that

⁵<https://github.com/jcboyd/detecting-lymphocytes>

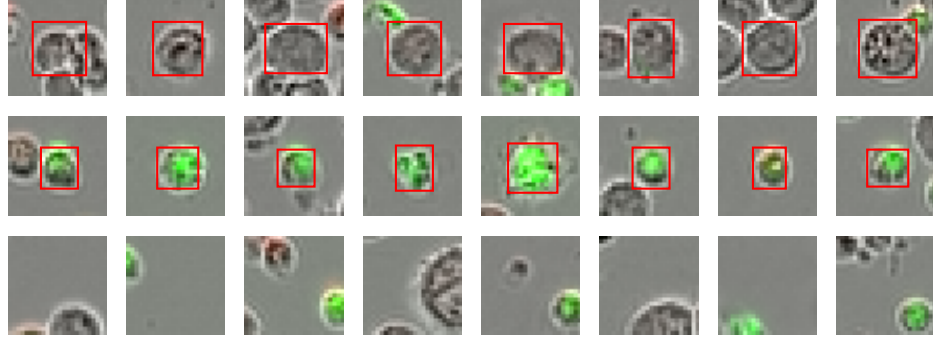


Figure 5.12: Samples of living Raji cells (top), dead cells (middle), background (bottom) annotated with bounding boxes. Fluorescence is included for clarity only and is not used in training.

Layer	Connection	Size	Output ($w \times h \times d$)
Input	-	-	$24 \times 24 \times 1$
Conv ₁	Input	3×3	$22 \times 22 \times 16$
Conv ₂	Conv ₁	3×3	$20 \times 20 \times 16$
MaxPool ₁	Conv ₂	2×2	$10 \times 10 \times 16$
Conv ₃	MaxPool ₁	3×3	$8 \times 8 \times 64$
Conv ₄	Conv ₃	3×3	$6 \times 6 \times 64$
MaxPool ₂	Conv ₄	2×2	$3 \times 3 \times 64$
Conv ₅	MaxPool ₂	1×1	$1 \times 1 \times 128$
Conv ₆	Conv ₅	1×1	$1 \times 1 \times 128$
Output _o	Conv ₆	1×1	$1 \times 1 \times 1$
Output _c	Conv ₆	1×1	$1 \times 1 \times 1$
Output _b	Conv ₆	1×1	$1 \times 1 \times 2$

Table 5.2: Specification of the network architecture. We distinguish three multitask outputs: Output_o, the probability of object presence; Output_c, the probability of class c given an object; and Output_b, the tuple of object width and height.

we make the assumption that our chosen crop size represents a hard maximum on the size of a cell’s bounding box, a reasonable simplification for our dataset. Our network is therefore trained to minimise the loss function,

$$\mathcal{L}(\mathbf{x}_i, o_i, c_i, w_i, h_i; \theta) = l_o(\hat{o}_i, o_i) + \mathbb{I}_o^i[l_c(\hat{c}_i, c_i)] + \mathbb{I}_o^i[l_b(\hat{w}_i, w_i, \hat{h}_i, h_i)], \quad (5.19)$$

with respect to model parameters θ , where l_o and l_c are each a standard

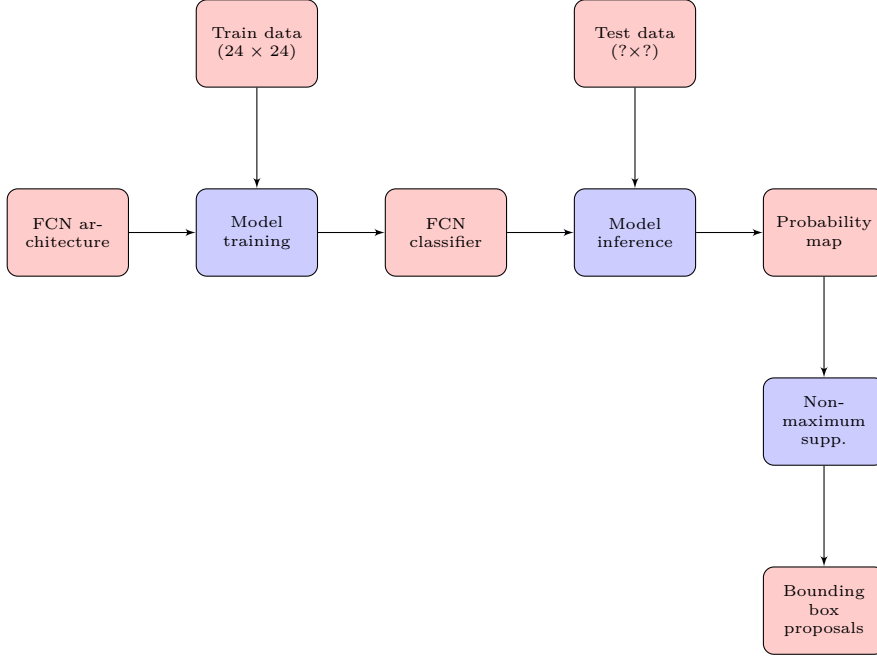


Figure 5.13: Pipeline for training a deployment of a fully convolutional classifier. Training may occur on fixed-sized input (24×24)px, but convolutions permit variable-sized inference.

cross entropy and $l_b(\hat{w}_i, w_i, \hat{h}_i, h_i) = (\hat{w}_i - w_i)^2 + (\hat{h}_i - h_i)^2$. The estimates $\hat{o}_i, \hat{c}_i, \hat{w}_i$, and $\hat{h}_i$ are the network outputs for object presence, object class, and bounding box width and height. The indicator function $\mathbb{1}_o^i = 1$ when training example $\mathbf{x}_i$ contains an object and $\mathbb{1}_o^i = 0$ otherwise.

We benchmarked our network as a classifier of cropped cells against a logistic regression trained on features extracted from a pre-trained 50-layer ResNet [He et al. \[2016\]](#), a standard CNN approach in bioimage analysis. In order to do this, we resized our cropped cells to 32×32 px and recorded the final convolutional layer of the ResNet, a vector of dimension 2048. This baseline achieved an accuracy of 0.83 on balanced test data, whereas our own network achieved 0.96. Though deep pre-trained networks are known to be powerful general-purpose feature extractors [Sharif Razavian et al. \[2014\]](#), they may also be over-parameterised for many problems [Raghu et al. \[2019\]](#).

Inference as a detector

Following [Sermanet et al. \[2013\]](#) we designed our network to be *fully convolutional* (FCN). A FCN is capable of performing inference on any size of input, and is extended naturally to object detection. The workflow is speci-

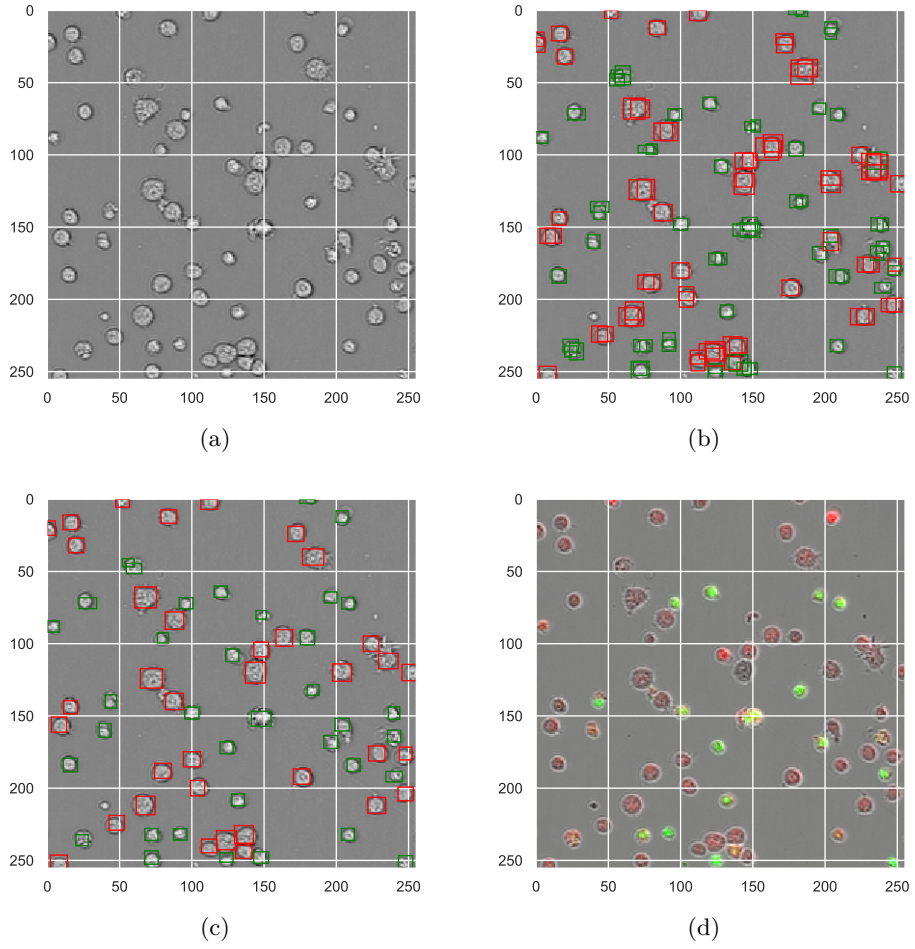


Figure 5.14: The processing of a test image: phase contrast input image (a); raw model bounding box predictions (b); non-maximum suppression post-processing (c); finally, for comparison, the corresponding full fluorescence image (d).

fied in Figure 5.13. Thus, once trained on cell crops as a classifier, inference may be performed on an entire image in a single forward pass, producing a map of softmax probabilities at every location in the image. Note that even on CPU, a full 1408×1040 px image is processed by the network in about 1s. Fully-convolutional whole-image inference emulates sliding-window detection, albeit without the tremendous inefficiency of executing the model separately at every spatial position. Note that the resolution of the output will depend on the number of pooling layers in the network. For example, our network includes two max pooling layers, hence we make detections at a stride of 4 across the input image domain.

At inference time, the object and conditional class probabilities are combined to give the marginal class probabilities $Pr(c) = Pr(c|o) \cdot Pr(o)$. Note that $Pr(c|-o) = 0$. These probabilities are thresholded and pruned with non-maximum suppression (NMS), providing a final detection mask for each class. For the NMS algorithm, we use an intersection over union threshold of 0.35. An example of this procedure is shown in Figure 5.14.

Smoothing probabilities in time

Because the cells are relatively stationary, we can improve the prediction of our system by leveraging information across time. We find a simple weighted average of prediction probabilities from consecutive frames, computed prior to NMS, improves overall performance. We thus define the *smoothed* probability $p_{ij}^{(t)} \leftarrow 1/4 \cdot p_{ij}^{(t-1)} + 1/2 \cdot p_{ij}^{(t)} + 1/4 \cdot p_{ij}^{(t+1)}$ for the probability at image position (i, j) at time t . The weights were tuned manually for both performance and parsimony.

5.4.3 Results

Evaluation strategy

To evaluate our system, we manually annotated three days worth of frames of size 256×256 px from each of two independent experimental replicates, totaling 72 images and approximately 7,000 test object detections. The replicates were chosen to represent different population dynamics: the first exhibits higher levels of cell proliferation; the second exhibits higher levels of apoptosis. We henceforth refer to these two datasets as *A* and *B* respectively. The annotations consist of manually annotated bounding boxes around the cells. We make this dataset publicly available along with the images used to train the network⁶. Note that despite this manually annotated evaluation

⁶<https://zenodo.org/record/3515446>

dataset, our model is still trained on a ground truth that is automatically generated from the experiment.

We score our detections in terms of the distance of the bounding box centers to the ground truth bounding box centers. We define the following metrics:

- True positive (TP) - a cell is detected in the vicinity of a ground truth cell.
- False positive (FP) - a cell is detected outside the vicinity of any ground truth cell.
- False negative (FN) - no cell is detected within the vicinity of a ground truth cell.

Here we define vicinity to be $\leq 10\text{px}$, the maximum distance a predicted cell center may fall from a ground true center while still falling within the typical cell bounding box ($14 \times 14\text{px}$). These metrics are computed per cell class, from which we calculate precision, recall, and F_1 scores. Note the F_1 score prevails over the commonly used Matthews correlation coefficient as it does not require us to define true negatives (a meaningless quantity in our framework). These are displayed in Tables 5.3 and 5.4 for test sets *A* and *B*. We see the effect of smoothing is globally positive, significantly improving the precision of the dead cell class, and giving the highest average F_1 scores of 83.86 for *A* and 81.19 for *B*. Note that the results on the dead cell class are markedly worse. We postulate this is due to the class imbalance at test time, as well as the difficulty of discerning individual cells from cell clusters.

Method	Class	Precision	Recall	F_1
Without smoothing	Living	0.8534	0.8636	0.8585
	Dead	0.7179	0.8693	0.7864
With smoothing	Living	0.8466	0.8883	0.8669
	Dead	0.7702	0.8549	0.8103

Table 5.3: Detection performance test set *A*, stratified by object class. Best results in bold.

Tracking population numbers over time

Our detection system is ultimately used to enumerate cell phenotypes over the course of CAR-T experiments. In Figure 5.15 we plot ground truth population numbers against the numbers inferred by our system. One can see the increasing number of living cells in Figure 5.15(a), corresponding to

Method	Class	Precision	Recall	F_1
Without smoothing	Living	0.9451	0.7778	0.8533
	Dead	0.6253	0.8935	0.7357
With smoothing	Living	0.9447	0.7957	0.8638
	Dead	0.6628	0.8904	0.7600

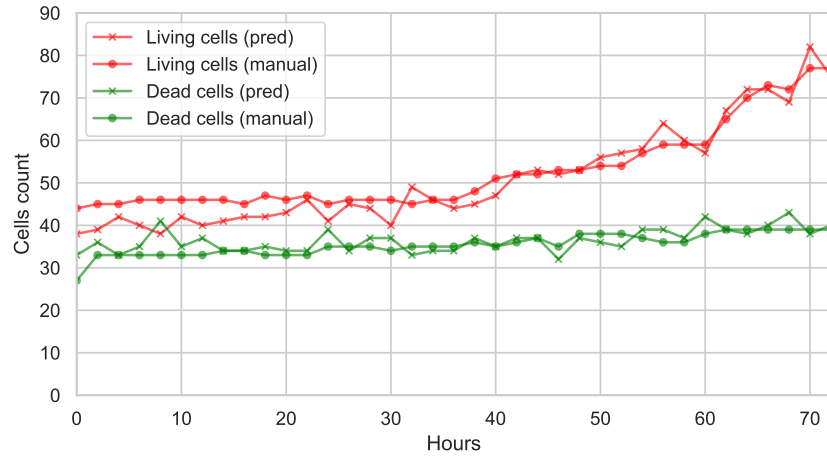
Table 5.4: Detection performance on test set B , stratified by object class. Best results in bold.

increasing amounts of cell division, whereas in Figure 5.15(b), one can see increasing amounts of apoptosis. In the former, our system achieves a mean relative error percentage of 5.95% and 5.56% (resp. living and dead cells) and 5.81% and 5.37% in the latter.

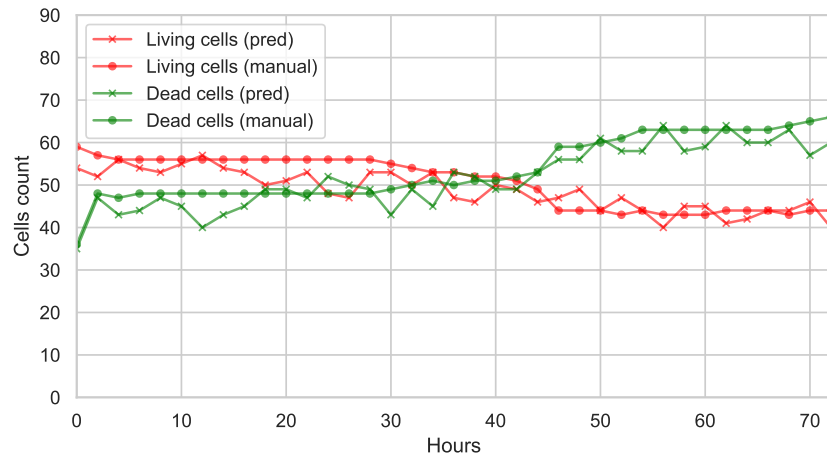
5.5 Discussion

This chapter has shown how fluorescence may be used to generate a robust ground truth for machine learning. It has presented two strategies for quantifying cells from phase contrast images, by leveraging paired fluorescence images. Section 5.3 showed how image-to-image translation models (including a generative adversarial variant) could be used as fluorescent predictors, and could successfully synthesise the fluorescent channels corresponding to a phase contrast microscopy input. Section 5.4 described the training of a neural object detection system and tested it on two manually annotated datasets. An example of how time information can be incorporated into the prediction task was also given. The system can likely be further improved with a more precise and expanded dataset.

Figure 5.16 compares the time series derived from applying each method for the complete frames of the first 72 hours of each time lapse movie in the Raji-only setting. The fields are pooled to obtain a total count for each of the four replicate wells of the Raji-only experiments at each time point. The time series for the detection system are obtained simply by counting bounding boxes as in Section 5.4.3. By contrast, the fluorescence predictor requires a more ad hoc approach. The central 1024×1024 px of each frame was cropped and passed through the labeler networks to generate the fluorescent images. A background subtraction was then performed using a mean filter to normalise the intensity levels, before thresholding to obtain a binary mask. The number of non-zero pixels is counted to obtain an index acting as proxy to the total number of cells. Figure 5.16 shows a strong correlation between the outputs of the two systems. This is corroborated in Table 5.5. Note that this represents roughly two orders of magnitude more



(a)



(b)

Figure 5.15: Population curves for manually-annotated test set A (a) and test set B (b), compared with detection system outputs.

data than the datasets A and B visualised in Figure 5.15.

Well	Alive	Dead
A01	0.9751	0.8935
A02	0.9851	0.8657
A05	0.9888	0.8230
A06	0.9734	0.8840

Table 5.5: Correlations between object detection and fluorescence prediction time series for alive and dead Raji cells in four experimental replicates.

We therefore see that the approaches are probably on par. The advantage of the labeling approach is that it does not require a handcrafted workflow to generate the ground truth from the experimental data: the ground truth is given by the image pairs themselves. In contrast, the object detection approach requires an ad hoc procedure to derive the cell class label from the fluorescence images. This workflow is particularly easy and can be designed with very basic image analysis methods in the present setting, but for other markers and imaging conditions it would require some re-tuning of parameters. On the upside, the object detection approach directly provides a cell by cell quantification, a level of information which is not directly available in the fluorescence predictions. Indeed, it is less satisfying from the perspective of a biologist to just have a general index that would not allow the assignment of interpretable phenotypic classes to each cell.

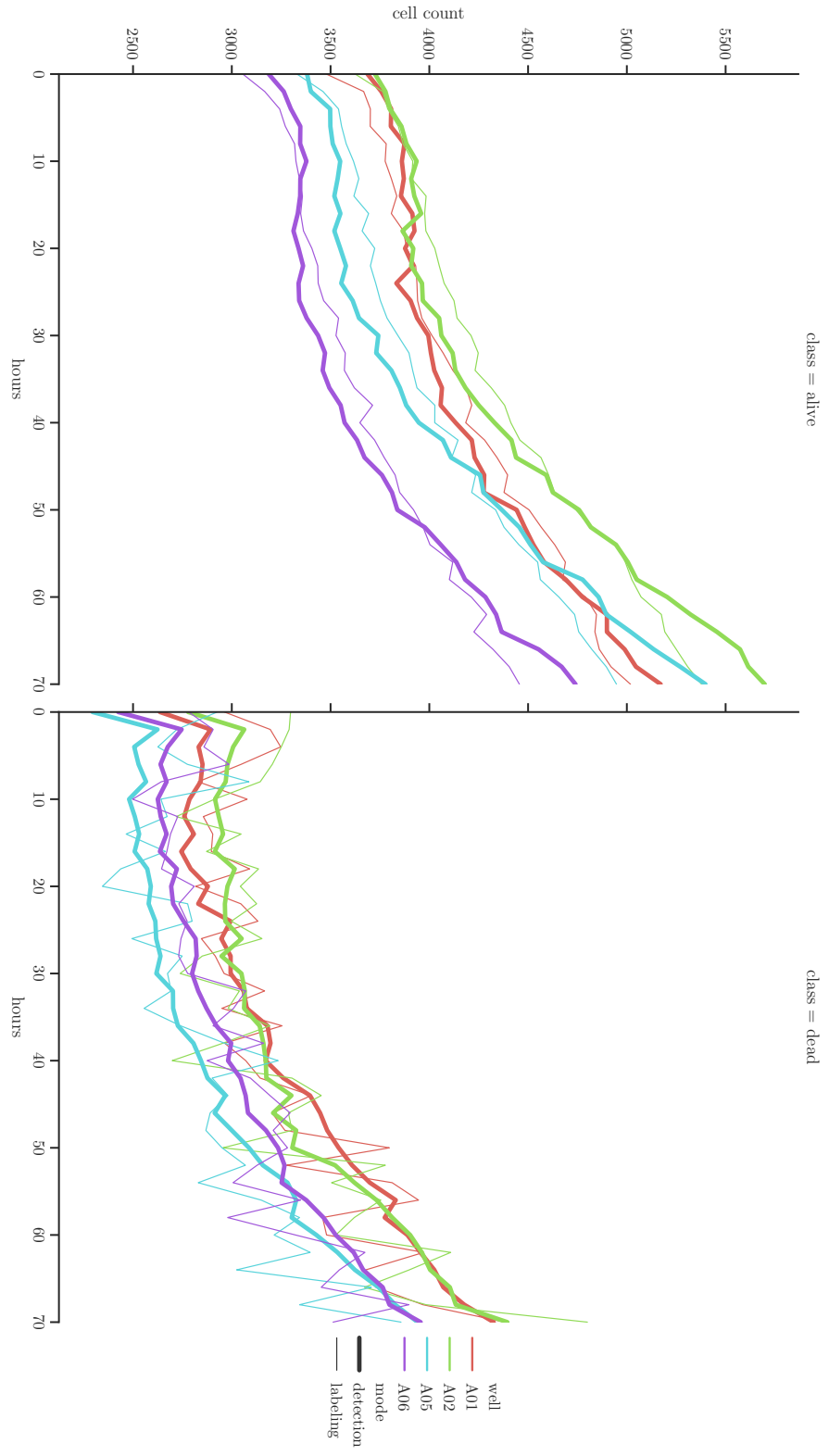


Figure 5.16: Comparing cell quantification strategies by accumulation of cell types over time in four well replicates, aggregating over fields of view. The labeling series are normalised to have the same mean as the detection series.

Chapter 6

Deep style transfer for synthesis of images of CAR-T cell populations

Summary: *The previous chapter illustrated the limitations of an automatically-generated training set for phenotyping cells in CAR-T experiments, whose accuracy degrades over time with fluorescent quenching. This final chapter formalises and tests novel ideas for circumventing these limitations. An approach based in style transfer is proposed for synthesising cell line populations according to user-defined content by leveraging unsupervised generative models. This succeeds in generating realistic phase contrast images with a complete ground truth sufficient to train a state-of-the-art object detection system.*

Résumé: *Le chapitre précédent a illustré les limites des données d'entraînement générées automatiquement pour le phénotypage des cellules dans les essais CAR-T, dont la précision se dégrade avec le temps. Ce dernier chapitre formalise et teste de nouvelles idées pour contourner ces limites. Une approche basée sur le transfert de style est proposée pour synthétiser des populations de lignées cellulaires selon une spécification de contenu définie par l'utilisateur en s'appuyant sur des modèles génératifs non supervisés. Cela permet de générer des images réalistes à contraste de phase avec une annotation complète suffisante pour entraîner un système de détection d'objets de pointe.*

6.1 Overview

Chapter 5 compared approaches for detecting and classifying cell types from phase contrast images: directly using an object detection system; and indirectly using a fluorescence predictor (which doubles as a visualisation tool). Each approach had its shortcomings: highly clustered cells prevent the harvesting of training data for object detection, and fluorescent quenching in later experimental frames caused problems for both. In this chapter we explore an entirely different approach that aims to circumvent these problems. This approach is based on generative models.

The desired end result is an endless supply of perfectly annotated synthetic images, on which we can train a state-of-the-art object detector. The intention is that such a detector will transfer seamlessly to the true data and outperform the Chapter 5 approaches. In that chapter, a custom object detection system was designed to be trainable without a full annotation of the data. With a full annotation, the options for detector are broadened considerably.

6.2 Feasibility study: synthesising cell crops

To assess the feasibility of using generative models for cell population generation, let us attack a simpler problem: generating crops of individual cells. There are grounds for optimism, given the low target resolution, coupled with the abundance of training data. Osokin et al. [2017] earlier achieved impressive results synthesising fluorescence readouts of cell crops. Using the automatic cropping strategy detailed in Chapter 5, a training set of 42241 24×24 px cell crops were assembled from four days of frames from the well A01, field 1 (a Raji-only experiment). The natural imbalance of cell states produced a class imbalance of 32507 living to 9734 dead cells. Samples from the training set are given in Figure 6.1.



Figure 6.1: Example cell-centered 24×24 px crops from the ground truth training set.

Fully-connected and deep convolutional GANs were trained, along with conditional versions of each (see Section 2.6 for an overview of GANs and see Appendix B.1, B.2 for network specifications). In training the conditional models, the number of living and dead Raji cells sampled were balanced in each

mini-batch. Otherwise, the training settings were held constant across the experiments: 50 epochs; batch sizes of 120 samples; 100-dimensional Gaussian noise input. All models were trained with the Adam optimiser (Kingma and Ba [2014]) with maximum learning rate $2e^4$, $(\beta_0, \beta_1) = (0.5, 0.999)$, and learning rate decay $1e^{-6}$, with the Jensen-Shannon objective function.

6.2.1 Generative adversarial networks

The simplest practicable variant of GANs incorporates D and G as fully-connected networks. Such models, show (perhaps cherry-picked) success in generating MNIST digits in Goodfellow et al. [2014b], which likewise are of low resolution. Figure 6.2 shows, however, underwhelming results sampled from the trained fully-connected generator, with only rudimentary intensity levels and cellular structure of the ground truth captured. Furthermore, difficulties with mode collapse were encountered more frequently than when dealing with the convolutional variants below. This lends support to the presumption that there is little to recommend fully-connected GANs for computer vision applications.

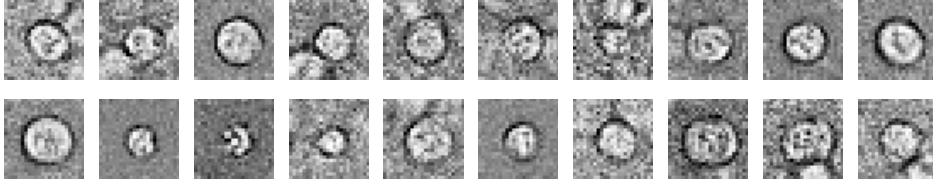


Figure 6.2: Sample 24×24 px crops from a fully-connected GAN.

6.2.2 Deep convolutional GANs

Deep convolutional GANs (DCGANs) represent a vast improvement over the MLP-based GANs trained above. Figure 6.3 provides samples from the convolutional generator that are qualitatively close to ground truth samples¹, with recognisable cellular characteristics, as well as convincing neighbouring cells in the crop periphery.

To establish the uniqueness of these samples, a k-nearest neighbour classifier (k-NN) ($k = 9$) was trained to retrieve the closest samples (in Euclidean distance) within the training data. Such a test is important, as one does not wish the network merely to memorise ground truth samples. A set of comparisons are made in Figure 6.4, illustrating the absence of sample

¹Indeed, when swapped with true samples, they succeeded in fooling a room full of experts as to their authenticity.

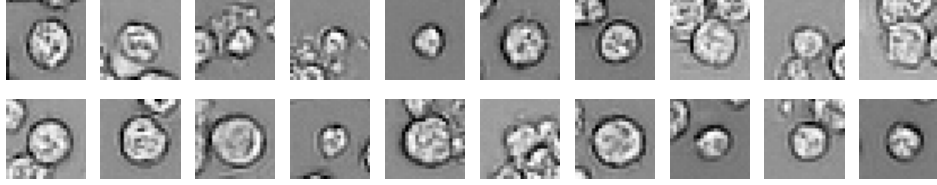


Figure 6.3: Sample 24×24 px crops from a DCGAN. One may notice the generator has learned to synthesise convincing peripheral cells.

“memorisation”. Note, however, that this is neither an altogether reliable nor conclusive test: a Euclidean k-NN could easily be fooled by a “parrot” GAN that simply emitted memorised samples with intensities scaled, but this does not appear to be the case here.

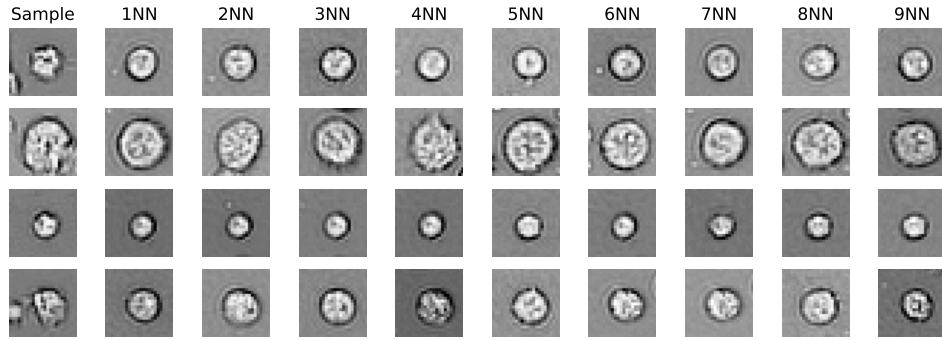


Figure 6.4: Comparison of cells generated with DCGAN (left-most column) against 9 nearest neighbours from training set.

6.2.3 Conditional GANs

Fully-connected and deep convolutional variants of CGANs are tried, where conditioning is made on class information for living and dead Raji cells. This is encoded as a two-dimensional one-hot vector. Note that this represents a first attempt to control the data generation process. As with the previous results with GANs, the results with fully-connected networks are underwhelming, and they are omitted from the discussion. Figure 6.5, however, displays samples from a conditional DCGAN, which are qualitatively both as convincing as our DCGAN samples, and also obey the conditional class signal, enabling on-demand generation of 10 samples apiece for the two classes.

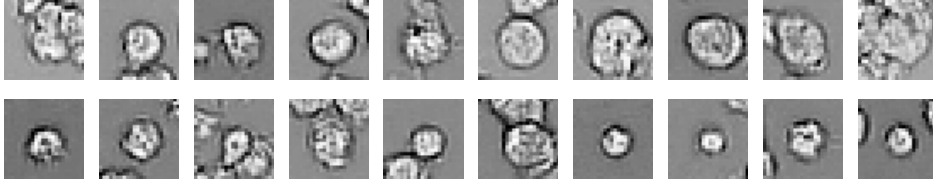


Figure 6.5: Sample 24×24 px crops from a conditional DCGAN. The top row are crops produced by conditioning for living Raji cells; the bottom are crops produced by conditioning for dead Raji cells.

6.3 Style transfer for simulating populations of Raji cells

Gatys et al. [2016] proposed a “neural algorithm of artistic style”, an algorithm to perform *style transfer* using neural networks, the synthesis of an image combining textures of one style image and another content image. Note that this involved neither the architecting nor the training of a new purpose-built neural network, rather the clever utilisation of (any) pre-trained CNN. The work combined two related ideas: content reconstruction and texture reconstruction using CNNs. **Content reconstruction** with deep networks has been explored since the very first wave of CNNs assumed their primacy. Zeiler and Fergus [2014] showed how their earlier “deconvolutional network” (Zeiler et al. [2010]) could be used to reconstruct input images from the activations inside CNNs². Simonyan et al. [2013] proposed a gradient-based method for visualising learned features of a pre-trained network. Thus, a randomly initialised image was transformed into a structured image by incrementally adding its (regularised) gradient with respect to a chosen neuron from the network, so as to maximise the activation of that neuron. The deepest neurons, those corresponding to semantic object classes, led to the synthesis of recognisable yet ethereal impressions of those class objects. The authors argued their technique roughly generalised that of Zeiler and Fergus [2014]. Mahendran and Vedaldi [2015] took the idea of using neuron-to-image gradients to full image construction. This is the idea that later inspired neural style transfer: a target image is fed into a pre-trained CNN and the tensor of its activations recorded; a randomly initialised synthesis image is then fed into the network, producing its own activation; the synthesis image is updated using gradient descent so as to minimise the mean square error of its activation to that of the target. The outcome may be a near-perfect reconstruction of the image. **Texture re-**

²The work is best remembered for proposing “ZF-Net”, a very modest modification of AlexNet, as well as using pre-trained CNNs as feature extractors for other models.). In so doing, the paper did a lot to demystify the feature hierarchies learned by deep CNNs.

construction with CNNs was achieved by the same authors of style transfer in Gatys et al. [2015], in a prelude to that later work. The principle is the same as Mahendran and Vedaldi [2015], yet rather than to reconstruct an image from its activations directly, one reconstructs for statistical properties derived from those features, namely their Gram matrix. The principle is that two images sharing such features are identical from a textural perspective. The authors write,

Textures are per definition stationary, so a texture model needs to be agnostic to spatial information. A summary statistic that discards the spatial information in the feature maps is given by the correlations between the responses of different features. These feature correlations are, up to a constant of proportionality, given by the Gram matrix.

In Gatys et al. [2016], the ideas of style and content reconstruction were combined. Thus, one elects style $\mathbf{s}$ and content $\mathbf{c}$ reference images, and simultaneously reconstructs gradient-based reconstructions of a randomly initialised image $\mathbf{x}$,

$$\mathcal{L}(\mathbf{s}, \mathbf{c}, \mathbf{x}) = \alpha \cdot \mathcal{L}_{\text{content}}(\mathbf{c}, \mathbf{x}) + \beta \cdot \mathcal{L}_{\text{style}}(\mathbf{s}, \mathbf{x}), \quad (6.1)$$

for tunable weights α and β and where the content loss,

$$\mathcal{L}_{\text{content}}(\mathbf{c}, \mathbf{x}) = \frac{1}{2} \|F^{(c)}(\mathbf{c}) - F^{(c)}(\mathbf{x})\|_F^2, \quad (6.2)$$

for the vectorised c th layer (chosen by the programmer) of the pretrained network $F^{(c)} \in \mathbb{R}^{(N_c M_c \times K_c)}$ with N_c , M_c , K_c its dimensions, and the style loss,

$$\mathcal{L}_{\text{style}}(\mathbf{s}, \mathbf{x}) = \sum_{s \in S} \frac{1}{4N_s^2 M_s^2} \|G^{(s)}(\mathbf{c}) - G^{(s)}(\mathbf{x})\|_F^2, \quad (6.3)$$

where $G^{(s)} = F^{(s)}(F^{(s)})^T \in \mathbb{R}^{K \times K}$ is the Gram matrix of the matrix of vectorised feature maps $F^{(s)}$ of layer s . Note that the content reconstruction is usually performed on a single, deeper layer of the pretrained CNN, which is more invariant to style, while the style reconstruction can be performed at several shallower layers at once, to capture style at different scales. The result is a fusion of content and style in a synthesised image. An example is given in Figure 6.6.

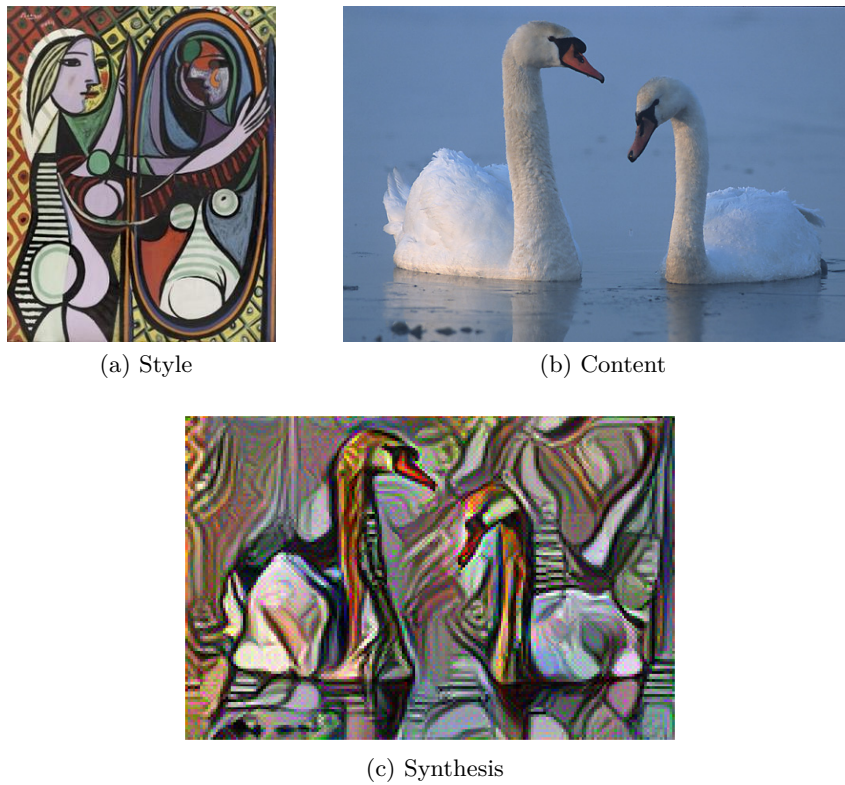


Figure 6.6: A neural algorithm of artistic style uses a pretrained CNN to combine properties from a style source image (a) and a content source image (b) to synthesise a new image combining their characteristics (c). Example taken from <https://github.com/jcboyd/vgg-fun>

6.3.1 Conditional dilation for tessellating cell contours

Our objective is an application of style transfer to synthesise images of cells in phase contrast images from our CAR-T dataset. This is motivated from the shortcomings of the automatic training set extraction presented in Chapter 5, which provides an incomplete annotation of the data, and fails, crucially, in scenarios where cells cluster profusely, as in the case of strong proliferation. It was moreover found that this translated into the degradation of performance in time for both the fluorescence predictor and object detector. In principle, a perfectly annotated ground truth could be constructed using a style transfer procedure:

1. specify a desired content image by “drawing” content
2. manually crop a region of a phase contrast image with similar content
3. run the style transfer algorithm on the above

Hypothetically, the outcome is a realistic-looking image of cells for which one has complete knowledge of its content, including cell position and size, thus the majority of information required for a fully-annotated object detection dataset (let us set aside the issue of class information for now), and indeed relates to the concept of data simulation. The idea of using style transfer for data augmentation has been shown to work in for example Zheng et al. [2019].

Given the simple geometric structure of cells, a good first approximation to a drawn cell is a circle. Therefore, in a space image plane X , the mask of an *isolated* cell k can be modeled as the set of points within distance r_k to a non-empty set or “site” P_k ,

$$R_k = \{x \in X | d(x, P_k) \leq r_k\}, \quad (6.4)$$

where the distance $d(x, P_k) = \inf\{d(x, p) | p \in P_k\}$, that is, the shortest distance to any point in the site. When P_k is a singleton set, the region becomes a circle. Otherwise, one may define the site P_k in any way desired (for example, a line or eroded mask) to model random, anisotropic variations in cell shape. The radius,

$$r_k \sim \mathcal{N}(\mu_i, \sigma_i), \quad (6.5)$$

can be sampled from an estimation of the radius distribution for class i . Once again, let us avoid the complication of prescribing classes to cells for the present.

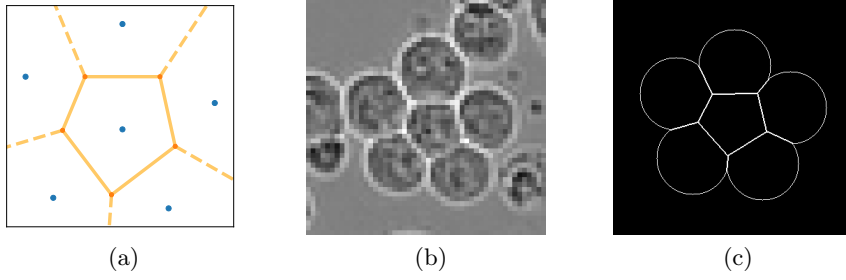


Figure 6.7: Example of Voronoi tessellation with six sites (blue points) and corresponding regions (yellow lines) (generated in `scipy` (Virtanen et al. [2020]) (a). Neighbouring cells exhibit a tessellating effect at the border (b). A conditional dilation algorithm can model this bordering effect when it comes to synthesising content images for style transfer.

Though isolated cells are highly circular, closely neighbouring cells manifest a shared, flattened contour (see Figure 6.7b). Previously, Bock et al. [2010] have modeled the border forces of neighbouring cells as Voronoi-tessellating circles. A Voronoi tessellation can be defined as K regions of the form,

$$S_k = \{x \in X | d(x, P_k) \leq d(x, P_j) \forall j \neq k\}. \quad (6.6)$$

The K regions constitute a partition of the space X , where points are assigned to the nearest site. An example is shown in Figure 6.7a where a Euclidean distance is invoked for a set of six singleton sites. The borders between regions are lines perpendicular to the line joining the site centers. It is therefore proposed to model cells as the intersection of 6.4 and 6.6,

$$T_k = \{x \in X | d(x, P_k) \leq r_k \wedge d(x, P_k) \leq d(x, P_j) \forall j \neq k\}. \quad (6.7)$$

To compute the T_k , an algorithm for *conditional dilation* is designed and shown in Algorithm 1. The principle is to select the sites (cell centres) and to grow outwards until either a maximum distance from the source is reached, or an object border is encountered. This resembles an implementation of the watershed algorithm, albeit with all seeds flooded from the same height (implemented as a priority of 0). The dilation of sites is scheduled according to a priority queue, for which priority is assigned as the distance from the source³. Thus, dilation propagates from the sites synchronously. The priority queue is implemented as a *heap*, as it can push and pop in $\mathcal{O}(\log N)$ time.

³The `DIST` function for calculating minimum distances to the site sets implements a lookup table based on a pre-computed distance map.

Algorithm 1 Dilates labeled image SiteImage until distance from source exceeds Radii.

```

1: procedure CONDITIONALDILATION(SiteImage, Radii)
2:   PriorityQueue  $\leftarrow$  HEAP()
3:   for marker in SiteImage do
4:     Elem  $\leftarrow$  HEAPELEMENT()
5:     Elem.value  $\leftarrow$  0  $\triangleright$  Top priority
6:     Elem.source  $\leftarrow$  marker  $\triangleright$  Record origin
7:     PriorityQueue.PUSH(Elem)
8:   end for
9:   while  $\neg$  PriorityQueue.ISEMPTY() do
10:    Elem  $\leftarrow$  PriorityQueue.POP()
11:    for Neighbour in Elem.GETNEIGHBOURS() do
12:      if SiteImage[Neighbour]  $>$  0 then
13:        continue  $\triangleright$  Neighbour already visited
14:      end if
15:      NewElem  $\leftarrow$  HEAPELEMENT()
16:      NewElem.value  $\leftarrow$  DIST(Neighbour, Elem.source)
17:      if NewElem.value  $>$  Radii[Elem.source] then
18:        continue  $\triangleright$  Cell border reached
19:      end if
20:      NewElem.source  $\leftarrow$  Elem.source
21:      PriorityQueue.PUSH(NewElem)
22:      SiteImage[Neighbour] = SiteImage[Elem.source]
23:    end for
24:  end while
25:  return SiteImage
26: end procedure

```

Applying this for the purpose of drawing cell population content images, the space X becomes an image “canvas”, that is, a two-dimensional array of numbers initialised to zero to indicate background. Sites are allocated as connected components (perhaps just points) of non-zero, foreground pixels. The canvas with its allocated sites and corresponding dilation radii are inputs to the CONDITIONALDILATION algorithm. This works well in practice as long as the sites are carefully selected. For the time being, sites are chosen uniformly at random (without replacement)⁴, however sites must be chosen outside all other regions. Provided the canvas is large enough relative to the cells and the cell number, there is a large space of feasible allocations of sites. For efficiency, a record is kept of the vacant coordinates in the drawing canvas, from which is subtracted the coordinates of the anticipated region of each allocated site on-the-fly. If sites are naively allocated in arbitrary order, there is the risk of the scenario in which a larger cell is allocated in the vicinity of a smaller one, placing the center of the smaller cell within the region of the larger. A neat solution is to simply allocate the sites in descending order of radius. This procedure is described in Algorithm 2.

Algorithm 2 Allocates sites for input to CONDITIONALDILATION algorithm. VacantCoords are a list of (x, y) coordinates available for allocation (initially a full image), and Radii are the presampled radii for drawing cells.

```

1: procedure ALLOCATESITES(VacantCoords, Radii)
2:   Sites  $\leftarrow$  []
3:   Radii  $\leftarrow$  SORT(Radii, descending=TRUE)
4:   for radius in radii do
5:     Site  $\leftarrow$  SAMPLE(VacantCoords)
6:     Sites  $\leftarrow$  Sites  $\cup$  Site
7:     Region  $\leftarrow$  {Coord  $\in$  VacantCoords | ||Coord - Site||2 < radius}
8:     VacantCoords  $\leftarrow$  VacantCoords  $\setminus$  Region
9:   end for
10:  return Sites
11: end procedure

```

The outcome of the CONDITIONALDILATION algorithm is a content image in the form of a labeled mask. While masks are a natural choice for cell content markers, an alternative that is found to be more useful is the cell contour defined as,

$$C_k = T_k - T_k \oplus B_D, \quad (6.8)$$

where $\oplus$ represents a morphological dilation or maximum filter, T_k is the

⁴One could imagine using a dynamically-updated distribution to simulate cell clustering.

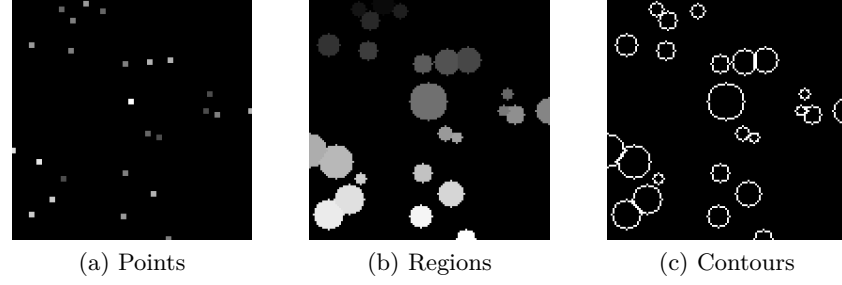


Figure 6.8: An example of generating a cell content image by first allocating sites with `ALLOCATESITES` (a) (site intensity represents radius, dilated for visibility), with radii drawn from Equation 6.5; running the `CONDITIONALDILATION` algorithm to obtain labeled regions as in Equation 6.7 (b); and extracting the contours with Equation 6.8 (c).

object mask, and B_D is a (for example, disk-shaped) structuring element. The full procedure for generating content images is illustrated in Figure 6.8.

Results

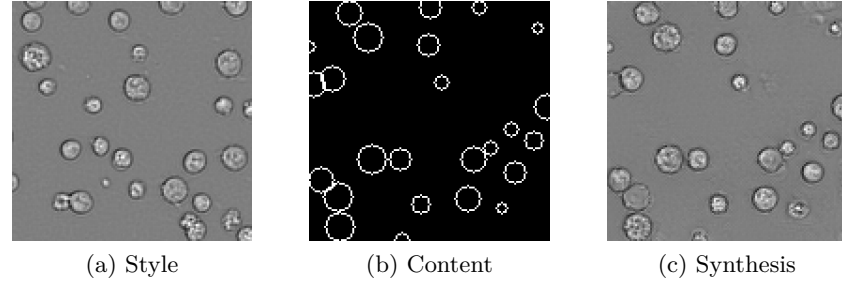


Figure 6.9: An application of a neural algorithm of artistic style to phase contrast images of CAR-T cells (a) and a style source image (b) to synthesise a new image combining their characteristics in a “simulated” image (c).

Figure 6.9 shows a first result with style transfer. The contours of cells are generated according to Equation 6.5, which can be estimated from a library of pre-cropped cells from Chapter 5. The number of cells is chosen to match the chosen style image, and cells are placed uniformly randomly on the zeroed background plane. Here we see that the transfer succeeds—the background and foreground textures have been suitably apportioned to the objects prescribed by the content image. Various artifacts exist, of course, but individual cells appear correctly rendered, and the main fault is the

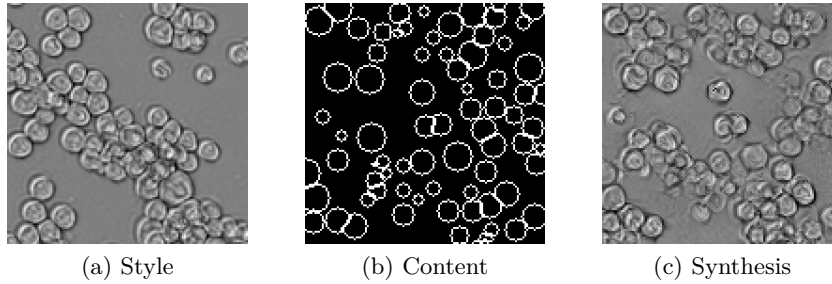


Figure 6.10: More cells, more problems. The style transfer algorithm fails when the content image does not offer adequate “scaffolding” for the target texture patterns.

global arrangement of the cell population lacking sense (recall objects have been placed uniformly randomly).

Note that the low-resolution and simple geometric structure of the cells are naturally advantageous. Of course, if more complicated objects were to be modeled, the contour simulator need likewise be more sophisticated. There are, nevertheless, limitations to this approach. We see in Figure 6.10 a larger population of cells with a greater degree of failure. In general, a tension is observed between the content specification and the chosen style crop. Note that Equation 6.3 aims to ensure the same co-occurring neural activations invoked by the style target are invoked by the synthetic image also, albeit the global arrangement is arbitrary due to stationarity of the Gram matrix. The algorithm preferences applying these textures in the vicinity of recognisable contours, which is why one can steer the synthesis reasonably well. Furthermore, it is found that the cell geometry is flexible: when the disk contours are replaced with diamonds, for example, the algorithm still performs well. Yet, it is crucial to approximately match the size and number of cells between the style and content images to obtain a good distribution of texture. Indeed, the style image appears to provide a certain “budget” that if underspent will be apportioned arbitrarily (such as to the background), and if overspent will mean some objects end up untextured.

Though as a proof of concept this approach is successful and yields various insights, the pre-trained network, with all its particularities, does not offer the necessary control for bootstrapping a robust ground truth. In addition, the approach entails the manual curation of a suitable style reference image. What is more, the generation is slow: one must backprop a large neural network (although Johnson et al. [2016] find ways of greatly improving the run time). To overcome these difficulties, in Section 6.4 let us reformulate the problem as a learning problem of unsupervised image translation.

6.4 CycleGANs for cell population synthesis

CycleGANs (Isola et al. [2017]) extend the `pix2pix` framework (see Section 5.3.1) to the *unpaired* image-to-image translation setting, that is, translating between two (necessarily similar) image domains in a fully unsupervised way. Though the underlying neural network architectures remain virtually unchanged, the framework is quite different. CycleGANs introduce an additional pair of generator and discriminator. Now, one generator learns the mapping $G : X \rightarrow Y$ while the other learns $F : Y \rightarrow X$. The respective discriminators train these generators adversarially in the usual way. However, at the same time, there are cycle consistency losses ensuring $F(G(X)) \approx X$ (forward cycle consistency) and $G(F(Y)) \approx Y$ (backward cycle consistency). The cycle consistency loss,

$$\mathcal{L}_{CYC}(G, F) = \mathbb{E}_{x \sim p_{data}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{data}(y)}[\|G(F(y)) - y\|_1], \quad (6.9)$$

that is, the expected l_1 error upon reconstructing real images via the stacked generators. Equation 6.9 is combined with the usual GAN losses for each direction of translation to give the CycleGAN objective,

$$\max_{G, F} \min_{D_X, D_Y} \mathcal{L}_{CG} = \mathcal{L}_{GAN}(D_Y, G, X, Y) + \mathcal{L}_{GAN}(D_X, F, Y, X) + \lambda \mathcal{L}_{CYC}(G, F). \quad (6.10)$$

Thus, it must be that such a consistent translation can be discovered between the domains. If the domains are similar, this works surprisingly well in practice. Note that unlike `pix2pix` (paired image translation) CycleGANs are no longer conditional in that the discriminators no longer have access to the source image. The generators are still conditioned on their input image, however, instead of the noise vector of a vanilla GAN. A successfully trained CycleGAN can learn to perform a (bidirectional) style transfer between, for example, photographs and paintings, and maps to satellite imagery. In contrast with “a neural algorithm of artistic style” (Section 6.3), however, it requires a full training corpus to learn from. At test time, the style transfer is performed in a single forward pass of the generator. Moreover, empirical results suggest the outcomes are superior (Isola et al. [2017]). This is unsurprising, as the CycleGAN is optimised to perform style transfer directly, rather than by a clever utilisation of a pretrained network. The results on paired image translation are usually not as impressive as `pix2pix`, which they assert to be an upper bound on performance. However, CycleGANs are still trained on a surprisingly small number of images, despite being very deep architectures. Even more recently, *RecycleGANs* (Bansal et al.

[2018]) achieve the astonishing feat of unpaired video-to-video retargeting. These build on the CycleGAN framework, introducing “recurrent temporal predictor” networks, and replace the cycle consistency losses, with a similarly defined “recycle loss”. Although the CAR-T dataset are indeed a form of video data, Recycle-GANs seem excessive for the problem at hand and CycleGANs should be sufficient.

6.4.1 First results with CycleGANs

CycleGANs are therefore proposed as an improvement for the style transfer algorithm in Section 6.3. The domain X is modeled as the content domain, where while training the CycleGAN, the content specifications are randomly generated according to Section 6.3.1. The domain Y is again the style domain of phase contrast images, however now the style image patches may be freely sampled at random, that is, without the manual discretion involved in Section 6.3. The idea of synthesising a training set with image-to-image translation models has some very recent precedents including with `pix2pix` for object segmentation (Hollandi et al. [2019]) and CycleGANs for segmentation (Fu et al. [2018]).

CycleGANs are therefore trained to map between synthetic cell drawings and authentic phase contrast crops. The alternating fashion of GAN training is followed: the two discriminators are trained on a single batch, then the two generators. The batch size is limited to a single image as in the related networks of Section 5.3.1, and, as in Section 6.2, the Adam optimiser is used with maximum learning rate $2e^4$ and $(\beta_0, \beta_1) = (0.5, 0.999)$. The generator and discriminator closely resemble the FNet and PatchGAN of Section 5.3.1. However, to decrease training time and avoid memory concerns, in practice the number of filters are halved in each generator and their depth is decreased. In addition, the batch normalisation layers are replaced by the instance normalisation layers advocated by Isola et al. [2017] (and first proposed by Krizhevsky et al. [2012]).

First results with this setup are given in Figure 6.11. The results come from feeding a synthesised content image to content-to-style generator G and reconstructing with the style-to-content generator F . We may deem these results to be highly satisfactory, with the system adhering closely to the specification given in our content drawings, while producing realistic textures for the cells over a broad range of cell population sizes.

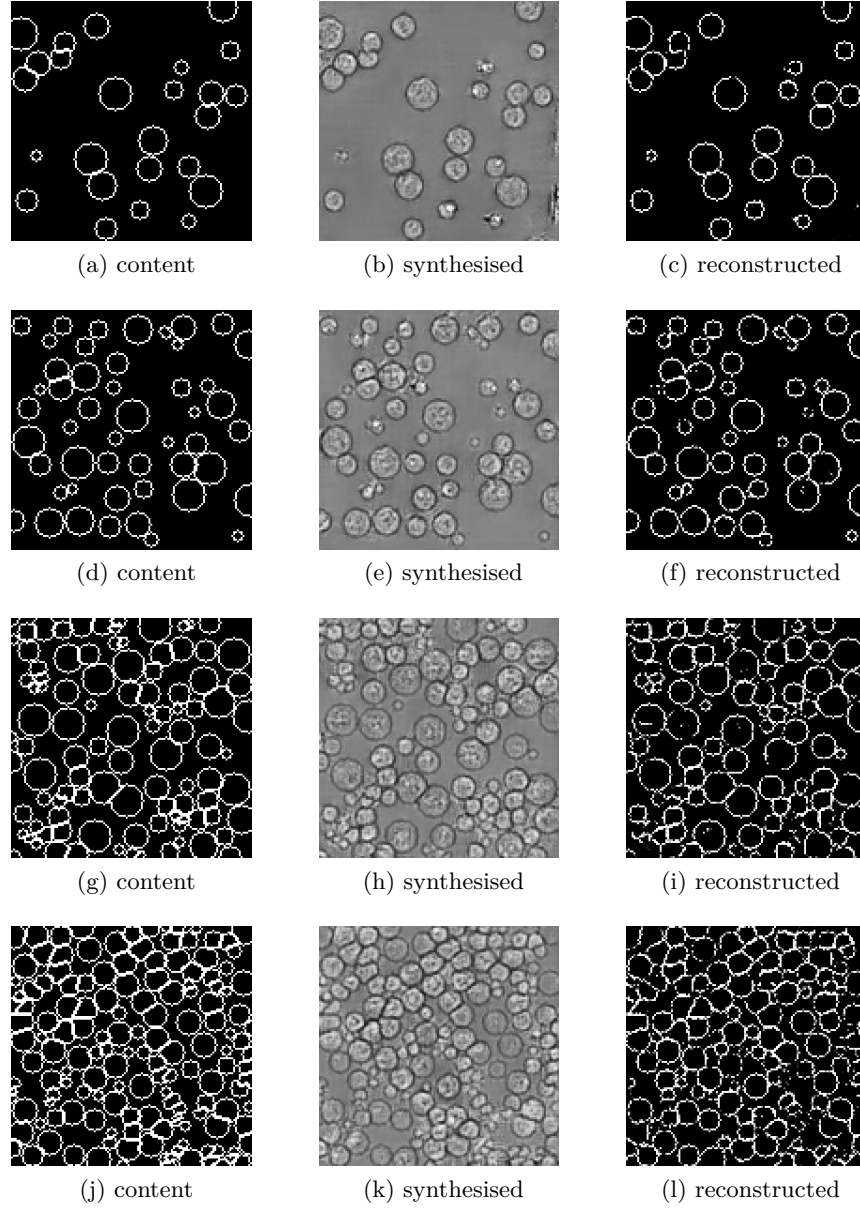


Figure 6.11: Examples of CycleGAN generated images. Columns organised with content specification input image X (left), $G(X)$ generated image (center) and reconstruction $F(G(X))$ (right). The number of cells are increased over the rows: 25, 50, 100, and 150.

6.5 Fine-tuning a state-of-the art object detection system

With the content-to-style generator from Section 6.4.1 acting as an “oracle” for perfectly-annotated training data, a state-of-the-art object detection system may be trained, such as that which remained elusive in Chapter 5. The Faster R-CNN model (Girshick [2015]) is chosen for this task. A full discussion of the RCNN series of object detectors is available in Section 2.3.1. The Faster RCNN implementation available in the PyTorch framework (Paszke et al. [2017]) is used. This has been pretrained on the COCO object detection dataset Lin et al. [2014]. A simple fine-tuning procedure is to replace the classification and bounding box regression “heads” of the network, that is, leaving the backbone and region proposal network untouched, and to continue gradient descent on the new data with a lowered learning rate. This is done, with a learning rate of 10^{-4} , training with SGD with momentum. Mini-batches are formed for training with 8 synthetic images. The (perfect) bounding box annotation (and, in principle, object class information) is determined by the drawing procedure of the synthetic image.

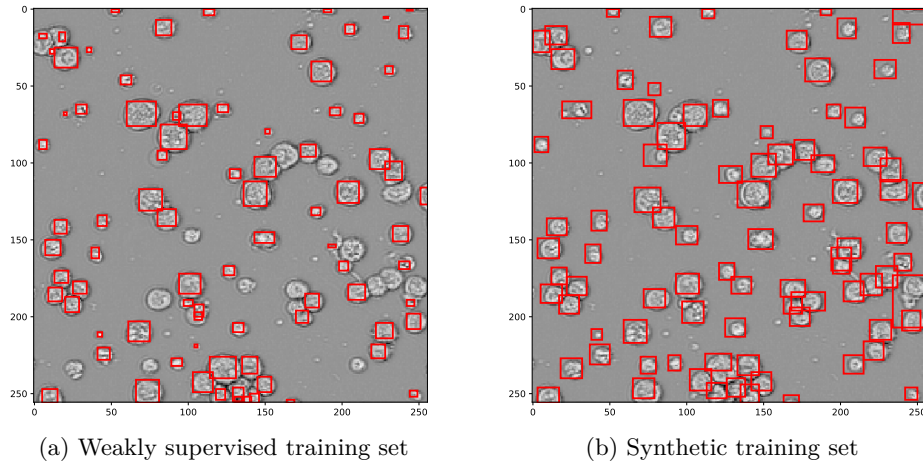


Figure 6.12: Comparison of outcomes of Faster R-CNN on a test image when a) fine-tuned on a weakly supervised dataset and b) fine-tuned on synthetic images.

The outcome of this fine-tuning is a detection system capable of detecting cells (but not classification, for now). A comparison is made between Faster R-CNN trained on the CycleGAN synthetic images against the incomplete annotations arising from the automatic pipeline proposed in Section 5.4.1. Recall, the argument for pursuing a custom detector in Chapter 5 was that such a *weakly supervised* dataset would be insufficient for training a state-of-

the-art system like Faster R-CNN. We see in Figure 6.12 evidence to support this intuition, in which are compared the outcomes of the fine-tuned R-CNN on an authentic test image, when fine-tuned first on a weakly supervised dataset, then on the CycleGAN-generated images.

6.6 Perspectives on synthesising a full object detection dataset

In Section 6.5 the viability of training a state-of-the art object detection system on fully-annotated synthetic images has been shown. The missing component is to incorporate classification into the framework. In principle, the R-CNN detector is designed to learn this task simultaneously with the object detection, however, the CycleGANs, being a fully unsupervised model have no means for injecting this information into the data-generating process. Clearly, one could use the R-CNN as a region proposal algorithm and classify regions with a separately trained classifier (such as in Section 5.4.2). Preliminary results with this approach demonstrate this is a viable approach. However, the R-CNN system should be altogether more powerful if trained on a full annotation of object class and location. One could, alternatively, attempt to incorporate the weakly annotated ground truth (as utilised in Section 6.5) into the R-CNN fine-tuning in an attempt to teach it about Raji cell classes.

In the following, however, an extension to the CycleGAN system above is conceptualised that would be theoretically capable of providing both a complete annotation of synthesised images, encompassing both object localisation and class annotations. This extension takes the form of an additional discriminator for the content-to-style generator, that, in contrast to the PatchGAN discriminator, discriminates on regions of interest. Such a *region of interest discriminator*, D_{roi} compares fake cells localised by bounding boxes inside generated images with a pre-cropped “library” of real cells. The library can include class information for the individual crops, which can be combined in the fashion of a conditional GAN. The content image would also have to encode class information in some way (for example, by intensity or a one-hot encoding). Thus, the D_{roi} discriminator could provide a mechanism to enforce class information at the resolution of specific regions of interest. For a preliminary implementation of this idea, see Appendix D.2.

6.6.1 Region of interest discrimination

Let us define an auxiliary discriminator for image-to-image GANs,

$$d_{roi} : X \times B \rightarrow \{0, 1\}, \quad (6.11)$$

for image domain X and bounding box domain B , with $d_{roi}(\mathbf{x}, b) = d(\rho(f(\mathbf{x}), b))$, for input image $\mathbf{x} \sim X$, bounding box $b \sim B$, and where we conceive of a neural network comprising of feature extraction layers f , classification layers d , and separated by RoIPool layer ρ . A RoIPool layer⁵ (Girshick [2015]) generalises the MaxPooling layer and quantises regions of interest (RoIs) into a fixed size,

$$\rho : X \times B \rightarrow \mathbf{y}, \quad (6.12)$$

for input tensor $\mathbf{x} \in \mathbb{R}^{1 \times W \times H \times C}$, set of K bounding box tuples B , and tensor output $\mathbf{y} \in \mathbb{R}^{K \times w \times h \times C}$ with quantised dimensions $w < W$ and $h < H$. To clarify, the incoming tensor $\mathbf{x}$ is transformed into a “pseudo-batch” of K tensors for the K bounding boxes of B . Akin to the PatchGAN (see Section 5.3.1), the full discriminator output averages over the individual bounding boxes,

$$D_{roi} = \frac{1}{|\mathcal{B}(\mathbf{x})|} \sum_{b \in \mathcal{B}(\mathbf{x})} d_{roi}(\mathbf{x}, b), \quad (6.13)$$

where the operator $\mathcal{B}$ returns the set of bounding boxes of image $\mathbf{x}$. Of course, in the present case, the bounding boxes are decided in advance by virtue of drawing the content images. This is trained in the loss function,

$$\max_{G, F} \min_{D_X, D_Y, D_{roi}} \mathcal{L}_{CG} + \mathcal{L}_{GAN}(D_{roi}, G, X, Y), \quad (6.14)$$

where $\mathcal{L}_{CG}$ is the CycleGAN loss function defined in Equation 6.10. The setup is illustrated in Figure 6.13. Just like the contour generation, the problem of obtaining a crop library will vary from problem to problem. In this instance, however, direct access to such a library comes from the automatic procedure described in Section 5.4.1. Some basic proofs of concept of using the RoIPool for classification and GAN discrimination are given in Appendix D.2.

⁵See also RoIAlign layer (He et al. [2017])

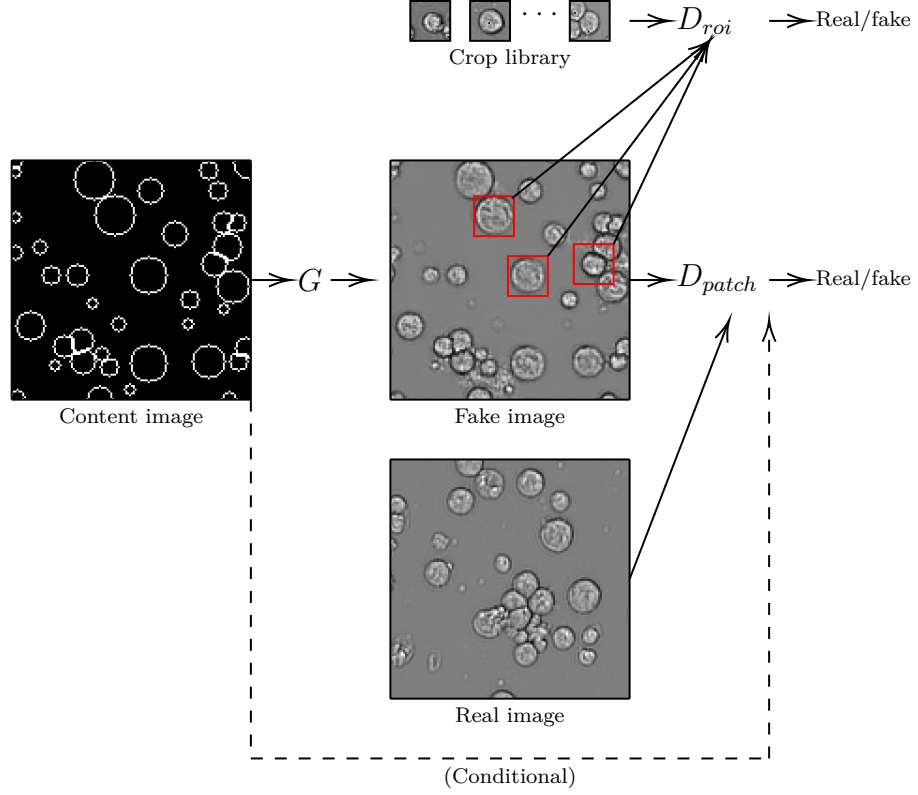


Figure 6.13: Conceptualisation of an extension to adversarial image-to-image translation networks with a region of interest discriminator D_{roi} . The D_{roi} relies on a library of real crops to compare to cells synthesised by G in the prescribed regions of the content image.

6.7 Discussion

This chapter illustrated the viability for the synthesis of images of cells based on phase contrast images of CAR-T experiments. This was shown both at the level of single cells with convolutional and conditional GANs in Section 6.2, and at the population level through applications of style transfer both classical (Section 6.3) and with CycleGANs (Section 6.4). Generating populations of cells according to a content image “blueprint” proves to be a powerful way of generating complete annotations in an entirely unsupervised way, sufficient to train a state-of-the-art object detection system (Section 6.5). The blueprint (Section 6.3.1) models various characteristics of individual cells and groups of cells in order to facilitate the style transfer. A means for incorporating class information was also conceptualised in Section 6.6 as a full realisation of the concept of unsupervised training data creation for the phenotyping of CAR-T experiments.

At least two aspects of the current pipeline can be improved. The first is the blueprint content image, which, as of yet, does not account for the tendency of Raji cells to cluster. As mentioned previously, this is a matter of changing the distribution of cell placement in the blueprint from uniform random placement. The second is to finalise the inclusion of class information in the training set synthesis. A concept was proposed in Section 6.6, with a number of alternatives also outlined.

Chapter 7

Conclusions

Computational phenotyping is an ascendant methodology for the study of phenotypes. This thesis has explored a range of solutions for two problems in computational phenotyping. Despite the manifest overlap, these problems ultimately sit on different axes of development. Part **I** explored ways of performing a joint study of multiple disease subtypes. The molecular heterogeneity of multiple cell lines led ultimately to a computational problem that was resolved with domain adaptation. Part **II** compared approaches to fluorescence-free phenotyping by leveraging a dataset pairing phase contrast and fluorescence microscopy images. Part **I** can be considered a stepping stone for computational phenotyping in the long road to precision medicine, and as such remains an open-ended study. On the other hand, Part **II** belongs to a recent trend in biomedical imaging, for which a more immediate outcome is likely, namely, that deep learning may soon revolutionise the microscopy tool set for conducting biological experiments. The following provides first a summary of the chapters and then speculation about the future of high content screening.

7.1 Chapter summaries

Chapter **3** introduced the high content analysis pipeline and applied it to drug and wild type screen datasets for multiple triple-negative breast cancer (TNBC) cell lines. A series of use cases were presented spanning each of the univariate-, multivariate-, and machine learning-based analyses formalised in Section **1.1.3**. While these were included as motivating examples, we saw in particular an analysis of double strand breaks that highlighted a systematic bias in a standard piece of analysis using causal inference formalisms. Elsewhere we observed a first instance of the differential effects of drugs on cell lines in the form of viability, which were additionally categorised by

drug mechanism of action. We also saw how TNBC cell lines could be analysed in unison, and explored their phenotypic differences across two axes of measurement: firstly at the population level, as different molecular subtypes manifested distinct patterns of cell cycle and aberrant morphologies; then at the cellular level, with the fundamental wild type morphologies easily distinguishable by a simple classifier. The sum of these insights was carried forward to the following chapter.

Chapter 4 explored in great detail the construction of phenotypic profiles for characterising drug effects. It reviewed and tested the most relevant existing approaches in a comparative study of mechanism of action (MOA) prediction. It then extended the methodology to a multi-cell-line setting, where it was shown that an autoencoder based on unsupervised domain adaptation could succeed in building domain-invariant features that outperform baseline models in MOA prediction. We additionally found evidence to suggest that pooled multi-cell-line data provides more powerful representations for predicting the MOA of hold-out drugs than single-cell line datasets of the same size. In contrast with previous studies, this use case for phenotypic profiling was framed without tailoring the prediction problem to those MOAs with the most striking visual effects, yet confirmed MOA classification could be performed at well above a rate of random chance. This dataset has since been released in full to the scientific community. In sum, this chapter validated a novel solution for managing the heterogeneity of multi-cell-line screens using deep and adversarial learning.

Chapter 5 compared two deep learning methodologies for quantifying CART and Raji cell line cells in time lapse movies. With a setup providing a paired image dataset of transmitted light and fluorescent channels, the key barrier for training deep phenotypers is toppled. Each methodology explored a different use of fluorescence for automatically training deep neural networks. In doing so, the trained models progressed towards supplanting fluorescence altogether. The first approach, based on image-to-image translation networks, including a conditional GAN, showed excellent results in the fluorescent labeling of cell populations for mCherry and GFP markers. Though the path to full quantification of cells remains uncertain, this already works well as a visualisation tool. The second showed how a training set for object detection could be built, and that this is a viable alternative to fluorescence labeling, that moreover gets to the heart of the problem of counting cell phenotypes in time series. In a final result, we were able to reconcile the two methodologies and demonstrate a high degree of correlation across a set of experimental replicates.

Chapter 6 sought a way to transcend the limitations of automatic ground truth assemblage encountered in Chapter 5. Here it was shown that generative adversarial networks produce excellent results in the synthesis of indi-

vidual cells, including when conditioned on a class indicator variable. This inspired the synthesis of full populations of cells, formulated as a problem of style transfer, where style was grafted from a curated ground truth image to a chosen content image. The content image was constructed to emulate the geometric characteristics of cells and cell populations. A feasibility study in classical style transfer demonstrated the validity of this concept. This was superseded by a learning approach to unpaired image-to-image translation, the CycleGAN, which conditioned one of its generators on the content image. Using this generator as an oracle for training data, a state-of-the-art object detection system was trained, which was shown to outperform a baseline of weakly annotated ground truth data. In a final section, an extension to the CycleGAN was conceptualised, so as to be able to inject cell class information in the content specification as well.

7.2 The future of high content screening

The future of high content screening lies plausibly in the data-driven domain of deep learning, to which so many areas of computer vision have yielded in recent years. After decades spent dormant, the power of deep learning was finally unleashed by large, annotated datasets of natural images like ImageNet (Russakovsky et al. [2015]). In high content screening, the enthusiasm generated by the stunning results provided by deep learning were attenuated by two factors: first, the results obtained by other machine learning methods were already very good, and often sufficient to answer the biological questions studied. Second, the amount of annotations to provide was prohibitory.

7.2.1 Overcoming massive image annotation

Most chapters in this thesis were about making intelligent use of a relatively limited data supply: first by domain adaptation in Chapter 4; then by leveraging fluorescence as annotation in 5; and finally augmenting the available data with style transfer and generative models in Chapter 6. In more general terms, there are several ways to overcoming the bottleneck imposed by annotation:

- In image segmentation, the size of annotated data sets required to train deep neural networks is not excruciatingly large. If the imaging conditions are coherent (which is normally the case in HCS), a few fully annotated images are actually often sufficient (Ronneberger et al. [2015]). Secondly, for common segmentation tasks, such as nuclear segmentation, the community felt that it would be worth going

to the trouble of annotating large image datasets, as was the case for the “Data Science Bowl 2018”¹, with 37333 manually annotated nuclei in 841 images of different modalities. In this case, the objective was to train a general neural network capable of segmenting nuclei irrespective of the imaging modality. This sparked the development of a number of deep networks that are indeed useful for nuclei segmentation (Hollandi et al. [2019]). However, it must be noted that this large community effort is so far unique for segmentation problems in light microscopy, which reflects the central importance of nuclear segmentation. It may be summarised that deep learning has become the state of the art in bioimage segmentation and has largely replaced other techniques.

- An invaluable discovery in deep learning was that upon training a deep classifier once, it could be redeployed to exceed the state-of-the-art of yesteryear on virtually any vision problem. This is the power of transfer learning and fine-tuning. High content screens become different very quickly, however. For one, the number and kind of fluorescent channels varies from screen to screen, as they are chosen for different sub-cellular regions of interest. As a result, intensity colocalisation across fluorescent channels generally have a very different meaning from screen to screen. Furthermore, this fine-tuning approach has been criticised as being badly adapted to the specific properties of biomedical images and heavily over-parametrised (Raghu et al. [2019]). This is due to fact that the relevant information for classifying biomedical images is often highly localised, as is the case for microcalcifications in mammogram classification (see, for example, Wang and Yang [2018]).
- Another strategy is for generating large annotated datasets for cell classification, namely the classification of subcellular protein localisation patterns, is crowd sourcing and gamification (Sullivan et al. [2018]). While this strategy has proven successful, it is also limited to large and prestigious projects with generous funding and is therefore not good solution for routine screens.
- Yet another technique that has been used specifically in screening is weak supervision, where instead of training classifiers for single cell phenotypes, representations are learned by predicting drug identity. Researchers such as Kraus et al. [2016], Kandaswamy et al. [2016], Godinez et al. [2017], and Sommer et al. [2017] have shown how the conventional pipeline can be partially or fully subsumed to a neural network optimised end-to-end using this approach. This is an interesting approach with high potential in drug screening, because it efficiently removes the burden of massive annotation. However, their im-

¹<https://www.kaggle.com/c/data-science-bowl-2018/overview/about>

pact has so far been rather superficial, primarily improving MOA prediction accuracy on a benchmark dataset by a few percentage points. Furthermore, deep learning ought to allow us to somehow transcend the conventional pipeline altogether. A further problem is that one no longer has a strong interpretability of intermediate steps, as it does not involve the classification of individual cells into biologically meaningful classes.

7.2.2 The renaissance of label-free microscopy

The use of deep learning is not limited to improving classification or segmentation accuracies. The advent of image-to-image translation models has opened new avenues for the use of label-free microscopy. Fluorescence microscopy has many significant advantages, as one can specifically highlight particular biomolecules and subcellular structures. It turns out, however, that many subcellular structures leave a fingerprint in completely non-invasive label-free microscopy techniques, such as phase contrast or bright-field microscopy. Whereas several years ago it was still very challenging to quantitatively analyse these label-free microscopy datasets, the tools to do so are now available.

It is therefore possible that we will witness a renaissance of non-invasive label-free microscopy techniques in high content screening, where fluorescence microscopy is used for: (i) *calibration*, that is for the training of neural networks to reconstruct the fluorescence microscopy images from label-free microscopy or else to recognise marked structures; and (ii) *visualisation* of the biomolecules that do not leave a fingerprint in the images. While fluorescence microscopy will remain an indispensable tool for studying cells, the augmented label-free microscopy may become a means of obtaining “more from less” from transmitted light microscopies such as phase contrast. Chapter 5 in particular showed how seamlessly deep convolutional networks can be assimilated into the screening process as fluorescent predictors or cell classifiers trained by cross-modality.

These approaches may turn out to be controversial. Is it still science if we predict measurements instead of taking them? How valid are the conclusions that can be drawn from predicted data? To what extent would a different training set influence the final conclusions? One toolset that can help in overcoming these justified criticisms is a proper statistical treatment of predicted measurements, as recently proposed in [Whitehill and Ramakrishnan \[2018\]](#). Indeed, if statistical tests are performed on experimental conditions where one actually works on predictions rather than direct measurements, one needs ensure also that the uncertainty of the prediction into account. Here there are many methodological developments still to be made.

7.2.3 Towards toxicogenetics

Besides the technical improvements and extensions that are brought by new learning techniques, another important evolution concerns the biological models (in the case of this thesis, the cell lines). Indeed, the screening of entire cell line panels has created the opportunity to develop predictive models that would allow for predicting the phenotype resulting from a drug treatment on a cell line given the chemical features describing the drug, alongside genomic and transcriptomic features describing the cell line. Such datasets mimic a precision medicine use case ([Eduati et al. \[2015\]](#), [Costello et al. \[2014\]](#)), where we wish to predict a targeted treatment instead of having a one-size-fits-all solution, which might work in a statistical sense at the population level, but not necessarily at the level of an individual patient. Despite this being a promising strategy, and in spite of the efforts of numerous teams working in the field, the success has been relatively modest so far. One of the reasons for this is certainly the representation of a complex drug effect by a single number (IC50, as in [Eduati et al. \[2015\]](#)). The approach developed in [Chapter 4](#) would allow for the analysis of such screens, and in particular for making meaningful comparisons of morphological phenotypes across cell lines with different morphological baselines. This is a cornerstone of the most powerful algorithms in the field, such as kernel methods in multi-task setting ([Playe et al. \[2018\]](#)).

Appendix A

Supplementary figures

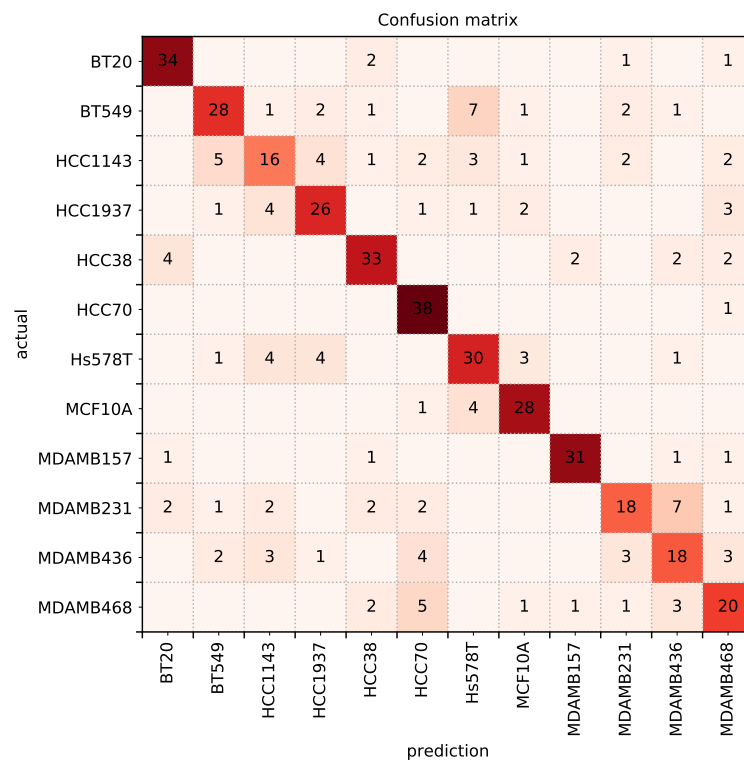
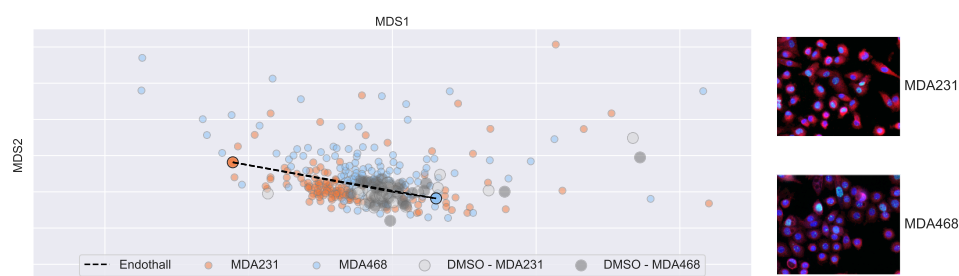


Figure A.1: Confusion matrix for a random forest classifier classifying cells from 12 TNBC cell lines.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	DMSO	Calpain Inhibitor II	PD-880593	Actinom D	Tyrosinase 46	Estetrol	AG-480 9,10-Phenanthreneq uione	Gaboxate mesylate	HA-1004	Tazobactam	5-Iodobutadiol		Erostatin analog		Tereic acid	DMSO
2	DMSO		Carbamic acid		Fenvalerate								Quercetin dihydrate		Triclorine	DMSO
3	DMSO	Calpeptin	U-0128	CL-82198	Tyrosinase 47 Tyrosinase 8	Presporamido n	AG-126 BML-260	Dec-RV/R- CMK	HA-1077	Lisinopril	LFM-A13					
4	DMSO		Cantharidin						HDBA (2- hydroxy-5-(2,5- dihydroxybenzyl) amino)benzoic acid)	Enalapril	SB-202190		SU1498		BML-257	DMSO
5	DMSO	E-64	SB-202590	Epigallocatech in gallate	Tyrosinase 51	PPACK	AG-370	Z-PLG-NHCH								
6	DMSO		Endonitil	GM6001	Cinigel		PD-144785	Z-FA-FMK	KN-62	Cilastatin	PF2		ZM 440829		SC-514	DMSO
7	DMSO	E-64-C	H-7		Tyrosinase 1	Z-ProM-Proinal	AG-879									
8	DMSO		Benzylphospho nic acid	NNGH (BML- 205)	Cinigel 2 Me		BML-267									
9	DMSO	E-64-d	H-9		Tyrosinase AG 1288	Thiopran (DL)	LY 294002	Z-VAD-FMK	KN-93	PKS-527	ZM-336372		BAV 11-7082		BML-259	DMSO
10	DMSO		L-p- Bromotetraniso le oxalate		BN-82002		BML-267 Ester									
11	DMSO	MDL-28170(Z-VF-CHO)	Staurosporine	Glicoxin	Tyrosinase AG 1478	TLCX	Wort- nannin	Etelactone B	ML-7		SU 4312		DRB (5,6- diphenyl-1,2,3,4- dihydro-2H-1,2,3- benzoxazole)		Aggenin	DMSO
12	DMSO		RK-682		Silikon		ORA									
13	DMSO	Z-Leu-Leu-CHO	AG-494		Tyrosinase AG 1295	TPCK		Aspirin	ML-9		AG-1296		HBDE (2,2', 3,3',4,4', 5,6,6'-hexahydroxy- 1,1'-biphenyl- 6,6'-dimethanol dimethyl ether)		BML-265 (Etiolmb analog)	DMSO
14	DMSO		FWJ-60475	MG-132		NSC-663284	OBAEster								Rapamycin	
15	DMSO	CA-074	AG-825	AAF-CMK	Tyrosinase 9		Hypericin	Agarctosan	2-Aminopurine		GW 5074		SP 600125			DMSO
16	DMSO		FWJ-60475 (AM)3		Cyclosporin A		Gossypol									
17	DMSO	CA-074-Me	Larvadulatin A	Apipranenine A	HNNPA (hydroxy-2- naphthalenyl thiophosphonic acid)	Funagilin	Rc 31-8220	URB-597	N9-isopropyl- dibutylamine		PamiloxyDL- carnitine Cl		Indirubin			DMSO
18	DMSO		Larvadulatin A		Pentamidine		Acetaminophen	4-Amino-D,L- benzylsuccinic acid								
19	DMSO	Napaul-ile-Tip-CHO	RG-14620	Beatin (Ubienox)	PRC-412	5-nitrostatin	Sphinganine	N- benzylsuccinic acid	Quercetin		Rotenone		Indirubin-3'- monoxime			DMSO
20	DMSO		Tetranitroic HCl		BVT-948											
21	DMSO	Peptastatin A	Tyrosinase 23	Boc-GVY-CHO	Picetatinol		H-89	Enytraleinide	Iso-Quercetin		Genistein		Y-27632			DMSO
22	DMSO		Cypemethim		B4-rhodamine											Oleparib
23	DMSO		Tyrosinase 25	Captopril	PP1	Nalmoxal mesylate	H-8	Aviclon	Roscovitine		Dactein		Kemparlone			DMSO
24	DMSO	3,4-Dichloroisocoumarin	Delamanthin		BML-268											Cisplatin

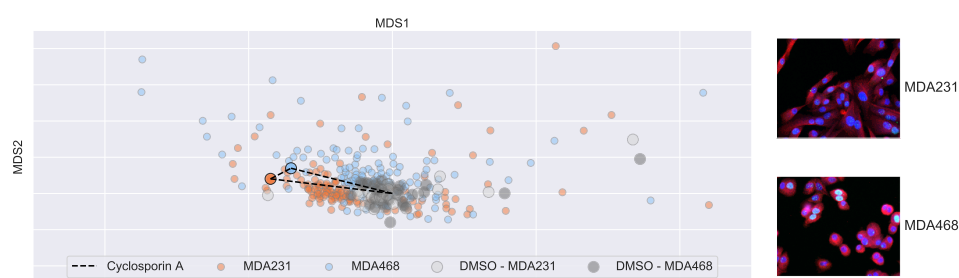
Figure A.2: Plate map of perturbations used for drug screen. Empty cells indicate empty wells. DMSO denotes negative controls.



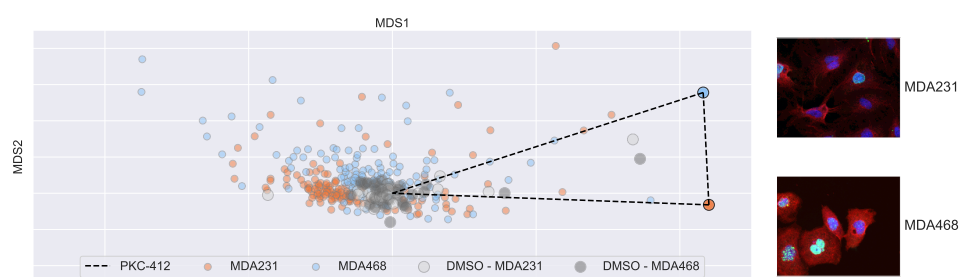
(a) Endothall takes effect in cell line MDA231 only.



(b) CL-82198 takes effect in cell line MDA468 only.



(c) Cyclosporin A takes a similar in both cell lines.



(d) PKC-412 takes differential effects in the two cell lines.

Figure A.3: MDS plots of each category of drug effect. The distances between the profiles are plotted as a line, as well as the respective distances to the centroid (origin).

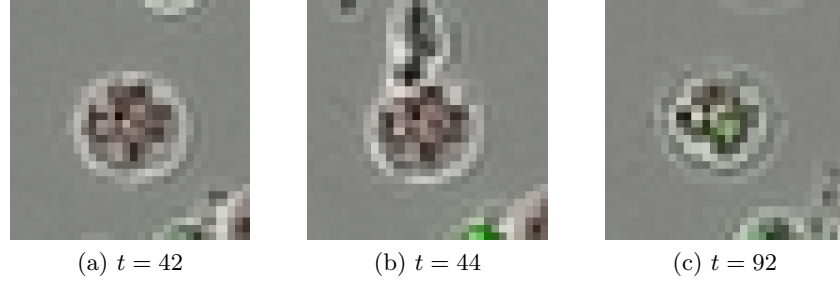


Figure A.4: A living Raji cell (a) is attacked by a CAR-T cell (b) and later dies (c).

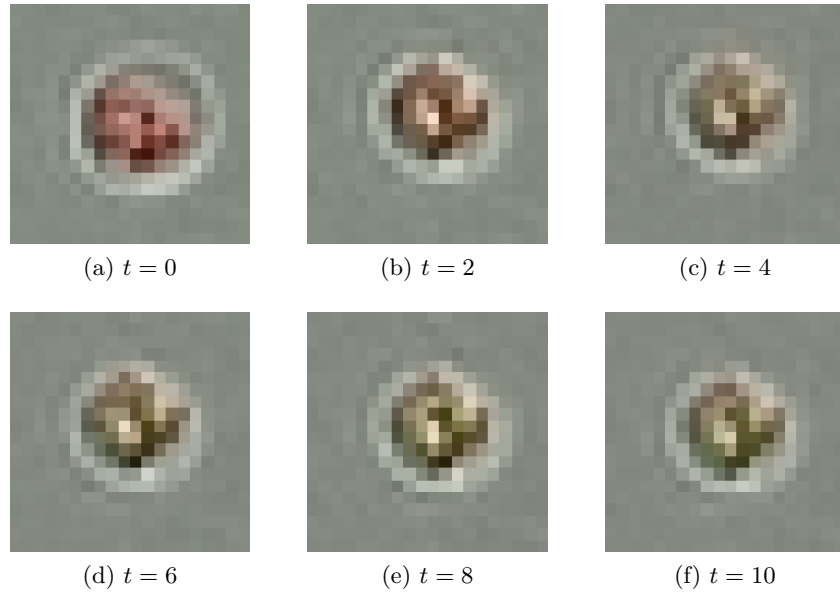


Figure A.5: Tracking the apoptosis of a Raji cell. Counterpart to Figure 5.2.

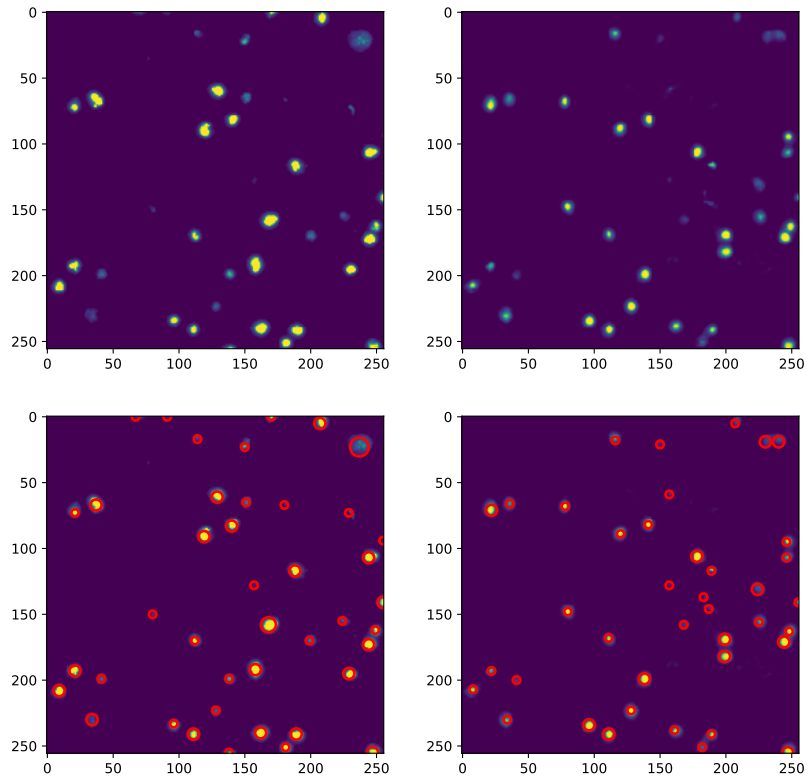


Figure A.6: Blob detection results. Left column shows ground truth, right column shows `pix2pix` prediction. With respect to the ground truth: true positives: 31; false positives: 9; and false negatives: 11.

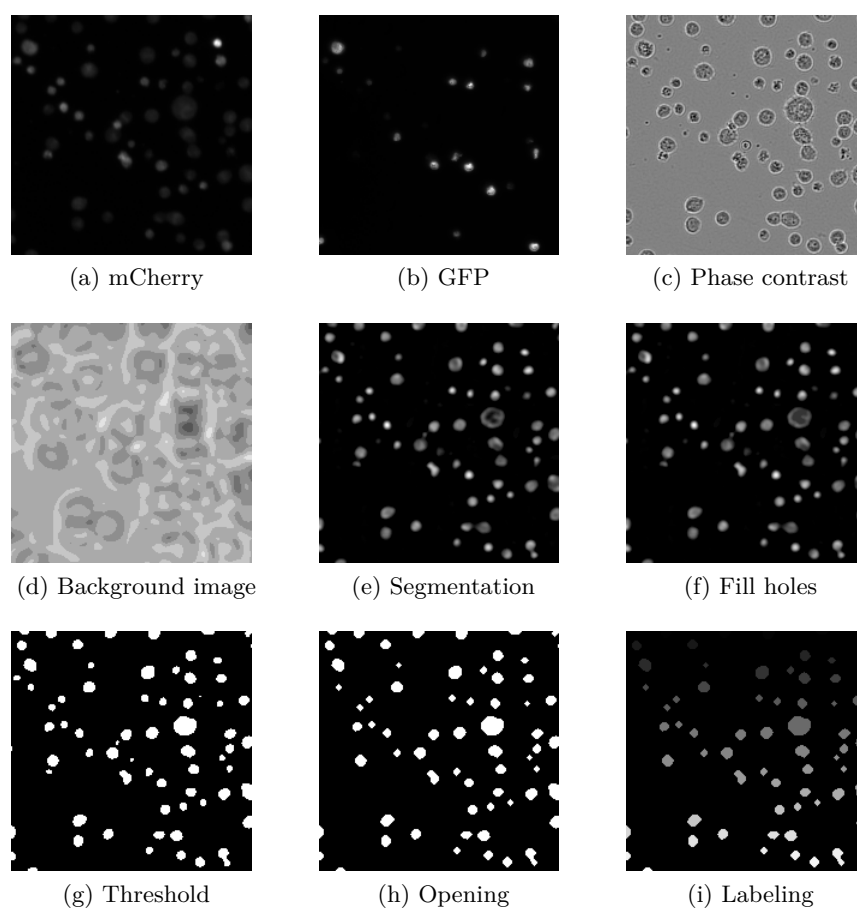


Figure A.7: An illustration of the automatic ground truth generation image processing pipeline. The first row are the image inputs.

Appendix B

Glossary of neural network architectures

B.1 Fully-connected GAN

Layer (type)	Output Shape	Param #
dense_42 (Dense)	(None, 256)	25856
batch_normalization_17 (Batch Normalization)	(None, 256)	1024
activation_17 (Activation)	(None, 256)	0
dense_43 (Dense)	(None, 784)	201488
batch_normalization_18 (Batch Normalization)	(None, 784)	3136
activation_18 (Activation)	(None, 784)	0
dense_44 (Dense)	(None, 576)	452160
reshape_9 (Reshape)	(None, 24, 24, 1)	0
Total params: 683,664		
Trainable params: 681,584		
Non-trainable params: 2,080		

Layer (type)	Output Shape	Param #
dense_45 (Dense)	(None, 512)	295424
leaky_re_lu_26 (LeakyReLU)	(None, 512)	0
dense_46 (Dense)	(None, 256)	131328
leaky_re_lu_27 (LeakyReLU)	(None, 256)	0
dense_47 (Dense)	(None, 1)	257
Total params: 427,009		
Trainable params: 427,009		
Non-trainable params: 0		

B.2 Deep convolutional GAN

Layer (type)	Output Shape	Param #
dense_16 (Dense)	(None, 4608)	465408
batch_normalization_7 (Batch Normalization)	(None, 4608)	18432
activation_7 (Activation)	(None, 4608)	0
reshape_4 (Reshape)	(None, 6, 6, 128)	0
up_sampling2d_7 (UpSampling2D)	(None, 12, 12, 128)	0
conv2d_13 (Conv2D)	(None, 12, 12, 64)	204864
batch_normalization_8 (Batch Normalization)	(None, 12, 12, 64)	256
activation_8 (Activation)	(None, 12, 12, 64)	0
up_sampling2d_8 (UpSampling2D)	(None, 24, 24, 64)	0
conv2d_14 (Conv2D)	(None, 24, 24, 1)	1601

Total params: 690,561
 Trainable params: 681,217
 Non-trainable params: 9,344

Layer (type)	Output Shape	Param #
dense_18 (Dense)	(None, 256)	25856
batch_normalization_9 (Batch Normalization)	(None, 256)	1024
activation_9 (Activation)	(None, 256)	0
dense_19 (Dense)	(None, 784)	201488
batch_normalization_10 (Batch Normalization)	(None, 784)	3136
activation_10 (Activation)	(None, 784)	0
dense_20 (Dense)	(None, 576)	452160
reshape_5 (Reshape)	(None, 24, 24, 1)	0
Total params: 683,664		
Trainable params: 681,584		
Non-trainable params: 2,080		

B.3 F-Net

Layer (type)	Output Shape	Param #	Connected to
input_2	(None, 256, 256, 1)	0	
conv2d_15	(None, 128, 128, 64)	1088	input_2
leaky_re_lu_8	(None, 128, 128, 64)	0	conv2d_15
conv2d_16	(None, 64, 64, 128)	131200	leaky_re_lu_8
leaky_re_lu_9	(None, 64, 64, 128)	0	conv2d_16

batch_norma_13	(None, 64, 64, 128)	512	leaky_re_lu_9
conv2d_17	(None, 32, 32, 256)	524544	batch_norma_13
leaky_re_lu_10	(None, 32, 32, 256)	0	conv2d_17
batch_norma_14	(None, 32, 32, 256)	1024	leaky_re_lu_10
conv2d_18	(None, 16, 16, 512)	2097664	batch_norma_14
leaky_re_lu_11	(None, 16, 16, 512)	0	conv2d_18
batch_norma_15	(None, 16, 16, 512)	2048	leaky_re_lu_11
conv2d_19	(None, 8, 8, 512)	4194816	batch_norma_15
leaky_re_lu_12	(None, 8, 8, 512)	0	conv2d_19
batch_norma_16	(None, 8, 8, 512)	2048	leaky_re_lu_12
conv2d_20	(None, 4, 4, 512)	4194816	batch_norma_16
leaky_re_lu_13	(None, 4, 4, 512)	0	conv2d_20
batch_norma_17	(None, 4, 4, 512)	2048	leaky_re_lu_13
conv2d_21	(None, 2, 2, 512)	4194816	batch_norma_17
leaky_re_lu_14	(None, 2, 2, 512)	0	conv2d_21
batch_norma_18	(None, 2, 2, 512)	2048	leaky_re_lu_14
up_sampling2d_8	(None, 4, 4, 512)	0	batch_norma_18
conv2d_22	(None, 4, 4, 512)	4194816	up_sampling2d_8
batch_norma_19	(None, 4, 4, 512)	2048	conv2d_22
concatenate_7	(None, 4, 4, 1024)	0	batch_norma_19 batch_norma_17
up_sampling2d_9	(None, 8, 8, 1024)	0	concatenate_7

conv2d_23	(None, 8, 8, 512)	8389120	up_sampling2d_9
batch_norma_20	(None, 8, 8, 512)	2048	conv2d_23
concatenate_8	(None, 8, 8, 1024)	0	batch_norma_20 batch_norma_16
up_sampling2d_10	(None, 16, 16, 1024)	0	concatenate_8
conv2d_24	(None, 16, 16, 512)	8389120	up_sampling2d_10
batch_norma_21	(None, 16, 16, 512)	2048	conv2d_24
concatenate_9	(None, 16, 16, 1024)	0	batch_norma_21 batch_norma_15
up_sampling2d_11	(None, 32, 32, 1024)	0	concatenate_9
conv2d_25	(None, 32, 32, 256)	4194560	up_sampling2d_11
batch_norma_22	(None, 32, 32, 256)	1024	conv2d_25
concatenate_10	(None, 32, 32, 512)	0	batch_norma_22 batch_norma_14
up_sampling2d_12	(None, 64, 64, 512)	0	concatenate_10
conv2d_26	(None, 64, 64, 128)	1048704	up_sampling2d_12
batch_norma_23	(None, 64, 64, 128)	512	conv2d_26
concatenate_11	(None, 64, 64, 256)	0	batch_norma_23 batch_norma_13
up_sampling2d_13	(None, 128, 128, 256)	0	concatenate_11
conv2d_27	(None, 128, 128, 64)	262208	up_sampling2d_13
batch_norma_24	(None, 128, 128, 64)	256	conv2d_27
concatenate_12	(None, 128, 128, 128)	0	batch_norma_24 leaky_re_lu_8
up_sampling2d_14	(None, 256, 256, 128)	0	concatenate_12

conv2d_28	(None, 256, 256, 1)	2049	up_sampling2d_14
=====			
Total params: 41,837,185			
Trainable params: 41,828,353			
Non-trainable params: 8,832			

B.4 PatchGAN

Layer (type)	Output Shape	Param #	Connected to
=====			
input_7	(None, 256, 256, 1)	0	

input_8	(None, 256, 256, 1)	0	

concatenate_15	(None, 256, 256, 2)	0	input_7 input_8

conv2d_39	(None, 128, 128, 64)	2112	concatenate_15

leaky_re_lu_23	(None, 128, 128, 64)	0	conv2d_39

conv2d_40	(None, 64, 64, 128)	131200	leaky_re_lu_23

leaky_re_lu_24	(None, 64, 64, 128)	0	conv2d_40

batch_norm_31	(None, 64, 64, 128)	512	leaky_re_lu_24

conv2d_41	(None, 32, 32, 256)	524544	batch_norm_31

leaky_re_lu_25	(None, 32, 32, 256)	0	conv2d_41

batch_norm_32	(None, 32, 32, 256)	1024	leaky_re_lu_25

conv2d_42	(None, 16, 16, 512)	2097664	batch_norm_32

leaky_re_lu_26	(None, 16, 16, 512)	0	conv2d_42

batch_norm_33	(None, 16, 16, 512)	2048	leaky_re_lu_26

```
-----  
conv2d_43      (None, 16, 16, 1)    8193      batch_norm_33  
=====
```

Total params: 2,767,297
Trainable params: 2,765,505
Non-trainable params: 1,792

```
-----
```


Appendix C

Analysing double-strand breaks in cultured cells for drug screening applications by causal inference

The contents of this appendix were published in the Proceedings of the 14th IEEE International Symposium on Biomedical Imaging (ISBI). Washington D.C., United State of America. April, 2018.

Double strand breaks (DSB) are a hallmark of DNA damage and genetic instability, which are important features of cancer cells. In addition, repair of DSBs provide interesting therapeutic targets. Fluorescence microscopy allows us to visualise DSBs in cells using a dedicated fluorescent marker, which is therefore an informative readout for drug screening applications. We therefore need robust methods in image analysis and statistical analysis to quantify DSBs in single cells and thereby to assess the drug effect with respect to the related pathways. The contribution of this paper is two-fold: first, we compare different DSB quantification schemes; and second we provide a sound statistical framework based on causal inference in order to detect drugs acting directly on DSBs. In particular this second aspect is so far notoriously neglected in the field, even though it is essential for the specific assignment of the drug effect.

C.1 Introduction

Dysfunction of the DNA repair mechanisms is a major hallmark of cancer. Monitoring DNA damage by fluorescently labeling double strand breaks (DSB) in cells is therefore an important readout in drug screening of cancer cell lines. DSBs occur when both strands of the DNA double helix are broken. DSBs have various causes, for example, cytotoxic radiation, or DNA replication over an existing single-strand break. Despite the natural DNA repair mechanisms of the cell, DSBs can be irreparable, leading ultimately to apoptosis, or hazardous DNA rearrangements. As such, DSBs are an interesting property to assess when analysing the effects of small compounds upon cancer cells.

The context of this research is a drug screen in triple-negative breast cancer (TNBC), a type of cancer that is characterised by the absence of genetic markers that are common targets for cancer therapies. The difficulty of treatment of TNBC make the search for effective new treatments particularly important. It must be noted however that DSB quantification is not limited to this application; it is a general marker often used in cancer research, and in particular in screening applications.

In Section C.2 we describe the experimental setup of the drug screen. In Section C.3 we compare three approaches to quantifying DSBs over the imaged cell populations. Then, we show that analysis of DSB numbers can be misleading for two reasons: first, the number of DSBs may depend on the cell cycle and drugs assigned to affect DSBs might actually only affect the cell cycle. Second, in drug screening, severely perturbed cells—such as mitotic or dead cells—are typically washed away prior to image acquisition. Such a *loss to followup* introduces a potential selection bias when performing significance tests. Both aspects have been neglected so far in the drug screening literature (Avondoglio et al. [2009], Garcia-Canton et al. [2013]). We use the tools of causal inference to probe for such systematic bias in our analysis in Section C.4. A comparison of the DSB quantifiers used and the analytic corrections are presented in Section C.5. Closing remarks are given in Section C.6.

C.2 Experimental setup

In a pilot study on multiple TNBC cell lines, we assembled 168 drugs, including two positive controls, for a drug screen at high concentration ($10\mu M$), alongside replicated negative controls of neutral agent *dimethyl sulfoxide* (DMSO) and untreated cells. The wells were seeded with a controlled 1000 and 1250 cells for MDA231 and MDA468 cell lines respectively, on sepa-

rate 384-well microplates. The drugs were administered identically on both plates. Following hibernation, the cells were fixed, washed, and stained with four fluorescent markers. Fluorescence microscopy ($20\times$ widefield with deconvolution image restoration) taken at four fields in each well produced two experimental datasets of 1536 multiplexed images apiece.

Four fluorescent stains were used for our microscopy: DAPI, Cyanine 3 (Cy3), Cyanine 5 (Cy5), and FITC. Of interest to our analysis are DAPI and Cy3, fluorescent markers for DNA and DSBs, respectively.

The final, washing step evacuates all unfastened cells from the well, such as mitotic or dead cells, as a consequence of perturbation or otherwise. The array of perturbations show a range of effects on cell mortality, from no apparent effect to a near or complete elimination of cells.

C.3 Approaches to measuring double-strand breaks

While biochemical techniques exist for DSB quantification ([Chailleux et al. \[2014\]](#)), obtaining faithful counts of DSBs directly from bioimages is problematic as the correspondence between DSBs and the recorded signal is unclear. Our strategy is therefore to extract a feature that has a high chance of being proportional to the amount of DSBs. The Cy3 signal typically manifests as a nebulous cloud, with intermittent peaks or spots of higher intensity. Samples are given in [Figure C.1](#). DSBs may be underrepresented by a simple spot count, as high intensity spots may correspond with multiple, localised breaks. On the other hand, DSBs may be overrepresented by a noisy Cy3 channel. We here compare three approaches to quantifying double-strand breaks. The approaches give correlated readouts, in particular between spot count and granulometry ([Figure C.2](#)) with $\rho = 0.84$ over the entire plate, and it seems that any will serve as a reasonable proxy to the true number of DSBs. Granulometry and spot density correlate with average intensity at $\rho = 0.44$ and $\rho = 0.42$ respectively.

In general, the features differ most in the rare cases when the marker is saturated and nothing stands out to be counted as a spot, yet the average intensity is artificially high (staining artifacts). We therefore give preference to spot-related features. The computation was performed by Cell Cognition ([Held et al. \[2010\]](#)), an open-source tool for the visualisation and analysis of HCS assays. As part of a standard computational pipeline, individual cell nuclei were first segmented by a combination of filtering and local thresholding ([Held et al. \[2010\]](#)). Touching nuclei are split as described in [Naylor et al. \[2017\]](#), using morphological local contrast dynamics (morphological dynamics) giving superior results to the traditional approach of filtering the

Euclidean distance map (data not shown). DSB analysis was then performed for each individual nucleus.

C.3.1 Counting spots with diameter openings

One approach to measuring DSBs is to count individual spots. For this, we used *diameter openings* (Walter et al. [2007]), a technique based on a similar flooding technique to the watershed algorithm, with the notable difference that no separation of regions is built and that the flooding stops as soon as the maximal extension of the region exceeds a user-defined diameter λ . Mathematically, the operator can be written as:

$$\begin{aligned} [\gamma_\lambda^\circ(f)](x) &= \sup \left\{ s \leq f(x) \mid \alpha \left(C_x \left[X_s^+(f) \right] \right) \geq \lambda \right\} \\ &= \bigcup \{ \gamma_B(f) \mid \alpha(B) \geq \lambda \}, \end{aligned} \quad (\text{C.1})$$

where $X_s^+(f)$ is the set of all pixels with $f(x) \geq s$, $C_x[A]$ is the connected component of set A containing point x and $\alpha(C_x)$ its maximal extension. Equation C.1 shows that this can also be written as the supremum of *all* morphological openings $\gamma_{B_i}(f)$ with structuring elements B_i whose diameter greater or equal to λ .

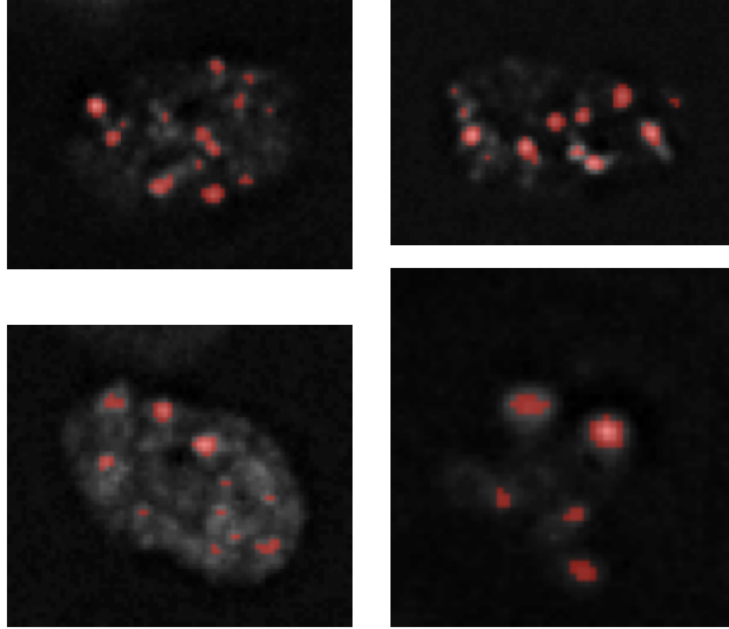


Figure C.1: Examples of spots (red) on cell nuclei detected with diameter openings on the Cy3 channel (grey).

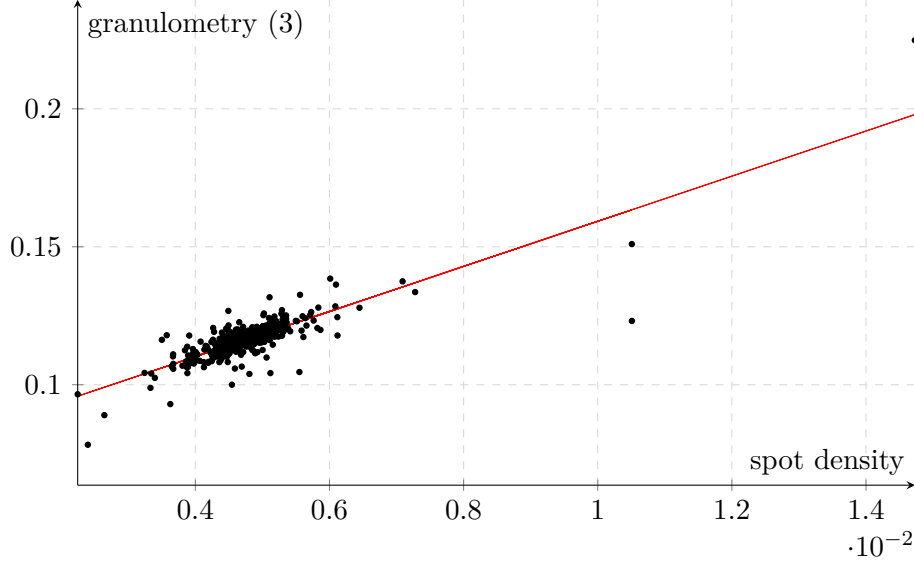


Figure C.2: Granulometry and spot density readouts show high correlation on average over all wells.

Building the difference to the original image $f - \gamma_\lambda^\circ(f)$ and a simple threshold operation thus allows one to extract all bright details with a maximal extension smaller than a certain value. The tunable parameters of the algorithm are the diameter of the structuring element and the value of the intensity threshold¹. Nuclei happen to have different sizes and we thus normalise the spot count into *spot density* by dividing by the nuclear area.

C.3.2 Granulometry-based features

An alternative strategy to get a proxy of DSB numbers without relying on hard segmentation, is to use *morphological granulometries*, i.e. the sum of the difference between opened versions of the initial image.

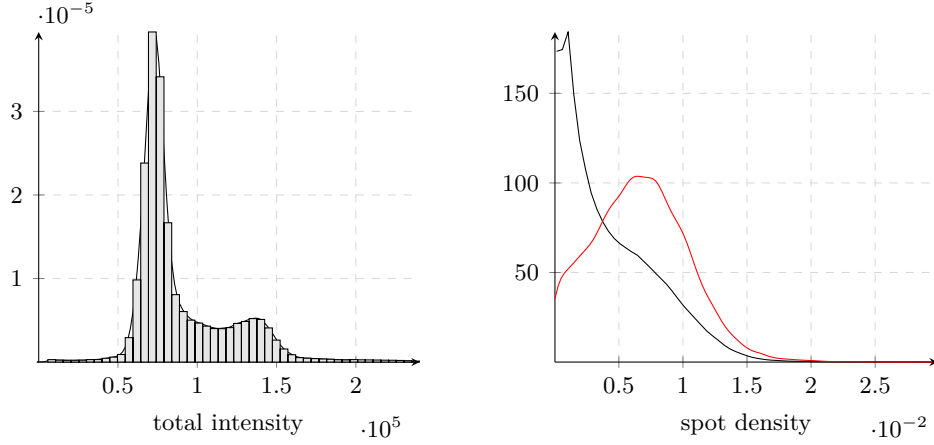
$$\sum_{x \in S_k} \gamma_{B_i} f(x) - \gamma_{B_{i+1}} f(x), \quad (\text{C.2})$$

where the structuring elements B_i fulfill the condition $B_i \subset B_{i+1}$ and S_k is the k -th cell.

C.3.3 Average intensity

We also hypothesised that an accumulation of γ -H2AX can be seen as an indicator of DNA breaks without a detectable organisation into spots, in

¹These were manually tuned to a diameter of 5 pixels, and threshold of 8.



(a) mCherry and GFP fluorescence (left and center) and phase contrast signals (right) at $t = 0$

(b) mCherry and GFP fluorescence (left and center) and phase contrast signals (right) at $t = 70$

Figure C.3: Distribution of cell total intensities on the DAPI channel for MDA231 cell line (a). The bimodal distribution is a consequence of the growth sustained between the G_1 (black) and G_2 (red) phases of the cell cycle. A suitable threshold of DAPI intensity significant divergence in the DSB distributions of the respective groups on the Cy3 channel (b).

particular in the case of very high densities. We therefore added the average intensity over the region of interest as a feature.

$$\frac{1}{\#S_k} \sum_{x \in S_k} f(x). \quad (\text{C.3})$$

C.4 Analysis

The aim of our analysis is to compare DSB levels from drug perturbations to levels in the negative control, DMSO. We used the Kolmogorov-Smirnov (K-S) test as the basis of comparison with significance declared at the 0.01 level.

As can be seen in Figure C.3a, there is a bimodal distribution in DAPI intensity considered over all cells, corresponding to cell cycle phases (pre- and post-DNA synthesis). The higher frequency of low intensities corresponds to the longer G_1 phase and more cells will be fixed in this phase. Secondly, the DNA content in this phase is constant, whereas the DNA content in S-phase is somewhere between G_1 and G_2 . Third, there are many aberrant morphologies, for which the DNA content is larger than the one of normal G_1 cells, but the exact value is not clearly determined. All of these elements

lead to a wide distribution of DAPI intensities for all nuclei that are not in G_1 . We can roughly estimate a suitable threshold between the two classes to create high and low intensity groups. Visualising the respective spot densities for these groups (Figure C.3b) reveals a shift in DSB distribution between the groups.

C.4.1 Causal considerations

Causal inference (Pearl et al. [2009]) is a framework for conducting statistical analysis that enables reasoning about causal factors in an experimental setting, so as to determine a relationship between a potential cause or *treatment* and a potential effect or *outcome*. Such an analysis is necessarily first endowed with a directed, acyclic causal graph, such as Figure C.4, by a domain expert. The causal graph postulates a feasible causal relation, respecting the otherwise impossible task of establishing causations from statistical data alone (Pearl et al. [2009]). In particular, causal models help to identify biases and necessary adjustments in analysis.

One aspect we looked into was the simultaneous effect of perturbations on cell death and DSBs, themselves potentially a cause of cell death. Due to the washing step in the experimental protocol, our observations are confined to those cells surviving the cytotoxic effect, that is, not undergoing apoptosis A . This is an example of *selection bias*, whereby the phenomenon of apoptosis determines those cells available for measurement. The bias is illustrated in Figure C.4, which traces a path from perturbation P to apoptosis A , both via DSBs levels D , as well as directly, in a path representing other, unmeasured, cytotoxic causes.

In certain scenarios, a selection bias may be corrected for by standardisation or IP-weighting, giving an unbiased estimate of the causal effect. However, the particular causal structure we observe here defies correction, as the levels of A (cell death or not) are not *exchangeable* with respect to D , a necessary condition for corrective techniques. This clearly indicates the need for experimental techniques where apoptotic cells remain observable. Despite

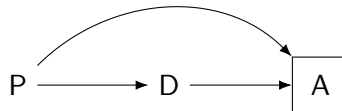


Figure C.4: Causal diagram including selection bias: P is the perturbation; D is the distribution of double-strand breaks and; A is the frequency of apoptosis. Conditioning (represented as a square) on the common effect of treatment P and outcome D creates a selection bias.

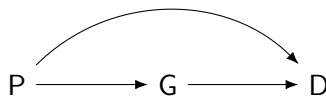


Figure C.5: Causal diagram showing effect modification of cell cycle phase: G is cell cycle phase; P is the perturbation, and D is the DNA damage.

this bias, independent analyses showed little difference in the distributions of DSBs within sparse (therefore highly apoptotic) and dense cell populations.

Another use case is in the effect of the cell cycle phase: given that some perturbations P affect the cell cycle phase G , and that DSBs levels D are influenced by both P and G , cell cycle variability acts as an *effect modifier* on D . These relations are expressed in the causal graph in Figure C.5. Though they are not a source of systematic bias, identifying effect modifiers can lead to an insightful stratified analysis. Through stratification, we were able to determine whether increased DSBs levels were a direct effect of perturbation, or an indirect effect of a modified cell cycle. In a stratified analysis, we compared the pooled cells of DMSO wells, as untreated subjects, with the cells of each perturbed well in turn. As we assessed 168 small compound perturbations, a multiple testing correction is due in our analysis. We compensate with the Benjamini-Hochberg correction for controlling the false discovery rate. We compare the effect of stratification in Section C.5.

C.5 Results

Despite the strong correlation between the three DSB measuring techniques, the number of perturbations exhibiting significance (hits) under each varies greatly. We reject the coarse approach of average intensity as being too sensitive to the ambient Cy3 signal and to staining artifacts. This shows in Table C.1 with unrealistically high hit rates for mean intensity. The most conservative are the granulometric features. The table presents the number of significant deviations from the negative control with and without stratification for the MDA231 cell line. The effects were seen for MDA468 also.

When we stratify, we report the number of hits without adjustment (all), for the G_1 and G_2 sub-populations of the wells separately, and the number of times these agree (joint). Taken apart, the subpopulations see fewer hits. This is a sign the perturbation influence on cell cycle is crucial.

Modifier	Double strand break quantifiers		
	Spot density	Granulometry	Mean intensity
All	45	34	87
G_1	40	10	62
G_2	15	14	34
Joint	9	4	18

Table C.1: Number of hits (out of 168) for each DSB quantifier on the MDA231 cell line. Significance at the 0.01 level with the Benjamini-Hochberg correction.

C.6 Conclusions

In this study, we have examined three viable approaches to measuring double strand breaks in fluorescence microscopy images. The more nuanced approaches of diameter openings and granulometry were found to give favourable outcomes. We have also introduced consideration of causal inference into a standard piece of analysis. We believe this framework can impact the statistical precision of other analyses in high content screening. An awareness of causal relationships and potential systematic biases has the potential to greatly improve experimental design and the precision of hit predictions in drug screening.

Appendix D

Supplementary analysis

D.1 Discovering phenotypic classes with unsupervised learning

Whereas in the wild type screen most cells fall neatly into recognisable classes (Section 3.4.4), the drug screen phenotypes are more obscure. By inspection, the cell populations are dramatically less rich phenotypically than the morphological screen. There are many aberrant morphologies (examples in Figure D.1) among the perturbed cell populations, but mitotic cells are scarce, and apoptotic cells are non-existent due to the washing of fluorescent dyes (and along with them, unfastened cells) prior to imaging. Machine learning is therefore an appropriate tool for wheedling out more subtle phenotypes.

Clustering algorithms are used on the respective cell line datasets in an attempt to discover cell classes. PCA is applied to reduce dimensionality and to select between features with high mutual correlation between some of the extracted features¹. The large dataset remains restrictive in size to applying certain clustering algorithms. The data is therefore subsampled. However, rather than uniform random sampling, the following is done: first, k-means clustering is applied to the dataset, choosing a potentially excessive number of clusters ($K = 30$). It is expected that this will identify small isolated clusters that may be lost by random sampling alone, as it is anticipated that interesting phenotypes will be scarce. Then, a subsample of the dataset is created by sampling 50 cells (where available) from each cluster. With this new, smaller dataset, clustering algorithms robust to anisotropic data are applied, including: DBSCAN (Ester et al. [1996]), spectral clustering

¹For example, many shape features extracted by Cell Cognition, such as area, perimeter, convex hull, will increase together.

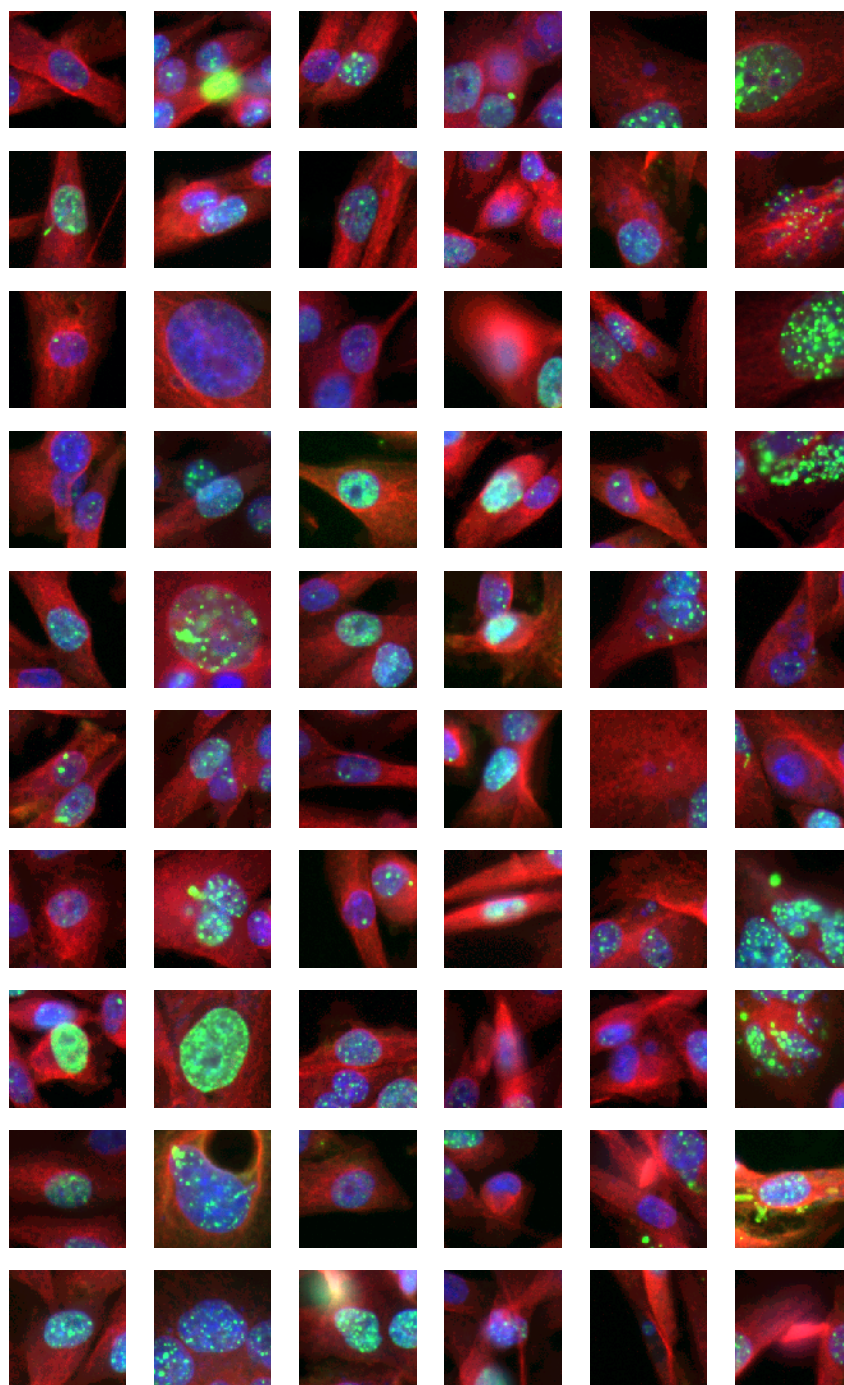


Figure D.1: Examples of spots (red) on cell nuclei detected with diameter openings on the Cy3 channel (grey).

(Von Luxburg [2007]), and hierarchical clustering (Ward Jr [1963]).

The clusters may be explored visually, by cropping examples of cells corresponding to each cluster. In Figure D.1 we see some subtle patterns emerge. The first cluster, the largest, appears to contain normal interphase cells. A second shows a mixture of normal and large interphase nuclei. Note that given the approach, it is anticipated that the predominant class, normal interphase, leaks into other clusters. Elsewhere, we see a smaller cluster comprising of many micro-nucleated cells, and another containing cells with high levels of DNA damage.

The ultimate specification of morphological classes should be determined by domain experts, with unsupervised learning merely providing a starting point. After this, one could pursue an analysis such as in Section 3.4.4. While this is an interesting approach, fully unsupervised methods are rather turned to in Chapter 3 to address the motivating questions of the assay.

D.2 Training a RoI classifier

A training set is designed based on randomly placing 16 randomly chosen 28×28 px digits from the MNIST digits dataset on a 256×256 background “canvas” image as shown in Figure D.2 (left).

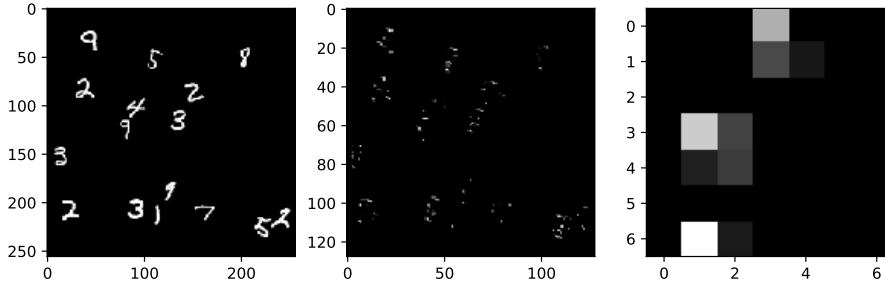


Figure D.2: From left to right: canvas input, activation map after MaxPooling layer, activation map after RoIAlign layer (reduced to single bounding box).

A region of interest (RoI) classifier is implemented as a convolutional neural network $\text{Conv}_{1,8} \rightarrow \text{ReLU} \rightarrow \text{MaxPool} \rightarrow \text{Conv}_{8,16} \rightarrow \text{ReLU} \rightarrow \text{RoIAlign} \rightarrow \text{FC}_{784,10}$, that is, a series of convolutional layers with ReLU activations followed by a RoIPool layer, and finally a fully-connected classification layer. The classifier is trained by a forward pass of such a canvas image, with the bounding box coordinates (x_0, y_0, x_1, y_1) for each of the digits. According to

the bounding boxes, the RoIAlign layer separates the image into a pseudo-batch of RoIs, which are subsequently classified into one of the 10 digit classes by a softmax function. Note that in this case, an RoIAlign layer is chosen to quantise the incoming 14×14 tensors to 7×7 tensors. The stages of the forward pass are shown in Figure D.2.

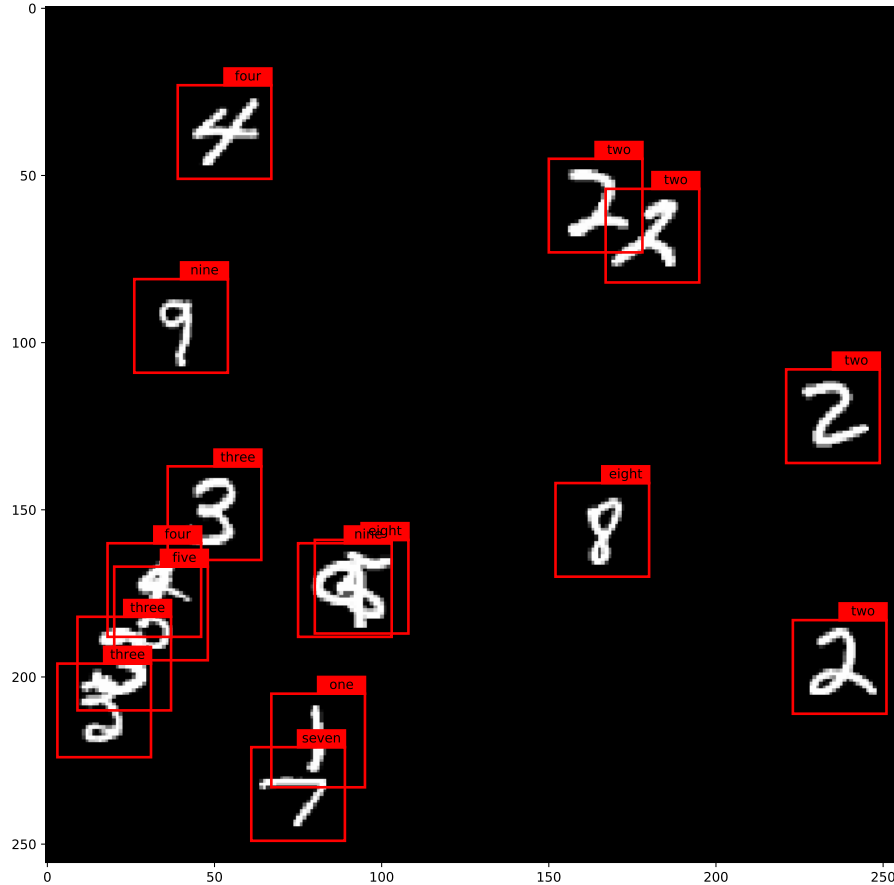


Figure D.3: Prediction of 16 digits over canvas, performed in a single forward pass.

Trained in this manner, the classifier achieves 97% accuracy as a classifier of MNIST digits, thus a small percentage shy of the accuracy of a typical MNIST classifier. In a single forward pass, the classifier can make predictions for the whole canvas, as illustrated in Figure D.4.

The RoI classifier can also be incorporated as a discriminator (that is, as a binary real/fake classifier) in a conditional image-to-image GAN(**pix2pix**) framework. Now, in addition to an input canvas of MNIST digits, conditioning images are created for each class (here ‘0’ or ‘1’ digits). An example is given in Figure D.4. A **pix2pix** system with a region of interest dis-

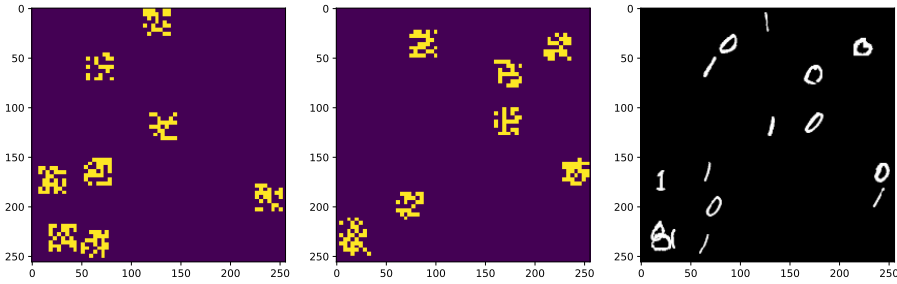


Figure D.4: Encodings of class and localisation specification (left and center) and corresponding train image for training a conditional GAN with RoI discriminator.

criminator is successfully trained. We see in Figure D.5 that the `pix2pix` generator has thus learned to synthesise images according to programmed localisation and classification information. This is compared (using the same specification of object localisation and class) to a ground truth image and the prediction of a plain `pix2pix` system. We see a far greater degree of mode collapse in this baseline compared with our proposed model.

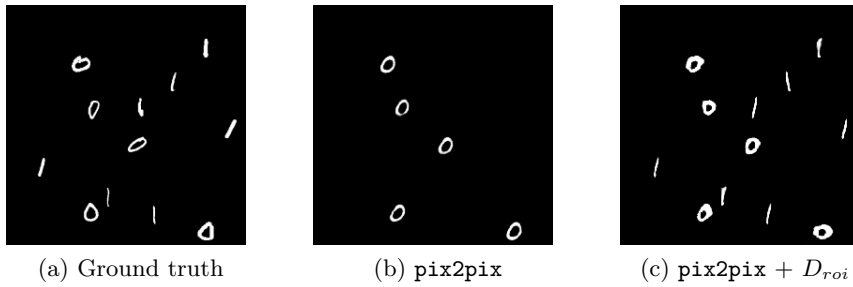


Figure D.5: With the same specification of object localisation and class, a ground truth image (a), a synthetic image from a plain `pix2pix` system (b) and synthetic image from our `pix2pix` with RoI discrimination.

Bibliography

- Eric S Lander, Lauren M Linton, Bruce Birren, Chad Nusbaum, Michael C Zody, Jennifer Baldwin, Keri Devon, Ken Dewar, Michael Doyle, William FitzHugh, et al. Initial sequencing and analysis of the human genome. 2001.
- David Houle, Diddahally R Govindaraju, and Stig Omholt. Phenomics: the next challenge. *Nature reviews genetics*, 11(12):855–866, 2010.
- Martin Mahner and Michael Kary. What exactly are genomes, genotypes and phenotypes? and what about phenomes? *Journal of theoretical biology*, 186(1):55–63, 1997.
- Gene Myers. Why bioimage informatics matters. *Nature methods*, 9(7):659, 2012.
- Steven A Haney. *High content screening: science, techniques and applications*. John Wiley & Sons, 2008.
- Beate Neumann, Thomas Walter, Jean-Karim Hériché, Jutta Bulkescher, Holger Erfle, Christian Conrad, Phill Rogers, Ina Poser, Michael Held, Urban Liebel, et al. Phenotypic profiling of the human genome by time-lapse microscopy reveals cell division genes. *Nature*, 464(7289):721, 2010.
- Jeremy C Simpson, Brigitte Joggerst, Vibor Laketa, Fatima Verissimo, Cihan Cetin, Holger Erfle, Mariana G Bexiga, Vasanth R Singan, Jean-Karim Hériché, Beate Neumann, et al. Genome-wide rnaï screening identifies human proteins with a regulatory function in the early secretory pathway. *Nature cell biology*, 14(7):764–774, 2012.
- Claudio Collinet, Martin Stöter, Charles R Bradshaw, Nikolay Samusik, Jochen C Rink, Denise Kenski, Bianca Habermann, Frank Buchholz, Robert Henschel, Matthias S Mueller, et al. Systems survey of endocytosis by multiparametric image analysis. *Nature*, 464(7286):243–249, 2010.
- Michael V Boland, Mia K Markey, and Robert F Murphy. Automated recognition of patterns characteristic of subcellular structures in fluorescence

- microscopy images. *Cytometry: The Journal of the International Society for Analytical Cytology*, 33(3):366–375, 1998.
- Steven A Haney, Peter LaPan, Jing Pan, and Jing Zhang. High-content screening moves to the front of the line. *Drug discovery today*, 11(19):889–894, 2006.
- Rainer Pepperkok and Jan Ellenberg. High-throughput fluorescence microscopy for systems biology. *Nature reviews Molecular cell biology*, 7(9):690–696, 2006a.
- David C Swinney and Jason Anthony. How were new medicines discovered? *Nature reviews Drug discovery*, 10(7):507–519, 2011.
- William F Scherer, Jerome T Syverton, and George O Gey. Studies on the propagation in vitro of poliomyelitis viruses: Iv. viral multiplication in a stable strain of human malignant epithelial cells (strain hela) derived from an epidermoid carcinoma of the cervix. *The Journal of experimental medicine*, 97(5):695–710, 1953.
- Vivien Marx. Cell-line authentication demystified. *Nature methods*, 11(5):483, 2014.
- Stefan Terjung, Thomas Walter, Arne Seitz, Beate Neumann, Rainer Pepperkok, and Jan Ellenberg. High-throughput microscopy using live mammalian cells. *Cold Spring Harbor Protocols*, 2010(8):pdb-top84, 2010.
- 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.
- Cynthia L Adams, Vadim Kutsyy, Daniel A Coleman, Ge Cong, Anne Moon Crompton, Kathleen A Elias, Donald R Oestreicher, Jay K Trautman, and Eugeni Vaisberg. Compound classification using image-based cellular phenotypes. In *Methods in enzymology*, volume 414, pages 440–468. Elsevier, 2006.
- 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. *Nature protocols*, 11(9):1757, 2016.
- Nikita Orlov, Lior Shamir, Tomasz Macura, Josiah Johnston, D Mark Eckley, and Ilya G Goldberg. Wnd-charm: Multi-purpose image classification using compound image transforms. *Pattern recognition letters*, 29(11):1684–1693, 2008.
- Virginie Uhlmann, Shantanu Singh, and Anne E Carpenter. Cp-charm:

- segmentation-free image classification made accessible. *BMC bioinformatics*, 17(1):51, 2016.
- Oren Z Kraus, Jimmy Lei Ba, and Brendan J Frey. Classifying and segmenting microscopy images with deep multiple instance learning. *Bioinformatics*, 32(12):i52–i59, 2016.
- Michael D Slack, Elisabeth D Martinez, Lani F Wu, and Steven J Altschuler. Characterizing heterogeneous cellular responses to perturbations. *Proceedings of the National Academy of Sciences*, pages pnas-0807038105, 2008.
- William J Godinez, Imtiaz Hossain, Stanley E Lazic, John W Davies, and Xian Zhang. A multi-scale convolutional neural network for phenotyping high-content cellular images. *Bioinformatics*, 33(13):2010–2019, 2017.
- Juan C Caicedo, Sam Cooper, Florian Heigwer, Scott Warchal, Peng Qiu, Csaba Molnar, Aliaksei S Vasilevich, Joseph D Barry, Harmanjit Singh Bansal, Oren Kraus, et al. Data-analysis strategies for image-based cell profiling. *Nature methods*, 14(9):849, 2017.
- Euan A Ashley. Towards precision medicine. *Nature Reviews Genetics*, 17(9):507, 2016.
- Chayakrit Krittanawong, HongJu Zhang, Zhen Wang, Mehmet Aydar, and Takeshi Kitai. Artificial intelligence in precision cardiovascular medicine. *Journal of the American College of Cardiology*, 69(21):2657–2664, 2017.
- Tim Hulsen, Saumya Shekhar Jamuar, Alan Moody, Jason Hansen Karnes, Varga Orsolya, Stine Hedensted, Roberto Spreafico, David A Hafler, and Eoin McKinney. From big data to precision medicine. *Frontiers in medicine*, 6:34, 2019.
- James C Costello, Laura M Heiser, Elisabeth Georgii, Mehmet Gönen, Michael P Menden, Nicholas J Wang, Mukesh Bansal, Petteri Hintsanen, Suleiman A Khan, John-Patrick Mpindi, et al. A community effort to assess and improve drug sensitivity prediction algorithms. *Nature biotechnology*, 32(12):1202, 2014.
- Steven J Altschuler and Lani F Wu. Cellular heterogeneity: do differences make a difference? *Cell*, 141(4):559–563, 2010.
- France Rose, Sreetama Basu, Elton Rexhepaj, Anne Chauchereau, Elaine Del Nery, and Auguste Genovesio. Compound functional prediction using multiple unrelated morphological profiling assays. *SLAS TECHNOLOGY: Translating Life Sciences Innovation*, 23(3):243–251, 2018.
- Scott J Warchal, John C Dawson, and Neil O Carragher. Development of the theta comparative cell scoring method to quantify diverse pheno-

- typic responses between distinct cell types. *Assay and drug development technologies*, 14(7):395–406, 2016.
- Christoph Sommer, Rudolf Hoefler, Matthias Samwer, and Daniel W Gerlich. A deep learning and novelty detection framework for rapid phenotyping in high-content screening. *Molecular biology of the cell*, 28(23):3428–3436, 2017.
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.
- Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.
- Vebjorn Ljosa, Katherine L Sokolnicki, and Anne E Carpenter. Annotated high-throughput microscopy image sets for validation. *Nature methods*, 9(7):637–637, 2012.
- 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. *Molecular cancer therapeutics*, 9(6):1913–1926, 2010.
- Chetak Kandaswamy, Luís M Silva, Luís A Alexandre, and Jorge M Santos. High-content analysis of breast cancer using single-cell deep transfer learning. *Journal of biomolecular screening*, 21(3):252–259, 2016.
- Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018.
- Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.
- Eric M Christiansen, Samuel J Yang, D Michael Ando, Ashkan Javaherian, Gaia Skibinski, Scott Lipnick, Elliot Mount, Alison O’Neil, Kevan Shah, Alicia K Lee, et al. In silico labeling: Predicting fluorescent labels in unlabeled images. *Cell*, 173(3):792–803, 2018.
- Chawin Ounkomol, Sharmishta Seshamani, Mary M Maleckar, Forrest Collman, and Gregory R Johnson. Label-free prediction of three-

- dimensional fluorescence images from transmitted-light microscopy. *Nature methods*, 15(11):917, 2018.
- Sajith Kecheril Sadanandan, Petter Ranefall, Sylvie Le Guyader, and Carolina Wählby. Automated training of deep convolutional neural networks for cell segmentation. *Scientific reports*, 7(1):7860, 2017.
- Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014a.
- Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.
- Anton Osokin, Anatole Chessel, Rafael E Carazo Salas, and Federico Vaggi. Gans for biological image synthesis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2233–2242, 2017.
- Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.
- Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.
- Stephan J Ihle, Andreas M Reichmuth, Sophie Girardin, Hana Han, Flurin Stauffer, Anne Bonnin, Marco Stampanoni, Karthik Pattisapu, János Vörös, and Csaba Forró. Unsupervised data to content transformation with histogram-matching cycle-consistent generative adversarial networks. *Nature Machine Intelligence*, 1(10):461–470, 2019.
- Joseph Boyd, Alice Pinheiro, Elaine Del Nery, Fabien Rey, and Thomas Walter. Analysing double-strand breaks in cultured cells for drug screening applications by causal inference. In *Biomedical Imaging (ISBI 2018), 2018 IEEE 15th International Symposium on*, pages 445–448. IEEE, 2018.
- Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, and Mario Marchand. Domain-adversarial neural networks. *arXiv preprint arXiv:1412.4446*, 2014.
- Joseph C Boyd, Alice Pinheiro, Elaine Del Nery, Fabien Rey, and Thomas Walter. Domain-invariant features for mechanism of action prediction in a multi-cell-line drug screen. *Bioinformatics*, 36(5):1607–1613, 2020a.

- Joseph Boyd, Fabien Rey, Alice Pinheiro, Elaine Del Nery, and Thomas Walter. Domain-invariant features for mechanism of action prediction in a multi-cell-line drug screen, May 2019a. URL <https://zenodo.org/record/2677923>.
- Joseph Boyd, Zelia Gouveia, Franck Perez, and Thomas Walter. Experimentally-generated ground truth for detecting cell types in an image-based immunotherapy screen. In *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI 2020)*. IEEE, 2020b.
- Joseph Boyd, Zelia Gouveia, Franck Perez, and Thomas Walter. Lymphocytes dataset, Oct 2019b. URL <https://zenodo.org/record/3515446>.
- Peter Naylor, Joseph Boyd, Marick Laé, Fabien Rey, and Thomas Walter. Predicting residual cancer burden in a triple negative breast cancer cohort. In *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pages 933–937. IEEE, 2019.
- Beyrem Khalfaoui, Joseph Boyd, and Jean-Philippe Vert. Adaptive structured noise injection for shallow and deep neural networks. 2019.
- Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.
- Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew

- Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019.
- Andrej Karpathy. What i learned from competing against a convnet on imagenet. *Andrej Karpathy Blog*, 5:1–15, 2014.
- Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.
- Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.
- Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.
- Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.
- Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.
- Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.
- Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.
- Kurt Hornik, Maxwell Stinchcombe, Halbert White, et al. Multilayer feed-forward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

- Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.
- Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.
- Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.
- Patrice Y Simard, David Steinkraus, John C Platt, et al. Best practices for convolutional neural networks applied to visual document analysis. In *Icdar*, volume 3, 2003.
- Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014b.
- Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.
- Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.
- Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.
- Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.
- Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*, 2014.
- Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.
- Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
- Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

- Aayush Bansal, Shugao Ma, Deva Ramanan, and Yaser Sheikh. Recycle-gan: Unsupervised video retargeting. In *Proceedings of the European conference on computer vision (ECCV)*, pages 119–135, 2018.
- Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.
- Clifford A Hudis and Luca Gianni. Triple-negative breast cancer: an unmet medical need. *Oncologist*, 16, 2011.
- Kathryn J Chavez, Sireesha V Garimella, and Stanley Lipkowitz. Triple negative breast cancer cell lines: one tool in the search for better treatment of triple negative breast cancer. *Breast disease*, 32(1-2):35, 2010.
- Homero Gonçalves Jr, Maximiliano Ribeiro Guerra, Jane Rocha Duarte Cintra, Vívian Assis Fayer, Igor Vilela Brum, and Maria Teresa Bustamante Teixeira. Survival study of triple-negative and non-triple-negative breast cancer in a brazilian cohort. *Clinical Medicine Insights: Oncology*, 12:1179554918790563, 2018.
- Rana Hatem, Rania El Botty, Sophie Chateau-Joubert, Jean-Luc Servely, Dalila Labiod, Ludmilla de Plater, Franck Assayag, Florence Coussy, Céline Callens, Sophie Vacher, et al. Targeting mtor pathway inhibits tumor growth in different molecular subtypes of triple-negative breast cancers. *Oncotarget*, 7(30):48206, 2016.
- Jorge Filmus, Michael N Pollak, Relda Cailleau, and Ronald N Buick. Mda-468, a human breast cancer cell line with a high number of epidermal growth factor (egf) receptors, has an amplified egf receptor gene and is growth inhibited by egf. *Biochemical and biophysical research communications*, 128(2):898–905, 1985.
- Xiaofeng Dai, Hongye Cheng, Zhonghu Bai, and Jia Li. Breast cancer cell line classification and its relevance with breast tumor subtyping. *Journal of Cancer*, 8(16):3131, 2017.
- Etienne Y Lasfargues and Luciano Ozzello. Cultivation of human breast carcinomas. *Journal of the National Cancer Institute*, 21(6):1131–1147, 1958.
- Adi F Gazdar, Venkatesh Kurvari, Arvind Virmani, Lauren Gollahon, Masahiro Sakaguchi, Max Westerfield, Duli Kodagoda, Victor Stasny, H Thomas Cunningham, Ignacio I Wistuba, et al. Characterization of paired tumor and non-tumor cell lines established from patients with breast cancer. *International journal of cancer*, 78(6):766–774, 1998.
- Adeline J Hackett, Helene S Smith, E Louise Springer, Robert B Owens, Walter A Nelson-Rees, John L Riggs, and Murray B Gardner. Two syn-

- geneic cell lines from human breast tissue: the aneuploid mammary epithelial (hs578t) and the diploid myoepithelial (hs578bst) cell lines. *Journal of the National Cancer Institute*, 58(6):1795–1806, 1977.
- Herbert D Soule, Terry M Maloney, Sandra R Wolman, Ward D Peterson, Richard Brenz, Charles M McGrath, Jose Russo, Robert J Pauley, Richard F Jones, and SC Brooks. Isolation and characterization of a spontaneously immortalized human breast epithelial cell line, mcf-10. *Cancer research*, 50(18):6075–6086, 1990.
- BR Brinkley, PT Beall, LJ Wible, ML Mace, Donna S Turner, and RM Cailleau. Variations in cell form and cytoskeleton in human breast carcinoma cells in vitro. *Cancer research*, 40(9):3118–3129, 1980.
- T Huang, GJTG Y Yang, and Greory Tang. A fast two-dimensional median filtering algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 27(1):13–18, 1979.
- Serge Beucher and Christian Lantuéjoul. Use of watersheds in contour detection. 1979.
- Michel Grimaud. New measure of contrast: the dynamics. volume 1769, pages 292–305, 1992. doi: 10.1117/12.60650. URL <http://dx.doi.org/10.1117/12.60650>.
- Thouis R Jones, Anne Carpenter, and Polina Golland. Voronoi-based segmentation of cells on image manifolds. In *International Workshop on Computer Vision for Biomedical Image Applications*, pages 535–543. Springer, 2005.
- Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.
- Michael Held, Michael HA Schmitz, Bernd Fischer, Thomas Walter, Beate Neumann, Michael H Olma, Matthias Peter, Jan Ellenberg, and Daniel W Gerlich. Cellcognition: time-resolved phenotype annotation in high-throughput live cell imaging. *Nature methods*, 7(9):747–754, 2010.
- Thomas Walter, Michael Held, Beate Neumann, Jean-Karim Hériché, Christian Conrad, Rainer Pepperkok, and Jan Ellenberg. Automatic identification and clustering of chromosome phenotypes in a genome wide rnai screen by time-lapse imaging. *Journal of structural biology*, 170(1):1–9, 2010a.
- Iurie Caraus, Abdulaziz A. Alsuwailem, Robert Nadon, and Vladimir Makarenkov. Detecting and overcoming systematic bias in high-throughput screening technologies: A comprehensive review of practical issues and methodological solutions. 16(6):974–986. ISSN 1467-5463. doi: 10.1093/bib/bbv004. URL <https://doi.org/10.1093/bib/bbv004>.

- Vebjorn Ljosa and Anne E. Carpenter. Introduction to the Quantitative Analysis of Two-Dimensional Fluorescence Microscopy Images for Cell-Based Screening. 5(12):e1000603. doi: 10.1371/journal.pcbi.1000603. URL <https://doi.org/10.1371/journal.pcbi.1000603>.
- DE Pegg. Viability assays for preserved cells, tissues, and organs. *Cryobiology*, 26(3):212–231, 1989.
- Leland McInnes and John Healy. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.
- Rainer Pepperkok and Jan Ellenberg. High-throughput fluorescence microscopy for systems biology. *Nature reviews Molecular cell biology*, 7(9):690, 2006b.
- 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, et al. Comparison of methods for image-based profiling of cellular morphological responses to small-molecule treatment. *Journal of biomolecular screening*, 18(10):1321–1329, 2013.
- Serge PJM Horbach and Willem Halffman. The ghosts of hela: How cell line misidentification contaminates the scientific literature. *PloS one*, 12(10):e0186281, 2017.
- Yansheng Liu, Yang Mi, Torsten Mueller, Saskia Kreibich, Evan G Williams, Audrey Van Drogen, Christelle Borel, Max Frank, Pierre-Luc Germain, Isabell Bludau, et al. Multi-omic measurements of heterogeneity in hela cells across laboratories. *Nature biotechnology*, 37(3):314, 2019.
- Shantanu Singh, M-A BRAY, TR Jones, and AE Carpenter. Pipeline for illumination correction of images for high-throughput microscopy. *Journal of microscopy*, 256(3):231–236, 2014.
- Thomas Walter, Michael Held, Beate Neumann, Jean-Karim Hériché, Christian Conrad, Rainer Pepperkok, and Jan Ellenberg. Automatic identification and clustering of chromosome phenotypes in a genome wide RNAi screen by time-lapse imaging. *Journal of structural biology*, 170(1):1–9, apr 2010b. ISSN 1095-8657. doi: 10.1016/j.jsb.2009.10.004. URL <http://www.ncbi.nlm.nih.gov/pubmed/19854275>.
- Daniel Müllner et al. fastcluster: Fast hierarchical, agglomerative clustering routines for r and python. *Journal of Statistical Software*, 53(9):1–18, 2013.
- Lit-Hsin Loo, Lani F Wu, and Steven J Altschuler. Image-based multivariate profiling of drug responses from single cells. *Nature methods*, 4(5):445, 2007.

- Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.
- Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Pasos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- François Chollet et al. Keras. <https://keras.io>, 2015.
- Stéfan van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Goullart, and Tony Yu. scikit-image: image processing in Python. *PeerJ*, 2:e453, 2014. ISSN 2167-8359. doi: 10.7717/peerj.453. URL <https://doi.org/10.7717/peerj.453>.
- Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.
- Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180, 2016.
- Mohamed-Reda Benmebarek, Clara Helke Karches, Bruno Loureiro Cadilha, Stefanie Lesch, Stefan Endres, and Sebastian Kobold. Killing mechanisms of chimeric antigen receptor (car) t cells. *International journal of molecular sciences*, 20(6):1283, 2019.
- Shannon L Maude, David T Teachey, David L Porter, and Stephan A Grupp. Cd19-targeted chimeric antigen receptor t-cell therapy for acute lymphoblastic leukemia. *Blood*, 125(26):4017–4023, 2015.
- M Sadelain, R Brentjens, and I Riviere. The basic principles of chimeric antigen receptor (car) design 3 (4):, 2013.
- Stephen P Hunger and Charles G Mullighan. Acute lymphoblastic leukemia in children. *New England Journal of Medicine*, 373(16):1541–1552, 2015.
- Shannon L Maude, Noelle Frey, Pamela A Shaw, Richard Aplenc, David M Barrett, Nancy J Bunin, Anne Chew, Vanessa E Gonzalez, Zhaohui Zheng, Simon F Lacey, et al. Chimeric antigen receptor t cells for sustained remissions in leukemia. *New England Journal of Medicine*, 371(16):1507–1517, 2014.

- Zhenguang Wang, Yelei Guo, and Weidong Han. Current status and perspectives of chimeric antigen receptor modified t cells for cancer treatment. *Protein & cell*, 8(12):896–925, 2017.
- MA Epstein, BG Achong, YM Barr, B Zajac, G Henle, and W Henle. Morphological and virological investigations on cultured burkitt tumor lymphoblasts (strain raji). *Journal of the National Cancer Institute*, 37(4): 547–559, 1966.
- Gyuhyun Lee, Jeong-Woo Oh, Mi-Sun Kang, Nam-Gu Her, Myoung-Hee Kim, and Won-Ki Jeong. Deephcs: Bright-field to fluorescence microscopy image conversion using deep learning for label-free high-content screening. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 335–343. Springer, 2018.
- Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.
- Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
- Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning with applications to medical imaging. *arXiv preprint arXiv:1902.07208*, 2019.
- Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *2010 IEEE Computer Society Conference on computer vision and pattern recognition*, pages 2528–2535. IEEE, 2010.
- Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.
- Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5188–5196, 2015.
- Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in neural information processing systems*, pages 262–270, 2015.
- Xu Zheng, Tejo Chalasani, Koustav Ghosal, Sebastian Lutz, and Aljosa Smolic. Stada: Style transfer as data augmentation. *arXiv preprint arXiv:1909.01056*, 2019.

- Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: <https://doi.org/10.1038/s41592-019-0686-2>.
- Martin Bock, Amit Kumar Tyagi, Jan-Ulrich Kreft, and Wolfgang Alt. Generalized voronoi tessellation as a model of two-dimensional cell tissue dynamics. *Bulletin of mathematical biology*, 72(7):1696–1731, 2010.
- Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.
- Reka Hollandi, Abel Szkalisity, Timea Toth, Ervin Tasnadi, Csaba Molnar, Botond Mathe, Istvan Grexa, Jozsef Molnar, Arpad Balind, Mate Gorbe, et al. A deep learning framework for nucleus segmentation using image style transfer. *bioRxiv*, page 580605, 2019.
- Chichen Fu, Soonam Lee, David Joon Ho, Shuo Han, Paul Salama, Kenneth W Dunn, and Edward J Delp. Three dimensional fluorescence microscopy image synthesis and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2221–2229, 2018.
- Juan Wang and Yongyi Yang. A context-sensitive deep learning approach for microcalcification detection in mammograms. *Pattern recognition*, 78: 12–22, 2018.
- Devin P Sullivan, Casper F Winsnes, Lovisa Åkesson, Martin Hjelmare, Mikaela Wiking, Rutger Schutten, Linzi Campbell, Hjalti Leifsson, Scott Rhodes, Andie Nordgren, et al. Deep learning is combined with massive-scale citizen science to improve large-scale image classification. *Nature biotechnology*, 36(9):820–828, 2018.
- Jacob Whitehill and Anand Ramakrishnan. Automatic classifiers as scientific instruments: One step further away from ground-truth. *arXiv preprint arXiv:1812.08255*, 2018.
- Federica Eduati, Lara M Mangravite, Tao Wang, Hao Tang, J Christopher Bare, Ruili Huang, Thea Norman, Mike Kellen, Michael P Menden, Jichen

- Yang, et al. Prediction of human population responses to toxic compounds by a collaborative competition. *Nature biotechnology*, 33(9):933–940, 2015.
- Benoit Playe, Chloe-Agathe Azencott, and Veronique Stoven. Efficient multi-task chemogenomics for drug specificity prediction. *PloS one*, 13(10), 2018.
- Dane Avondoglio, Tamalee Scott, Whoon Jong Kil, Mary Sproull, Philip J Tofilon, and Kevin Camphausen. High throughput evaluation of gamma-h2ax. *Radiation Oncology*, 4(1):31, 2009.
- Carolina Garcia-Canton, Arturo Anadon, and Clive Meredith. Assessment of the in vitro γ h2ax assay by high content screening as a novel genotoxicity test. *Mutation Research/Genetic Toxicology and Environmental Mutagenesis*, 757(2):158–166, 2013.
- Catherine Chailleux, François Aymard, Pierre Caron, Virginie Daburon, Céline Courilleau, Yvan Canitrot, Gaëlle Legube, and Didier Trouche. Quantifying dna double-strand breaks induced by site-specific endonucleases in living cells by ligation-mediated purification. *Nature protocols*, 9(3):517–528, 2014.
- Peter Naylor, Marick Lae, Fabien Rey, and Thomas Walter. Nuclei segmentation in histopathology images using deep neural networks. In *Proceedings of the 12th IEEE International Symposium on Biomedical Imaging (ISBI): From nano to macro*, pages 933–936, 04 2017.
- Thomas Walter, Pascale Massin, Ali Erginay, Richard Ordonez, Clothilde Jeulin, and Jean-Claude Klein. Automatic detection of microaneurysms in color fundus images. *Medical Image Analysis*, 11(6):555–66, 2007.
- Judea Pearl et al. Causal inference in statistics: An overview. *Statistics Surveys*, 3:96–146, 2009.
- Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.
- Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.
- Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

RÉSUMÉ

Le phénotypage computationnel est un ensemble de technologies émergentes permettant d'étudier systématiquement le rôle du génome dans l'obtention de phénotypes, les caractéristiques observables d'un organisme et de ses sous-systèmes. En particulier, les essais cellulaires permettent de cribler des panels de petites molécules ou de moduler l'expression des gènes, et de quantifier les effets sur les caractéristiques phénotypiques allant de la viabilité à la morphologie cellulaire. Le criblage à haut contenu étend les méthodologies des criblages cellulaires à une lecture à haut contenu basée sur des images, en particulier les canaux multiplexés de la microscopie à fluorescence. Les cribles basés sur de multiples lignées cellulaires sont aptes à différencier les phénotypes de différents sous-types d'une maladie, représentant l'hétérogénéité moléculaire concernée dans la conception de thérapies médicales de précision. Ces modèles biologiques plus riches sous-tendent une approche plus ciblée pour le traitement de maladies mortelles telles que le cancer. Un défi permanent pour le criblage à haut contenu est donc la synthèse des lectures hétérogènes dans les cribles à multiples lignées cellulaires. Parallèlement, l'état de l'art établi en matière d'applications d'analyse d'images et de vision par ordinateur est l'apprentissage profond. Cependant, son rôle dans le criblage à haut contenu ne fait que commencer à être réalisé. Cette thèse aborde deux problématiques de l'analyse à haut contenu des lignées cellulaires cancéreuses. Les contributions sont les suivantes : (i) une démonstration du potentiel d'apprentissage profond et de modèles générateurs dans le criblage à haut contenu ; (ii) une solution basée sur l'apprentissage profond au problème de l'hétérogénéité dans un criblage de médicaments sur plusieurs lignées cellulaires ; et (iii) de nouvelles applications de modèles de traduction d'image à image comme alternative à la microscopie à fluorescence coûteuse actuellement nécessaire pour le criblage à haut contenu.

MOTS CLÉS

Apprentissage profond, criblage à haut contenu, lignées cellulaires, phénotypage computationnel.

ABSTRACT

Computational phenotyping is an emergent set of technologies for systematically studying the role of the genome in eliciting phenotypes, the observable characteristics of an organism and its subsystems. In particular, cell-based assays screen panels of small compound drugs or otherwise modulations of gene expression, and quantify the effects on phenotypic characteristics ranging from viability to cell morphology. High content screening extends the methodologies of cell-based screens to a high content readout based on images, in particular the multiplexed channels of fluorescence microscopy. Screens based on multiple cell lines are apt to differentiating phenotypes across different subtypes of a disease, representing the molecular heterogeneity concerned in the design of precision medicine therapies. These richer biological models underpin a more targeted approach for treating deadly diseases such as cancer. An ongoing challenge for high content screening is therefore the synthesis of the heterogeneous readouts in multi-cell-line screens. Concurrently, deep learning is the established state-of-the-art image analysis and computer vision applications. However, its role in high content screening is only beginning to be realised. This dissertation spans two problem settings in the high content analysis of cancer cell lines. The contributions are the following: (i) a demonstration of the potential for deep learning and generative models in high content screening; (ii) a deep learning-based solution to the problem of heterogeneity in a multi-cell-line drug screen; and (iii) novel applications of image-to-image translation models as an alternative to the expensive fluorescence microscopy currently required for high content screening.

KEYWORDS

Deep learning, high content screening, cell lines, computational phenotyping.