UPPSALA
UNIVERSITET

Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology 2025

Image Processing, Machine Learning and Visualization for Tissue Analysis

LESLIE SOLORZANO

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2021

ISSN 1651-6214
ISBN 978-91-513-1173-9
urn:nbn:se:uu:diva-438775

Dissertation presented at Uppsala University to be publicly examined in Room IX, Universitetshuset, Biskopsgatan 3, Uppsala, Wednesday, 12 May 2021 at 13:00 for the degree of Doctor of Philosophy. The examination will be conducted in English. Faculty examiner: Alexandru Cristian Telea (Universiteit Utrecht).

Abstract

Solorzano, L. 2021. Image Processing, Machine Learning and Visualization for Tissue Analysis. *Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology* 2025. 66 pp. Uppsala: Acta Universitatis Upsaliensis. ISBN 978-91-513-1173-9.

Knowledge discovery for understanding mechanisms of disease requires the integration of multiple sources of data collected at various magnifications and by different imaging techniques. Using spatial information, we can build maps of tissue and cells in which it is possible to extract, e.g., measurements of cell morphology, protein expression, and gene expression. These measurements reveal knowledge about cells such as their identity, origin, density, structural organization, activity, and interactions with other cells and cell communities. Knowledge that can be correlated with survival and drug effectiveness. This thesis presents multidisciplinary projects that include a variety of methods for image and data analysis applied to images coming from fluorescence- and brightfield microscopy.

In brightfield images, the number of proteins that can be observed in the same tissue section is limited. To overcome this, we identified protein expression coming from consecutive tissue sections and fused images using registration to quantify protein co-expression. Here, the main challenge was to build a framework handling very large images with a combination of rigid and non-rigid image registration.

Using multiplex fluorescence microscopy techniques, many different molecular markers can be used in parallel, and here we approached the challenge to decipher cell classes based on marker combinations. We used ensembles of machine learning models to perform cell classification, both increasing performance over a single model and to get a measure of confidence of the predictions. We also used resulting cell classes and locations as input to a graph neural network to learn cell neighborhoods that may be correlated with disease.

Finally, the work leading to this thesis included the creation of an interactive visualization tool, TissUUmaps. Whole slide tissue images are often enormous and can be associated with large numbers of data points, creating challenges which call for advanced methods in processing and visualization. We built TissUUmaps so that it could visualize millions of data points from in situ sequencing experiments and enable contextual study of gene expression directly in the tissue at cellular and sub-cellular resolution. We also used TissUUmaps for interactive image registration, overlay of regions of interest, and visualization of tissue and corresponding cancer grades produced by deep learning methods.

The aforementioned methods and tools together provide the framework for analysing and visualizing vast and complex spatial tissue structures. These developments in understanding the spatial information of tissue in different diseases pave the way for new discoveries and improving the treatment for patients.

Keywords: Image processing, data analysis, machine learning, visualization, microscopy

Leslie Solorzano, Department of Information Technology, Division of Visual Information and Interaction, Box 337, Uppsala University, SE-751 05 Uppsala, Sweden.

ISSN 1651-6214
ISBN 978-91-513-1173-9
urn:nbn:se:uu:diva-438775 (http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-438775)

Dedicated to all women whose efforts led to our freedoms and rights
and to all the women who will come after.

List of papers

This thesis is based on the following papers, which are referred to in the text by their Roman numerals.

I **Solorzano, L.**, Partel, G., Wählby, C. "TissUUmaps: Interactive visualization of large-scale spatial gene expression and tissue morphology data", *Bioinformatics OUP*, August 1, 2020.

II **Solorzano, L.**, Pereira, C., Martins, D., Almeida, R., Carneiro, F., Almeida, G., Oliveira, C., Wählby, C. "Towards automatic protein co-expression quantification in immunohistochemical TMA slides", *IEEE Journal of Biomedical and Health Informatics*, July 13, 2020.

III **Solorzano, L.** Almeida, G., Mesquita, B., Martins, D., Oliveira, C., Wählby, C. "Whole slide image registration for the study of tumor heterogeneity", *Computational Pathology and Ophthalmic Medical Image Analysis. COMPAY 2018*, published in *Lecture Notes in Computer Science book series (LNCS, volume 11039)*, September 14, 2018.

IV **Solorzano, L.** Wik, L., Bontell. T.O., Klemm. A. H. , Wang, Y., Öfverstedt, J., Jakola, A.S., Östman A., Wählby, C. "Machine learning for cell classification and neighborhood analysis in glioma tissue", *under review*, February, 2021.

Reprints of papers I,II and IV were made without the need for permission since they are published under an open-access licence. Reprint of paper III was made with permission from the publishers.

For paper I, Solorzano, L. is the main contributor to the software development and architecture. Solorzano wrote the paper with inputs from the co-authors.

For paper II, Solorzano, L. is the main contributor to the image analysis methods and software development. Solorzano wrote the paper with inputs from the co-authors.

For paper III, Solorzano, L. is the main contributor to the image analysis methods and software development. Solorzano wrote the paper with inputs from the co-authors.

For paper IV, Solorzano, L. is the main contributor to the image analysis methods, machine learning experiment design, software development and visualization. Solorzano wrote the paper with inputs from the co-authors.

Related work

In addition to the papers included in this thesis, the author has also written or contributed to the following publications:

R1 Partel, G., Hilscher, M. M., Milli, G., **Solorzano, L.**, Klemm, A. H., Nilsson, M., Wählby, C. "Automated identification of the mouse brain's spatial compartments from in situ sequencing data", *BMC Biology*, October 19, 2020.

R2 Ström, P., Kartasalo, K., Olsson, H., **Solorzano, L.,** Delahunt, B., Berney, D.M., Bostwick, D.G., Evans, A.J., Grignon, D.J., Humphrey, P.A., Iczkowski, K.A., Kench, J.G., Kristiansen, G., vanderKwast, T.H., Leite, K.R.M., McKenney, J.K., Oxley, J., Pan, C.-Chen , Samaratunga, H., Srigley, J.R., Takahashi, H., Tsuzuki, T., Varma, M., Zhou, M., Lindberg, J., Bergström, C., Ruusuvuori, P., Wählby, C., Grönberg, H., Rantalainen, M., Egevad, L., Eklund, M. "Artificial intelligence for diagnosis and grading of prostate cancer in biopsies: a population-based, diagnostic study", *The Lancet Oncology*, January 8, 2020.

R3 Gupta, A., Harrison, P., Wieslander, H., Pielawski, N., Kartasalo, K., Partel, G., **Solorzano, L.**, Suveer, A., Klemm, A., Spjuth, O., Sintorn, I., Wählby, C., "Deep learning in image cytometry: a review", *Cytometry Part A*, December 19, 2018.

During the course of this research, the author has also contributed to the following pre-publications and conferences

P1 Andersson, A., Partel. G., **Solorzano, L.**, Wählby, C. "Transcriptome-Supervised Classification of Tissue Morphology Using Deep Learning", *IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, April 2020.

P2 Bombrun. M., Ranefall. P., Lindblad. J., Allalou. A., Partel. G., **Solorzano, L.** Qian. X., Nilsson. M., Wählby, C. "Decoding gene expression in 2D and 3D", *Scandinavian Conference on Image Analysis (SCIA) 2017*, published in *Lecture Notes in Computer Science book series (LNCS, volume 10270)*, June 12, 2017.

Contents

1. Background

> The most exciting phrase to hear in science, the one that heralds the most discoveries, is not "Eureka!" but "That's funny..."
>
> ISAAC ASIMOV
> *Prolific writer, scientist, professor, humanist*

1.1 Motivation

Humans are curious creatures and like to see patterns and understand the way the world works. Humanity works on mapping the cosmos, mapping planet earth (and more recently Mars as well), and also on mapping biological tissue and cells. Once mapped, relationships between these entities can also be studied and so can their behaviors and their influence in the progression and function of disease. One of the most important tools for that purpose is digital image analysis along with data analysis and data visualization. Understanding how diseases work and how to stop them is key and is today being used for personalized treatments, for example targeted cancer therapy. The myriad of techniques for imaging, methods for analysis and mathematical models has made it possible to study the same phenomenon from different points of view, it is important to be able to combine the study from a single method with many others. This is only possible today thanks to advanced technology and efficient use of computational resources to help reduce the load and increase the throughput and confidence in the results.

This thesis is the summary of an interdisciplinary work with the goal of helping researchers in the medical and biological fields to answer questions in their domains through the analysis of digital images and the data that can be extracted from them. This can be achieved with computer science which allows the understanding of mathematical models and their efficient implementation and also provides the tools for creating software where the exploration, processing and visualization of data is possible. [1].

In Figure 1.1 I wanted to show a summary of all the steps in an image and data analysis project applied to biology. I trust I manage to convey how they all play a part on gaining insight and knowledge. Also how every project and question requires the integration with other projects and sources of data that came before and have gone through the same process.

1.2 Digital image analysis

Digital images are pictures or photographs captured by electronic sensors. In digital cameras tiny sensors detect and measure light. This measurement is turned into color or intensity that will be associated with one pixel. Digital images are composed of many of such pixels. Digital cameras such as the ones in mobile phones and even more advanced ones, are coupled to microscopes which allow to record the observed micro-scenes, akin to observing stars through telescopes or buildings and roads in satellite imagery, a plethora of objects, be it stars, buildings or cells, are organized and working together to perform different activities with different goals.

In all of these images, a demarcation of objects of interest is usually performed and several features or characteristics are computed from them, for example finding which pixels compose different instances of cells, and getting information about size, shape, color, location, individual and relative to others. Tissue is composed of many such cells, and their spatial organization can inform about the location or the status of a disease. Another example is to find gene expression in an image, where small objects within a cell inform of the genetic profile which characterizes a cell and its function.

1.3 Data analysis

In order to make sense of the information gathered from many sources about an event or disease, there needs to be a workflow to analyze, classify, organize and represent the knowledge that can be learned from it. Samples can be organized by the types of their cells or even by the types of interactions between their cells. Cells can be classified by their properties, and such properties can come from many sources and in many shapes, 2D points, 3D points, ND points, connected points, disconnected points, grids etc.

Data analysis encompasses methods, and efficient algorithms to use statistics and mathematical modelling to extract understand the information extracted from the images. Methods can range from simple (but not less interesting) to intricate, from linear regression to deep neural networks. The goal of data analysis is to be able to categorize or model the different features of tissue, cells and gene expression. Data analysis also includes the models of transformations between all the different data modalities. Microbiologist works in micrometers and nanometers and image analysts work in pixels, data and image analysis include ways to transform from one space into the other.

1.4 Machine learning and deep learning

Some of the most commonly used methods today are machine learning (ML) methods. Contrary to the popular belief, these methods are not new. Already in 1958 the perceptron was proposed [2] as a model for supervised learning. But with the advent of more powerful computers these methods have also developed into enormous and complicated models.

ML is basically a form of applied statistic which takes advantage of the recent computational tools to statistically estimate complicated functions. ML has the challenge of fitting models to the data and the challenge of finding patterns that generalize to new data. Unlike pure statistic, it started with little emphasis on proving confidence intervals around the models and functions but recently a whole field has been trying to combat this, it is called "explainable AI", which develops methods to unveil the ML *blackbox*.

Most ML algorithms can be divided into two categories: supervised and unsupervised learning. At the heart of ML is optimization, and the most used algorithm is called stochastic gradient descent. Borrowing from optimization, ML combines optimization algorithms, a cost function, a model and a dataset. Through time, more data, more sources and more complex questions brought to light the limitations of traditional ML to generalize to new scenarios. These challenges motivated the development of deep learning (DL) algorithms.

ML and DL can be used, for example, to detect objects in images, to characterize those objects or to classify them. These methods can be used in ND disconnected points, or in graphs or in grids, such as images. It has become a standard component of data analysis.

1.5 Quantitative microscopy

Digital image analysis has offered unique opportunities in the field of microscopy. Observations made in microscopes can be recorded and stored digitally to be observed at any time without the need to store the biological sample. In these images a number of measurements can be extracted, allowing the observation of detailed parts of cells, detailed structures in tissue and even molecular structures, all ranging from micrometers to nanometers.

Being able to analyze, categorize and extract knowledge from biological samples gave rise to digital pathology where the nature and mechanisms of disease can be studied by imaging samples of biopsies and providing insights at a very fast pace.

Quantitative microscopy and digital pathology have now advanced at an amazing speed, incorporating fast and efficient computation strategies, use of ML and DL and advanced autonomous scanners with hardware to perform chemical staining procedures and handling optics, lenses and lasers which are needed to capture the images.

1.6 Combining multiple sources of information

A single event or object can be observed with different kinds of microscopy, in different resolutions and different times. For example, the type of a cell can be determined by knowing the proteins it expresses, the genes that are active in it, its shape, its location and its neighbors. All of that information comes from different sources which work in different dimensions, units, types and fields of view. During the course of this research, tools and methods were developed to combine these different sources of information and explore it in a single convenient point of reference.

1.7 Visualization

With the arrival of big data and high throughput experiments, images can be created by the million, and information has to be summarized and the most important aspects have to be brought to light. This is where image analysis, data analysis come in. Images can be selected, processed and summarized to then become candidates for visualization.

Results obtained after processing of images and data are key to gain knowledge. Excellent visualization is thoughtful design and presentation of interesting data, it's a matter of content, statistics and design [3]. It is achieved when complex ideas can be communicated with clarity, precision and efficiency. A visualization of an idea should give the viewer the greatest number of ideas in the shortest time with the least amount of objects and the smallest space. Graphical excellence is nearly always multivariate and graphical excellence requires telling the truth about the data.

In more recent times, with the available computational infrastructure, GPUs, and affordable advanced technology, data can (and is almost expected to) be explored interactively, where a user can continuously input parameters and a more localized view of the results can be obtained. Such exploration is useful to gain further knowledge and highlight new ideas of previously unseen relationships in a field, like biology.

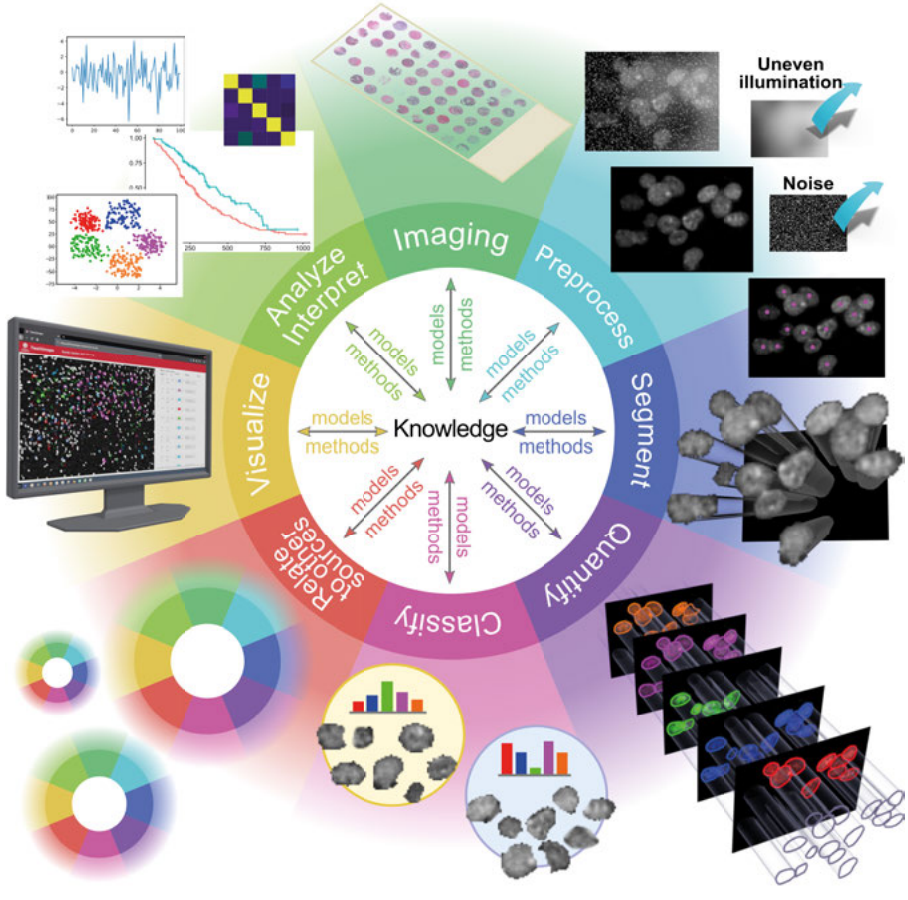

Figure 1.1. Image processing and data analysis. Knowledge can be derived through methods and models applied to images and data. Objects and regions can be segmented, quantified and classified. If more data coming from other sources is available it can be brought together. Such data has usually undergone a similar process of discovery. The results can be visualized on top of the original image data and data can be explored, analyzed and modeled thus answering a question, furthering the knowledge that allows for posing newer more relevant questions. Here an example is shown. When a question arises, tissue samples can be collected and arranged on a glass slide and stained. The image depicts small pieces of tissue arranged in a micro array. Each piece can contain thousands of cells. Cell nuclei are segmented in fluorescence microscopy images and associated biomarkers are quantified within the area marked by the nuclei and this quantification serves as a set of features to classify the nuclei. Once a class is defined it could be related to other data such as gene expression expected to appear only in specific classes or regions, or perhaps the nuclei can be observed under a different kind of microscopy. Once all information is gathered it can be observed, validated and interpreted to answer a biological question or describe a phenomenon or mechanisms of disease. New insights can inspire new questions and an even more targeted imaging can be done and the process can begin again.

2. Quantitative microscopy

There are two main kinds of microscopy, electron and optical. I will focus on optical which uses illumination and lenses to direct and filter light. Biological samples are usually transparent which means they have to be prepared and stained in order to be visible before going into the microscope or scanner. Different microscopy techniques require very specialized staining and chemistry before they are ready to be imaged. Microscopy images that contain a micrometer thin slice of a full piece of tissue mounted on a glass slide are usually referred to as Whole Slide Images (WSI). Other images contain arrays of smaller circular pieces of tissue resulting from punched out pieces from tissue collected and arranged in a paraffin block, and then sliced, mounted on glass and stained. These images are called Tissue Microarrays (TMA). In this chapter I present the type of microscopy used to capture the images in the datasets for the different projects I was involved in and the numerous challenges presented when working with microscopy images.

2.1 Microscopy techniques

2.1.1 Fluorescence microscopy

To capture an image in fluorescence microscopy, a sample has to be stained. Occasionally the imaged sample can contain self-fluorescent materials. Once the stain has attached to its intended target the sample is illuminated and excited fluorescent molecules (fluorophores) will emit light that is captured by the sensors that create the digital image [4]. Using different filters for different wavelengths, each stain can be captured in a different image channel. The most frequently used stain is that of nuclei, generally obtained using 4'6-diamidino-2-phenylindole (DAPI). Images of this stain is used to detect cell nuclei which are later associated with complete cell objects [5]. The area outlined by the nucleus and cell objects is used to extract the rest of the cell features in all the other channels of an image.

Amongst fluorescence microscopy there are many subgroups related to the particular molecules to which stains attach to, or the kind of light used to excite them, or the kind of chemical treatment used, or the pathways the light follows to image the sample. Multiplexed immunofluorescence (mIF or mIHC) is a kind of preparation of a sample so that using fluorescence microscopy an image with information on multiple molecular markers can be

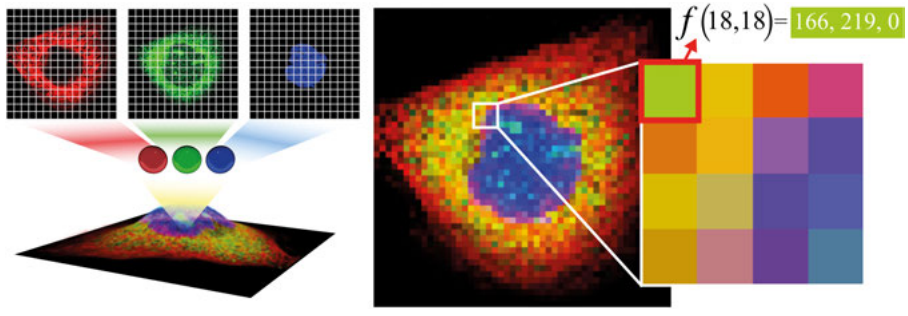

Figure 2.1. Illustration of the process of image acquisition in fluorescence microscopy. A sample, cell or tissue is lit with a light source, the reflected or emitted light passes through band filters to separate into different channels (3 are depicted but there can be several). Each channel is grayscale but can be pseudo colored for visualization purposes. When combined using RGB space, the full cell can be observed. Each pixel x contains the channel intensities $f(x)$

obtained. A fluorophore is attached to an antibody (hence the immuno-) and this antibody targets a specific protein of interest. Wherever this protein is present, the fluorophore attached will light up when excited and the result is an image where the location of the protein is mapped [6, 7]. Using mIF, up to eight different proteins can be imaged in the same sample, and the goal is to observe and characterize cells by the combination of proteins they express. WSI and TMA contain from thousands to hundreds of thousands of cells. We work with mIF in paper IV. Recently a technique called co-detection by indexing (CODEX)[8] allows the simultaneous study of up to fifty biomarkers at the cost of being low throughput. This means that it is possible to quantify up to fifty markers for every cell.

Tissue and cells can be characterized by protein expression but also by gene expression. Using specialized chemistry and fluorescence microscopy, single mRNA molecules can be detected, amplified and sequenced directly in the tissue, preserving the natural histological context, and allowing the connection of the mRNA sequence with spatial information. This method is called *in situ* sequencing [9]. Gene expression images can be usually accompanied by a DAPI stain channel where it is possible to associate individual gene expression to a specific nucleus. Specific combinations of RNA molecules can characterize a cell or indicate about its behavior or membership to a particular class. Millions of data points arise as a result of *in situ* gene expression mapping.

Hundreds of thousands of cells and millions of gene expression points are part of the data that is used for quantitative analysis of tissue behavior in conditions of disease and for survival studies. They are big data and require special image and data analysis techniques, along with specialized software to explore and visualize. We use gene expression data in paper I.

2.1.2 Brightfield microscopy

In this type of microscopy, tissue is illuminated using a white light source, commonly a halogen lamp in the microscope stand. Unstained tissue will appear as transparent (background color). For this reason the tissue has to be sliced into thin sections, fixated and then stained to display the desired object. Thin sections are required so that light can shine through, to have depth of focus and for the stain to reach and bind to its target. The transmitted light will result in a colored image that can be viewed or captured with a digital camera. The specimen is illuminated from below and observed from above.

Two of the most common staining techniques are called Haematoxylin Eosine (H&E) and Haematoxylin Diaminobenzidine (HDAB). In H&E, acidic cellular components react with a basic stain, H, while basic components react with an acidic stain E. H&E staining is not specific to a molecule, it is used to reveal the distribution of cells and of elements within cells. On the other hand, in immunohistochemistry (IHC), DAB staining is used to reveal the existence and localization of a target protein in the context of different cell types, biological states, or sub-cellular localization within complex tissues. [10].Brightfield microscopy, H&E and HDAB are the most common techniques by pathologists to observe the morphology of tissue, distribution of cells and structures and characteristics of cells that are indicatives of specific diseases an behaviours [11]. We use IHC HDAB images in papers II and III.

2.1.3 Multiplexed analysis

Fluorescence, brightfield and other kinds of microscopy provide different kinds of information that, when combined, reveal new information that might not be possible to obtain with a single technique. H&E and IHC are more commonly used to characterize a disease through the texture and intensities which reveals the overall scaffold and structure of the tissue. Fluorescence can indicate the location and amount of more than one marker of interest at once, thus giving clues on what proteins may interact or be involved in the same cellular (or sub cellular) process. When using fluorescence data overlaid on brightfield data it provides insight on the mechanisms of the activity that is happening in the areas detected by H&E [12], for example, pathologists can detect tumor areas in H&E images and the proteins can be quantified in fluorescence images. More recently overlaid fluorescence data has been used as a label for supervised machine learning methods as opposed to labeling the H&E stained images manually [13]. Additionally new exploratory methods translate H&E staining to mIF or to IHC [14], and from mIF imaging, pseudo IHC or H&E colored images can be created to allow the transfer of knowledge from traditional histopathology to fluorescence imaging [15]. Figure 2.2 illustrates the different kinds of microscopy used in

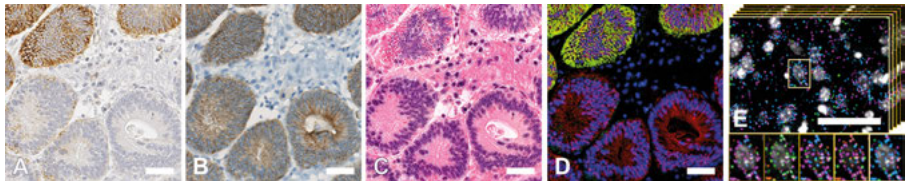

Figure 2.2. Illustration of a piece of tissue under different microscopy modalities and stainings. A and B are IHC stained with HDAB. A) and B) are two different images from two different but consecutive tissue slices each targeting a different protein by IHC, therefore there are two different patterns C) is a synthetic recreation of H&E staining, the darker purple spots are nuclei. D) is a synthetic recreation of fluorescent staining, blue corresponds to nuclei, and red and green for the same proteins in A and B. In the studies of coexpression of multiple markers it is preferable to have mIF but when there is no access to it, several IHC can be performed on consecutive slices. E) shows nuclei in white and their accompanying gene expression. The small dotted colors represent the first base of an mRNA sequence (ACGT). One cell is displayed with all its bases. This is an example of bringing together different images to extract information

this thesis. It shows a synthetic recreation of an H&E and mIF view from real IHC images to depict how the same tissue would look under different modalities. Additionally in (E) it shows a view of in situ sequencing gene expression. The nuclei can be observed in white and are physically as big as the nuclei represented in blue dots in fluorescence (D). The scale bars indicate that the view of in situ sequencing is of higher magnification than A,B,C and D.

2.2 Challenges of using microscopy data

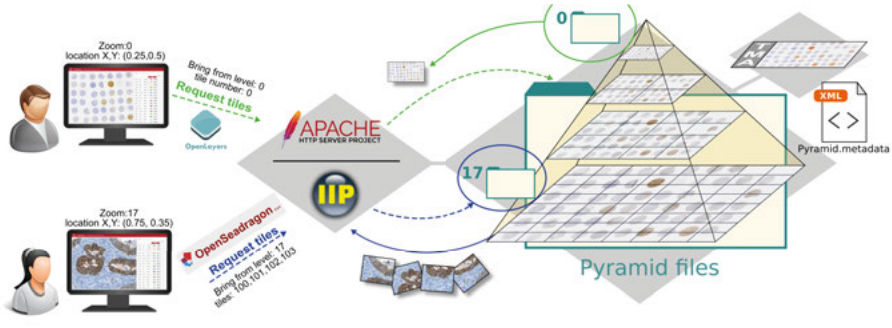

Figure 2.3. Process of loading and viewing a TMA. For a particular location and zoom level a request is made for the tiles that cover the desired view. A server software (local or remote) receives the request and procures the tile either by creating a new image or loading an existing one.

Despite depicting a relatively small piece of physical biological tissue, microscopy images are obtained at a very high resolution that allows us to observe individual cells to a high level of detail. This results in one of the major challenges of working with microscopy: the size of the image. While the image from a phone can be on average 2000x1500 pixels, images in microscopy can range from 5 to 50 times that, resulting in 25 to 2500 times the amount of pixels, which is why they are referred to as gigapixel images.

WSI and TMA images can be a single channel or a multichannel or multispectral image, where several wavelengths can be represented. To address the size problem, images are saved in different levels of detail, different fields of view and different image compression of the same sample in a pyramidal scheme[16], resulting in a file that contains images of different sizes inside in the form of tiles. This solves the problem of structure of data but introduces a new problem: hundreds of different formats created by all microscope vendors. Science is gradually becoming more open [17] and formats are being standardized [18, 19] and more specialized libraries are now able to load big images into memory. Taking advantage of the pyramidal scheme, microscopy images become distributed data, it can be streamed or served, so that it doesn't have to be stored locally. For that purpose various servers were created to serve microscopy tiles belonging to one of several levels of the pyramid. This also means that specialized processing is required for this data since it has to be done either in downsampled levels in the pyramid or in tiled high resolution patches or in a multilevel fashion.

Challenges are several and interconnected and anybody interfacing with a system that produces this kind of data has to, undoubtedly, make design choices when performing any kind of research project involving microscopy. Given the image size a researcher can choose to store images locally or in a specialized repository. Given the microscope, only certain formats and readers (and sometimes writers) are available. For processing needs, the researcher must decide if the pyramid should be accessed in an offline fashion: the pyramid will be stored in separate tiles for quick access at the cost of space (and cost of transfer if served), or online: if the pyramids should be stored compressed to save space at the cost of creating the tiles every time they are needed and served.

Images inside the pyramids usually contain known file formats and algorithms like jpg, png, tiff, jpeg2000, all with different parameters. There are also private ones. Most of the time pyramids can be read but not stored if unlicensed meaning that after processing a pyramid has to be created in a different open format.

Libraries that have been created to deal with these specific biological pyramidal images in local environments are OpenSlide[20], Bio-formats[21], slideio [22]. Libraries that are able to open pyramids of any kind are VIPS[23] and BigTiff/Libtiff. They are mostly developed in C, C++, Python and Java programming languages.

To serve images, whether they are stored locally or stored in the cloud, software like the IIPImage server was created by the International Image Interoperability Framework [24]. Web servers like Apache can receive http requests for tiles.

Online viewers that can read local and IIIF served images are OpenSeadragon[25], Zoomify[26], and OpenLayers[27].

To give an idea of the image sizes that microscopy data uses, the smallest size displayed so far in TissUUmaps (the visualization software developed during the course of this thesis) is of 9000 times 6000 pixels. The biggest is 110000 times 100000 pixels. In comparison, most mobile phones and hand held devices used by the general public today reach 4000 times 3000 pixels.

3. Image analysis

In digital images as the ones coming from microscopes, objects of interest can be detected as specific ranges in the signal. Signals can be unmixed into components. Patterns can be found in them, which represent distinctive tissue characteristics. This approach can be used to obtain meaningful information in fields like microscopy. To achieve this goal, in this thesis, workflows have been developed for object detection, object classification and image description. This chapter will present a recapitulation of the necessary concepts to achieve the results presented in the published works and a few related methods for reference.

3.1 Digital images

Digital images are a finite sized grid of elements commonly referred to as pixels. While the grid doesn't necessarily have to be rectangular it is its most common shape and is the shape of the images used in this thesis. An image can be typically captured using an optical system with lenses that receive a continuous light signal and a sensor samples it to create a digital reproduction of the observation.

More formally, the digital image can be seen as a discrete vector valued function $f(\mathbf{x})$ where $\mathbf{x}$ is an ND spatial coordinate (mostly 2D in this thesis). In that location the vector assigned to it contains the values of each captured channel.

3.2 Image segmentation

3.2.1 Object segmentation

Segmentation can be described as pixel classification. In order to perform object detection in an image, a rule or set of rules is applied to find a set of pixels that corresponds to a specific object leaving out the pixels that correspond to background, separating the pixels into groups. In microscopy, the most common object to be detected is the nucleus of a cell [5]. Nuclei usually have a distinct ellipsoidal shape and distinct pixel intensity levels which allow them to be distinguished from non-nuclei objects. For segmentation, texture and image features are used. To differentiate objects

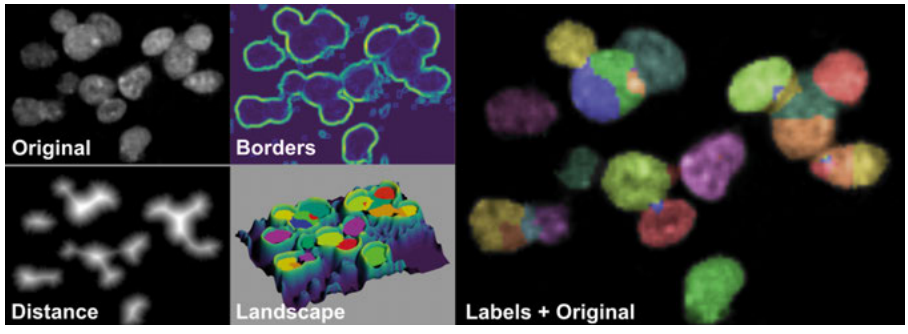

Figure 3.1. Illustration of watershed for nuclei segmentation. From the original image, edges can be detected with a sobel filter. Then using the distance transform, the peaks (or basins) can be found and then watershed is done on the edge image using the basins as starting points.

within a segmentation, shape and other features can be taken into account. We use nuclei detection and segmentation on paper IV.

Amongst the most common methods used for nuclei detection is thresholding, edge detection, watershed segmentation [28] and graph cuts. There are methods that make use of manual labels provided by a human annotator. The most popular are machine learning methods such as Random Forests (RF)[29] and convolutional neural networks (CNN). In recent years, one of the most used CNN architectures for nuclei detection is the U-net architecture [30] and derived architectures like Tiramisu [31]. Figure 4.1 in the next chapter shows a basic architecture of a CNN.

Objects can be detected by their contours. Contour or edge detection algorithms [32] include Laplacian of Gaussian (LoG), Canny edge detection, Sobel and Prewitt filters. Once borders are detected, closed or nearly closed borders can be filled and connected components can be considered objects. If the object intensities in the image do not vary greatly, thresholding can also be a way to binarize an image and connected components of specific sizes can be deemed objects.

Watershed algorithms, conceptually, treat the image intensities as a landscape, which when filled with water would create "puddles" that correspond to connected components which can be object candidates. Local minima are treated as locations where water can spring and start filling up the valley. For paper IV we use nuclei objects segmented using watershed as implemented in the popular digital pathology software QuPath [33].

Machine learning methods for object detection perform a study of the pixels under given labels and create a classifier that indicates whether a pixel belongs to a specific object or to background. The most commonly used method which is included as an option in most of the clinically used software [34, 33] is RF, which takes as input many features coming from each pixel and its neighboring pixels, mainly intensity, edge and textures. Such features

serve to characterize a pixel as belonging to a particular class and RFs are able to distinguish them, resulting in the separation of objects' pixels from background. Texture descriptors which are still commonly used in this area are, Gabor filters, edge filter, and filters in the frequency domain like filtering by band, Haan windows and wavelet filters. Classical image analysis methods applied to microscopy are reviewed here [12]. More modern methods like CNNs (described in more detail in chapter 4) learn their own unique features that allow them to classify and separate the pixels. The inputs are small portions of the images referred to as patches, which undergo a series of convolutions with matrices that contain the features. These features change as the network learns by means of back-propagating errors using gradient descent.

3.2.2 Region segmentation

Segmentation is not only used to find objects, but also bigger regions of interest. The search for such regions focuses on texture instead of the shape of a particular object. All the methods explained previously, apply for region segmentation. In microscopy, image segmentation is, for example, focused on separating regions of diseased tissue from healthy tissue, or finding the boundary between them. It is also used to find specific types of tissue and distinguish them by the organ or location they come from. If a whole image contains a single region, textures measures can serve as a features to classify the image as belonging to a single class and separating it from other images in a dataset. In paper II we used image segmentation to distinguish cancer regions and quantify only within such regions.

3.3 Image registration

Images of a single sample can be taken in different modalities, or the same modality but at different times and conditions. Sometimes, in order to create a large image in high resolution it is necessary to obtain smaller overlapping fields of view that need to be *stitched* together. In any case, multiple views of a sample can contribute to additional information and they need to be brought to the same spatial frame of reference. The process of aligning images is called image registration. Its specific use in medical images is surveyed in [35].

The transformation can have various degrees of freedom. The simplest one is called rigid, when it only requires translation, rotation and isotropic scaling (same scaling in all dimensions), such transformations preserve distances within the image and preserve parallel lines. When more distortion is required such as shear, the transformation is called affine, it preserves parallel lines but does not necessarily preserve distances. When the deformation goes in different directions and magnitudes along the image the registration

transformation is called deformable/elastic/non-rigid and the most commonly used kind is B-spline [36, 37].

Image registration is expressed as an optimization problem that is solved by iteratively searching for the parameters θ of a transformation T that transforms an image A (moving) into the reference space of image B (fixed). The best alignment is decided based on a distance measure d between the reference image and the moving image. Registration can then be defined as:

$$\arg\min_{\theta} d(T(A,\theta),B) \tag{3.1}$$

Image registration can be feature based or intensity based. Feature based means that several matching points have to be found in the images and then a transformation that is able to minimize the distance between these points. Intensity based methods use pixel intensities to guide the registration. There are a few kinds that include both features and intensities, such as [38] which is used in paper II to find affine transformations between cores in a TMA. Using a single transformation for a whole image does not take into account possible disconnected objects of interest that *move* and transform independently. In such case, piecewise transformations can be used, this can be useful for very large image such as those in microscopy. In paper III the size of image, the disconnected pieces and the image artifacts called for a piecewise affine registration using feature based registration. While features can be any of the ones presented in previous chapters, in this case features invariant to scale called SIFT [39] were used to find the parameters of the affine transformation.

3.4 Color transformation and unmixing

The colors we see in images in computer screens are usually the combination of several grayscale channels. The most commonly known one is RGB (red,green,blue). Most color images are stored in such an RGB format. When we open the image using a viewer the channels are combined and the result are many different hues. As seen in previous chapters microscopy images come from brightfield and fluorescence microscopy and the colors they have vary depending on the scanner, the stain, the camera settings and many more factors.

When a method is designed to work for a particular colored dataset, it seldom works on different colored data if unchanged. For this reason, it is desirable to transform the color of the images into a common space where a single method can work. This can be done with methods as simple as linear transformation of the color channels or with complex deep learning tools like generative adversarial networks (GAN) and U-Net architectures [40, 41].

To extract a color from an image, i.e. separate stains into channels other than RGB, say, H&E or HDAB, it is necessary to know a color base. Since

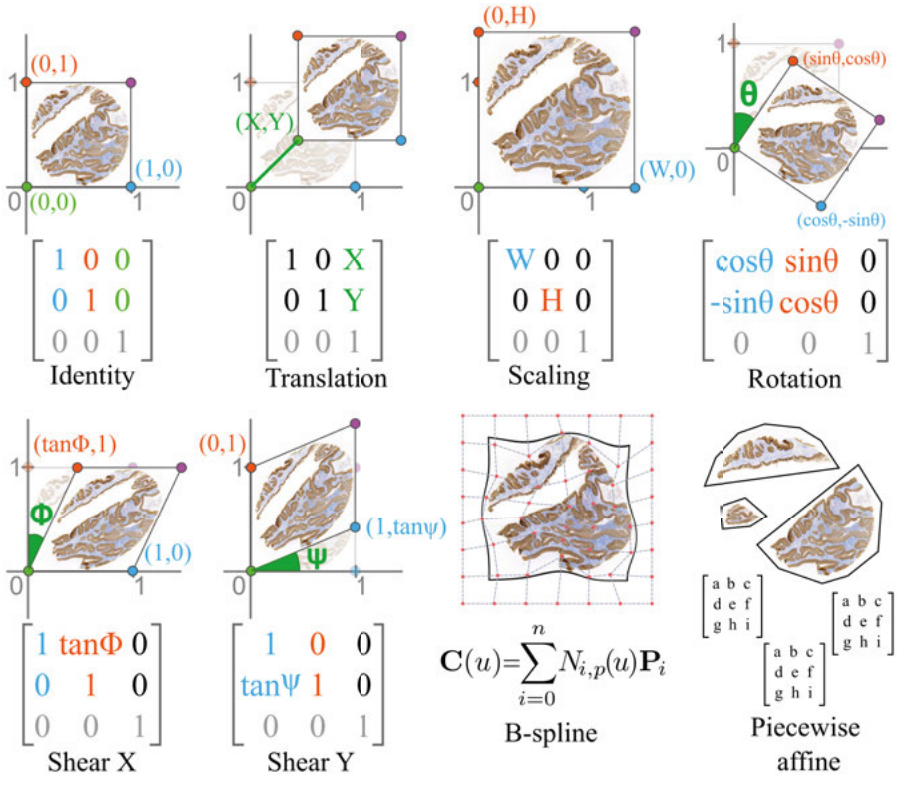

Figure 3.2. Examples of registration. Affine matrices contain 9 values in 2D out of which 6 represent a linear transformation of the X and Y coordinates. Affine transformations preserve parallel lines as observed in the squares surrounding the tissue. B-spline transformations require a set of control points and knot points. Piecewise affine transformations require the polygon around the piece and its corresponding affine transformation matrix.

RGB is a 3 channel color then a 3D base can be found from a reference image in which 3 vectors can separate the image into 3 channels different from RGB that are a linear combination of them. We use color transformations to transform RGB colored images to HDAB in papers III and II.

3.5 Open source software tools for image analysis

This is a short section where I describe the libraries and software that are building blocks in image analysis and that are close to the mathematical models that will be discussed in the thesis. While there is many software out there built as a tool for the use of image analysis for the general public, I will mention the key players that still to this day continue to develop methods and

make them available through their libraries and software under open source licences.

In 2007, when I started my journey in image analysis I was introduced to C++ programming language for image analysis and computer graphics. The python programming language was still not big in image or data analysis. Mainly the open source software for image analysis I was aware of in C++ was ITK [42] which started in 1999 but was released in 2002. It is still widely used and is the backbone of specialized image processing libraries. At the same time, at Intel Research, the Open Computer Vision (OpenCV) [43] library was formed. In 2003 based on ITK, Elastix[44] introduced a general framework for image registration.

More specialized libraries and software for microscopy start to be built. Joining the race since 1997, the National Instutes of Health (NIH) provide grants to different groups work on C++ and Java to create an image analysis software that would become ImageJ [45] and FIJI [46] which is one of the most popular software for image analysis in life sciences. The NIH also develops Bio-Formats+[18] which come out 2006.

Using C and C++ *under the hood*, building on work in numerical methods and a library, the library Numpy [47] comes to the world in 2006 making it easier to work with multidimensional arrays such as images. The now ubiquitous python's scikit-learn (2007) [48] and scikit-image (2009) [49] come to the world for scientific machine learning and image processing. Bindings in python for OpenCV and Elastix are developed.

In 2005 CellProfiler [50] by the Broad Institute comes out and is specially designed for quantitative microscopy image analysis. In 2008 CellProfiler Analyst [51] comes out for the use of more advances machine learning methods for cells classification. For segmentation, in 2011, Ilastik [52] offers machine learning methods such as random forests to perform image segmentation with an interactive GUI.

With the improved computational power deep learning came back into the mix and libraries like Torch are ported into python creating Pytorch [53] in 2016, and while it is a machine learning library it contains its own matrix representation, tensors. In 2018 pytorch-geometric [54] comes out for working with deep learning for graphs.

I can't possibly name all the software that I have encountered during this research but I consider these to be corner stones for image analysis and it's use in microscopy (not exclusively). There is software more interested in the application of methods rather than on the development, so it's not mentioned in this section but rather in context in the next sections.

4. Quantitative Data Analysis

To gain understanding on a phenomenon we measure and collect data, be it numerical or pictorial. Any kind of measurement includes noise, so data is first cleaned, or denoised. Then it can be inspected in an exploratory phase where statistical modelling can be of use. It helps to understand the scope, the range in which data is present.

Data can be used not only for exploration but also for analysis and prediction. Models can be created from the data and their errors can be assessed so that the best available models can be used to associate features to specific points. These models can offer insights as to what characterizes the data and allows us to separate it into different and useful categories. The models can also be used to predict new information and to make decisions.

One of the most important uses of data analysis today is the combination of multiple sources of information. It is often not enough to analyze a single source but to bring information of multiple experiments designed in different ways and performed at different times under different conditions and producing different kinds of output.

During the research leading to this thesis, data was extracted from microscopy images so that tissues and cells could be characterized. The data was analyzed so that tissues could be assigned, for example, an origin, a grade, an amount of cells, an amount of pixels and more. Cell can have several features computed, such as size, intensities, textures, classes and more. All this information can come from different sources, such as different images and from expert observations, such as those from pathologists. Observations from experts can be used as labels for further processing by e.g. machine learning.

4.1 Classification

Given a set of observations with a set of features and an associated label, we want to learn how to separate them into categories so that a new observation can be given a label corresponding to one such category based on the features associated to it. For example, is it a cancer cell or not? or which cell type does a cell resemble? is it a neuron or is it a macrophage? Can I discriminate (distinguish) them with the information I have or do I have to collect more information and use more sources? This type of problem is called supervised learning.

Classification produces a function (or model) that assigns a category to an input commonly described by a vector which contains independent variables, called the feature vector. In a probabilistic setting, the model can also assign probabilities instead of a single class prediction. To train a classifier, a set of training data with labels for each of the observation should be available. The most basic model, which also serves as a building block in more advanced models for classification, is a linear model for binary classification (meaning belonging to a class or not). In a linear model a prediction $\hat{y}$ is found from feature vector input x of length n and a weighted linear combination of its components. The weights, or parameters θ is what we want to find. A binary linear classifier can be expressed as

$$\hat{y} = \sum_{j=1}^{n} \theta_j x_j. \tag{4.1}$$

To find θ, the classification problem is posed as an optimization problem with an objective function J

$$J(\theta) = L(\theta) + \Omega(\theta), \tag{4.2}$$

where L is a loss function and Ω is a regularization function. The loss measures how similar the predicted labels are to the original labels, while the regularization function, as its name suggests, regulates the values for θ by penalizing the appearance of undesired properties to try to avoid overfitting to a particular dataset.

However if we look closely at equation 4.1 what the model is doing is not yet exactly classification[55], but regression. We are approximating the values of y. This means that if our labels are a vector of 0 and 1, $\hat{y}$ will still be penalized if it is far away despite being in *the right direction*, for example having predicted a 3 instead of a 1 or a -2 instead of a 0. To find the final class prediction one can threshold $\hat{y}$ at $\frac{1}{2}$.

4.2 Unsupervised learning

4.2.1 Clustering

When there are no available labels for a dataset, it can still be separated into classes by clustering, which is a type of unsupervised (without labels) learning. Clustering works on the basis of dividing observations into groups that share similar features, or even groups them by the neighbors that an observation has rather than its inherent features. Clustering will also yield a model so that any new input can be assigned a class based on its similarity to the divisions found. A really good review and implementation of classical methods for clustering is available in [56, 48] and a general overview can be found in [57].

There is no one-size-fits-all clustering method. There are methods that require as a parameter the number of clusters and others that attempt to discover the number of clusters. Additionally, every method makes assumptions about the data. If data is not distributed according to the assumptions, the model will behave unexpectedly. Since clustering is based on the similarity between observations, this similarity can be almost any pairwise distance measure. Usual measures are euclidean and cosine, it depends on the manifold the data lies on.

The most common clustering method that requires specifying the number of clusters is k-means. Its goal is to divide a set of observations into k separate clusters C characterized by their centroids. K-means minimizes the euclidean distances from every observation to each cluster in C. The observation will be assigned the label of the cluster whose mean it is closest to.

$$\sum_{i=0}^{n} \min_{\mu_j \in C} (\left\| x_i - \mu_j \right\|^2) \tag{4.3}$$

But then again, k-means makes assumptions of data, any clustering based on k-means will result in a partition that is equivalent to a voronoi partition which is convex and therefore restricts the resulting shapes of clusters. This may often be ill-suited for actual data, which is why less restrictive methods have been devised.

Another type of clustering methods create a tree that iteratively splits the dataset into smaller subsets until each subset consists of only one observation [58]. The tree can either be created from the leaves up to the root (agglomerative) or from the root down (divisive) by merging or dividing clusters at each step. These methods require in most cases a stopping criterion which become another parameter.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [59] finds observations of high density and expands clusters from them. It was one of the first clustering methods based on density and is resistant to noise. DBSCAN is insensitive to the shape of the cluster but is sensitive to the density variation.

Mode-seeking methods estimate the density of all the observations and then identify the local density maxima which become candidates for centers of clusters. The most common method in this category is called mean-shift [60], which iteratively searches the neighborhood around candidates to cluster centers and computes a vector m that is the difference between the weighted mean of values within a kernel and the center of such kernel. This vector m is the *mean shift* vector, which goes in the direction of a region with a higher density of observations.

Clustering can be done in several data representations, be it an image or a graph. When objects like cells (or objects inside cells) have a spatial location and distances to neighbors can be quantified, the objects can be represented by a graph G with nodes V and edges E. Edges are only formed when two

points are close enough according to a chosen metric. Once data is represented as a graph, clustering becomes a matter of finding communities [61, 62]. For clustering in graphs, the Leiden algorithm [63] guarantees well connected communities. Leiden improves on the Louvain algorithm [64] which had been used for big datasets. Louvain follows a simple yet elegant set of steps to move nodes from one cluster to another to improve the quality of the clusters, and merge clusters. Without regularization, Louvain creates bridge nodes, i.e. nodes that could be their own clusters and that connect two distinct clusters. If such a node is moved, the connectivity is broken and it cannot be fixed. To solve that problem, Leiden adds a step for dividing clusters if necessary, and adds a measure of unstable nodes, so that only those nodes that are unstable are candidates for moving to other clusters. This resulted in a faster algorithm and better connected clusters.

4.2.2 Contrastive learning

Within unsupervised learning there is a kind of L function called contrastive loss, from which the triplet loss [65] allows a model to learn by showing it positive examples (examples that are similar to the input) and negative examples (examples that are different from the input). The input itself is called anchor. A triplet is made of anchor, positive and negative examples. The goal is to create a representation of the anchor that will be similar to the representations of the positive examples and will be different from the representations of the negative examples.

The method for determining what is positive and what is negative varies. One way is to use the spatial information associated with the data points. If a point is far away from the anchor it has a higher probability of being different and can be a candidate for a negative example, and points near the anchor have a higher probability of being similar making them candidates for positive examples.

In Papers IV and [66], data is represented as graphs. The representation of a point is given by contrastive learning, where a positive examples is given by a short random walk from the anchor, and the negative example is given by a long random walk from the anchor.

4.2.3 Curse of dimensionality

These methods can perform well in relatively small dimensions, i.e having a low number of features in the feature vector. The terms “curse of dimension” or “curse of dimensionality” are umbrella terms for several issues with working with high dimensions. The cost of computing an approximation is exponential in the dimension of the problem but can be compensated with smoothness or structured data [67]. As dimensionality increases, the distance

to the nearest neighbor approaches the distance to the farthest neighbor. In other words, the contrast in distances to different data points becomes nonexistent [68] thus reducing the utility of the features for discrimination. In big data analysis it is common to see attempts at approximating complex data such as images with long feature vectors that are queried in search of similarities using distances. It is common to try to reduce dimensionality to work with the same observations but in a convenient lower dimension that still manages to approximate the relationships between data points as in the higher dimensions. Such a process is called dimensionality reduction.

4.3 Dimensionality reduction

Reducing dimensions is also called projecting data or transforming data. When projecting to lower dimensional data it is desirable to keep as much information as possible from higher dimensions, which means that relative distances and densities should ideally be preserved while mapping the features of an observation to a space where it is easier to divide into groups and visualize the data.

The most comprehensive and up-to-date survey of dimensionality reduction techniques which shows and compares almost a hundred techniques and their parameters on tens of datasets is presented in [69]. It shows that to perform dimensionality reduction, a large number of things have to be considered when choosing the method, such as the amount of parameters, their sensitivity, their variance, properties they preserve and how well they preserve them and resistance to noisy data. From datasets, one should consider: the amount, the dimensionality of the feature vectors that is to be reduced, the intrinsic dimensionality (principal components that explain a given fraction of the data variance) and sparsity (the sparser, the harder to project).

Among the projections with best quality were determined in [69] to be Uniform Manifold Approximation (UMAP) [70] and t-distributed stochastic neighbor embedding (t-SNE) [71].

Both techniques try to preserve similarity in the high dimensional space when projecting to a low dimensional space. Both use graph layout algorithms to spatially organize data in low-dimensional space. In the simplest sense, they construct a high dimensional graph representation of the data, and then optimize a low-dimensional graph to be as structurally similar as possible.

In t-SNE a map is created in 2D or 3D (although the process works for any dimension) by computing how likely a point x_i is to choose a point x_j as its neighbor with a symmetrized conditional probability p_{ij}. This probability depends on a gaussian with variance σ_i centered on the point x_i. A similar probability q_{ij} is computed on the low dimensional space and to measure how faithfully q_{ij} represents p_{ij}, the Kullback-Leibler divergences are minimized

over all data points using gradient descent. Using such a cost results in a crowded map around the origin and for this q_{ij} is computed with the student t-distribution, thus giving the "t" to t-SNE. As σ_i increases, so will the entropy of P_i, the conditional probability distribution over all other data points given data-point x_i. Therefore, a parameter called perplexity is introduced which according to the authors can be interpreted as a smooth measure of the effective number of neighbors and is related to the Shannon entropy of P_i. This parameter is controversial and has been studied deeply by [72]. It was shown that the perplexity parameter is very sensitive and that the interpretation of the map must be done carefully, as cluster sizes, distances between clusters, and cluster shapes do not really mean anything in context which are expected qualities in maps.

UMAP is a new method that overcomes the challenges faced by t-SNE. It increases the speed of computation and provides a more interpretable map. UMAP, as its name says, attempts to approximate the manifold in which the data lies, which might not necessarily be $\mathbb{R}^n$. To join points, a ball is centered in every point and the radius is deemed to be the one that includes k-nearest neighbors. This way, instead of choosing a distance (like t-SNE's σ) UMAP chooses the number of neighbors. Even though distances to the k-nearest neighbors vary, these balls are all considered to have a radius of 1 in their own metric space. This way, UMAP is assuming that the points are uniformly distributed in the manifold and can take advantage of topological tools both to estimate the manifold and to obtain the features which would be projected to a low dimensional space. To achieve a similar structure as that in the high dimensional space, the cost function J to optimize is the cross entropy between the weights w_h,w_l of edges in the set of all edges E in their corresponding high (w_h) and low (w_l) dimension.

$$J = \sum_{e \in E} w_h(e) log\left(\frac{w_h(e)}{w_l(e)}\right) + (1 - w_h(e)) log\left(\frac{(1 - w_h(e))}{(1 - w_l(e))}\right) \tag{4.4}$$

In this way the position of a point in the low dimensional map of the data is determined by push and pull forces determined by the weights of the edges in the topological representation of the high dimensional data. This results in a map that can be interpreted; distances between points and clusters mean something in context.

4.4 Machine learning and Deep Learning

4.4.1 History

In 1646, Fermat proposed a general approach to compute local maxima and minima of a differentiable function by equating the derivative of the function to zero. Today, this approach is still included in almost all textbooks of

calculus as an application of differentiation [73]. Differentiation and the exploration of the solution landscape are basic ingredients of most machine learning (ML) and deep learning (DL) methods.

Linear models (as mentioned in section 4.1) were conceived for regression for the first time around 1894 by Galton, but was properly described by Pearson in 1930 [74]. In 1958, Rosenblatt's perceptron algorithm [2] fits a binary logistic regression model by taking equation 4.1 and doing an iterative procedure to estimate θ. In each iteration, if $\hat{y}_i$ is negative the current solution θ is modified in the direction of an approximate gradient $g_i \approx (\hat{y}_i - y_i)x_i$. This method of gradient descent (invented in 1847 by Cauchy) is the method that tricked people into believing machines could think.

There are other ways to do regression and classification that are also ML algorithms that only later adopted gradient descent. In 1963 Morgan and Sonquist created the first regression tree [29, 75]. Regression tree models fit a piece-wise-constant function by recursively splitting data into two subsets whose variance is used as a measure of *impurity*. The best split is that which minimizes variance. In 1984 the term Classification And Regression Tree (CART) is introduced by Breiman [58] and adds new rules for splitting, new measures of impurity and more. Later CARTs borrowed *boosting* from statistics, which iteratively focuses on those observations which are difficult to classify (or separate or model). After boosting came gradient boosting which combined all the advances in trees with the improvement using gradients of a cost function, the first of such cost functions will look familiar: $\frac{1}{2}(\hat{y}_i - y_i)^2$.

For a long time ML was used in many applications and laying the ground for DL (which now has almost replaced ML in hard tasks). ML models suffer of a generalization problem, meaning that once they were trained for a dataset, the weights θ they find are not useful or transferable to new data. ML also underperformed in the tasks of recognizing speech and recognizing objects.

Data and features for ML have been traditionally quite small, and feature engineering was also needed to select features that are more useful than others. As soon as the dimensions grew, so did the *curse of dimensionality*, where the ML methods either took too long, didn't converge or behaved in strange ways.

To be able to generalize ML/DL needs to be guided by prior beliefs and the more data is show to it, the better. So with the increased number of observations that lead to Big Data came DL, with an exponentially growing number of parameters θ and its own set of rules and architectures.

4.4.2 Decision trees

Formalized in 2016, "Extreme gradient boosted trees" (XGBoost) [76] became one of the most commonly used regressors and classifiers. XGBoost is robust to sparse data and uses approximations for the tree learning which contributes

to scalability. For some given dataset with n data points x and m features, a tree ensemble that uses K trees f can be expressed as

$$\hat{y}_i = \sum_{k=1} K f_k(x_i) \tag{4.5}$$

Each tree f has its own structure and number T of leaves and weights θ. This is common to all decision tree algorithms, but XGBoost uses a regularized cost function which can be extended to any differentiable function l allowing it to perform different regression and classification tasks and even do multilabel classification. Let $\hat{y}_i^{(t)}$ be the prediction of the i-th data point at iteration time t, the cost function becomes

$$J^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_k) \tag{4.6}$$

where

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \|\theta\|_2 \tag{4.7}$$

.

Equation 4.6 shows two innovations from XGBoost: Regularization that penalizes the complexity of the trees and helps smooth the final learned weights and avoid overfitting, and the *greediness* of adding the best f_t to the cost is the gradient boosting. Using the approximation of the derivatives of equation 4.6 is another improvement towards the scalability of XGBoost.

Additionally XGBoost introduces shrinkage and column subsampling. Shrinkage scales newly added weights by a factor η after each step of tree boosting which is similar to a learning rate.

XGBoost makes a hyper efficient use of algorithmic optimization tools such as cache access patterns, data compression and sharing to build their tree boosting system which scales beyond billions of data points using few resources. XGBoosts are the most commonly used kind of decision tree algorithms in the main ML competitions in the world. One reason they are still used today even though more advances methods exist, is their interpretability; it is possible to assess which features are important for a decision and why. We used XGBoost in paper IV.

4.4.3 Deep learning

Deep neural networks, also known as neural networks, multilayer perceptrons, or even feed-forward networks are the paradigm of DL. Sometimes called universal function approximators, a DL network's goal is to approximate a function f, for example a classifier $y = f(x)$ that maps a feature vector x to a class y. The quintessential neural network is basically a function $y = f(x, \theta)$, where we want to learn the θ that approximates our

function best. The main difference between neural networks and the linear models we've been talking about is that their non-linearity makes most cost functions non-convex, so any guarantees of finding global optima are lost and instead, gradient descent is used in smaller (and smart) steps.

4.4.4 Neural networks

Neural networks are composed of several interconnected nodes (or neurons). These connections form an acyclic graph that weighs the connections with θ plus a bias and then uses what is called an activation function to transform the input into a decision that will go to what are called the hidden layers of a network. After going trough several hidden layers of weighted connections they arrive to the output layer where they are aggregated into the same space as y to give a solution. If the solution is not good enough according to a loss (or cost) function J then the error "goes back" in the form of a θ update using the gradients of all the activation functions. Some of the first neural networks created are called fully connected neural networks (FNN), where every node of each consecutive layer is connected. An example can be seen in Figure 4.1. FNNs were used in paper IV.

Fully connected networks work on disconnected data points. But in the fields of image analysis and computer vision the goal is to work with images and data that has a known grid organization. The first of such networks was called "convolutional neural network (CNN)". To work with sequences (of text or images) the networks are called " Recurrent Neural neworks (RNN)". Later on for the analysis of data in different topologies, like graphs, the "graph neural networks (GNN)" were designed. Also called "geometric deep learning"[77], these models attempt to translate ML/DL models into non-Euclidean domains such as graphs, meshes and manifolds. A short summary of GNNs is presented below since it is used in paper IV.

4.4.5 GNNs and node representations

While deep learning efficiently captures hidden patterns of Euclidean data, in data analysis there are a myriad of applications where data is represented in graph form. For example, in commerce data graphs represent the interactions between users and products. In a citation network, articles are linked to each other by citations. In social networks, users are connected as *friends*. Also sensor networks and computer graphics meshes are represented as graphs. In particular, biological networks connecting molecules, gene expression patterns and cellular neighborhoods in tissue are of interest in this thesis. The irregular topologies of graphs present significant challenges on the existing ML/DL algorithms, as graphs have a variable size of un-ordered nodes with

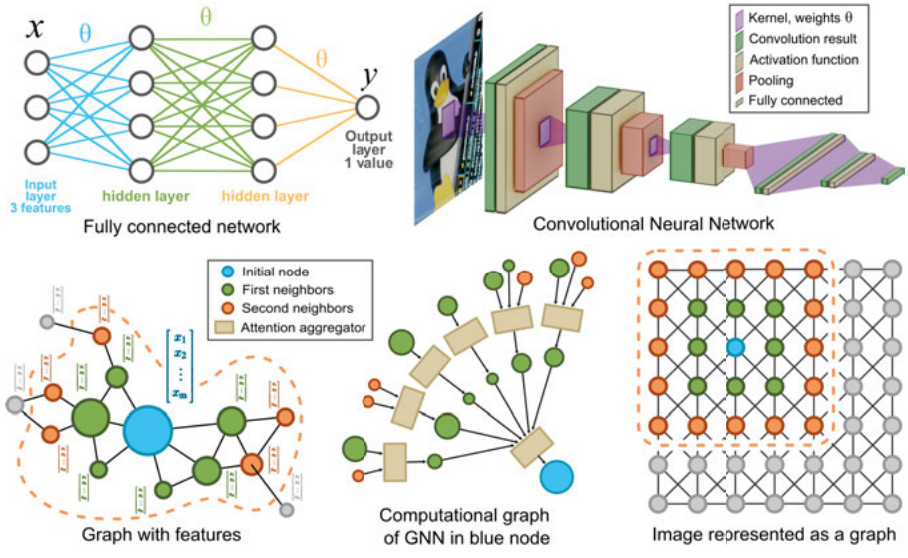

Figure 4.1. Different neural network architectures, FNN, CNN and GNN. FNNs connect all nodes between each layer. CNNs learn kernels that capture the main features to describe an image. A GNN is illustrated with an example graph and the computations necessary to create a representation for one node based on its neighborhood. The graph nodes are color coded to show the first neighbors and the second neighbors and the size reflects the degree of the node. In contrast an image can be represented as a graph where every pixel is a node and it is connect with its eight neighboring pixels. CNNs work better in images while GNNs are able to work on graphs with nodes that have different degrees.

different numbers of neighbors. As a result, operations like convolutions have to be redefined.

However goals remain similar. While GNNs can be used for many purposes, one of the most interesting for data mining is graph embedding or network embedding. The goal of network embedding is to represent the nodes from the graph as low-dimensional vector representations, preserving graph structure and information associated with the nodes. With such a low-dimensional vector representation, any subsequent downstream analysis task, such as classification, clustering, and recommendation, can be performed with easier methods. Not all GNNs are used for node embedding and not all node embedding methods are GNNs (or deep). But in the intersection of both worlds came Gilmer's neural message passing [78], Hamilton's GraphSAGE [79] and, and Veličković's graph attention networks (GAT)[80] and defined a model for learning a representation of nodes that captures its relationship with neighbors and also its features. A general model

for GNNs for node embedding:

$$h_u^{(0)} = x_u, \tag{4.8}$$

$$h_u^{(k+1)} = \text{UPDATE}^{(k)}\left(h_u^{(k)}, \text{AGGREGATE}^{(k)}(\{h_v^{(k)}, \forall v \in \mathcal{N}(u)\})\right) \tag{4.9}$$

where $h_u^{(k)}$ is the k$^{\text{th}}$ update of the *hidden embedding* of u, and the aggregation operation takes the information of nodes only in the neighborhood $\mathcal{N}$ of u. The aggregation function can be almost anything as long as it has the following properties: i) Permutation invariance (the order of nodes should not affect the result). ii) Insensitivity to node degree. For example, an aggregation like a sum is unstable if there are nodes with very high degrees (many neighbors). iii) Locality, consider only neighbors. iv) Weight sharing, so that the number of nodes does not affect the model and also to be inductive and generalize to unseen nodes.

The aggregation used in paper IV is that of attention, which is also very popular in RNNs. In [80] GATs use attention aggregation which is a weighted sum of neighbors:

$$\mathbf{m}_{\mathcal{N}(u)} = \sum_{v \in \mathcal{N}(u)} \alpha_{u,v} \mathbf{h}_v, \tag{4.10}$$

in which $\alpha_{u,v}$ is the attention weight on neighbor $v \in \mathcal{N}(u)$ only during the aggregation of features of node u. The original α implemented in the library used in paper IV is:

$$\alpha_{u,v} = \frac{exp\left(\mathbf{a}^\top [\mathbf{W}\mathbf{h}_u \bigoplus \mathbf{W}\mathbf{h}_v]\right)}{\sum_{v' \in \mathcal{N}(u)} exp(\mathbf{a}^\top [\mathbf{W}\mathbf{h}_u \bigoplus \mathbf{W}\mathbf{h}_{v'}])} \tag{4.11}$$

where $\mathbf{a}$ is a trainable attention vector, $\mathbf{W}$ is a trainable matrix and $\bigoplus$ denotes concatenation

Figure 4.1 shows different kinds of neural networks. In particular it focuses on GNNs which have complex architectures. An example graph is provided along with the computational graph for one node. The computation and aggregation of features of its neighbors, gives the representation to the central node which will be able to give it an identity summarizing its neighborhood.

5. Data visualization

So far we have discussed datasets consisting of millions of points and enormous images but how do we avoid drowning in information and quench our thirst for knowledge? Data visualization.

Already in 1986, the book "Visual display of quantitative information' [3] offers a guideline to create visualizations of data:

- Show the data.
- Avoid distorting what the data has to say.
- Make large datasets coherent.
- Encourage the eye to compare different pieces of data.
- Reveal the data at several levels of detail, from a broad overview to the fine structure.

To which I would add:

- Open and look at your images. You don't want to run a lengthy and costly processing on saturated data or images without content!
- Do an initial exploratory analysis of your data, how many dimensions? What ranges? How is it distributed? Does it have outliers? What does each value represent?
- Talk to your collaborators and plan and ask relevant questions, get a *sense* of the data.

All of the above will prevent you from making decisions on spurious correlations [81] .

In 1986 large datasets meant 21000 entries as opposed to various sets of millions of points and the *gigapixel* images of today. However the rules remain as valid as they were back then.

Since then, computers have kept evolving and fields such as computer graphics and design have evolved with them and enable us to create hyperrealistic visualizations and simulations. More importantly, it allows us to interact with the data so that we are able to support or discard hypothesis and create new ones based on new discoveries and *insight*.

The book "Data visualization" by Telea [82] uses the word insight to describe two types of information that can be obtained from a visualization application: answers and facts.

If we knew the questions we were asking, a good visualization should show answers to those questions. Additionally if we were unaware of a fact, a good visualization can attract our attention towards interesting locations.

5.1 Maps of high resolution data

When we talk about spatial information we mean that any data we have, i.e sizes, colors, shapes, we have also a coordinate of such a point within a frame of reference. In images and the data we associate to it, we have a typical square Cartesian space, with coordinates X, Y and Z (sometimes Z is not spatial). We have an origin, usually denoted with a zero vector $(0,0,0)$ and it is usually the corner of an image. Each pixel then has a coordinate and so do the objects and information we find in them. While this can sound obvious to anyone who has seen a map, it is not always the case for microscopy data and images. In microscopy, occasionally, the preparation of a sample requires its disassembly, making it impossible to study tissue heterogeneity and histological information in the form of tissue architecture and organization is lost in the process [83]. As we have seen throughout this thesis, such information is very important to understand locations, identity and relationships of cells, which work together and not in isolation. In this way, we can know where in an image a cell is located and we can have information about its characteristics and also the ones of its neighbors and relate them this way. We can know if a gene was activated and in which cell inside a tissue, all of this within millions of data points.

In images, high resolution means there is a lot of information per unit area of space. The higher the resolution, the more pixels per inch, or per centimeter, millimeter or micrometer (although more pixels does not does not necessarily mean higher resolution). This also means that we can *resolve* (observe or identify) tiny details like cracks in a painting that we can only see when we are very close.

To look at large maps, we have to decide what level of information to show given the limitations of our screens. In 2007, a software called Seadragon was developed at Microsoft. Later it was open-sourced and renamed OpenSeadragon[25] and maintained by one of its original developers. Its creator once said: "The only thing that ought to limit a system like this one is the number of pixels on your screen". What that means is that we only care about what is happening on my 1980 times 1200 screen pixels and not whatever is happening outside so it doesn't really matter (for visualization) if the image has 10^6 pixels or 10^9 (gigapixels). What matters is the area that the screen covers and the zoom level. Figure 5.1 shows an illustration of the view in a screen and the data that is loaded for a specific view. In the back ground, all the information for the map exists but we don't bother loading if we are not looking at it.

In order to disregard the are outside the screen, large images of any kind are divided into tiles and in various magnification levels, as explained in chapter 2.2. The tiles and the magnification (or pyramid level) that is loaded is only that which is needed to cover the current view. This way the search for a

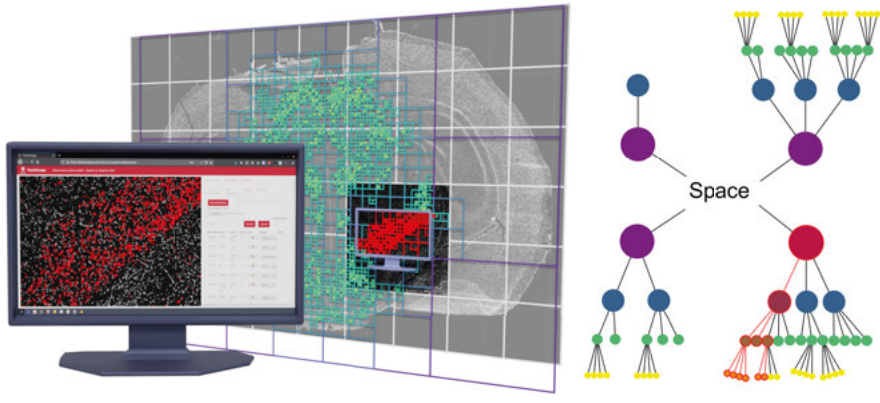

Figure 5.1. Illustration of efficient tissue map visualization. The figure shows one level of a tiled image from which only the needed tiles for the current view are loaded. A quad-tree for a set of points is depicted. It shows that dense areas require more resolution levels while empty areas are not divided further. An illustration of the tree is presented where the main space is divided in four quadrants but the tree does not necessarily have data or children in each node. Requesting the data from a view in the user interface sends the four corners of the view in image space and this is used to query the tree only in the relevant branches, thus only following a path on the tree instead of a full array of points. In TissUUmaps, each gene has it's own tree and only the currently displayed genes' trees will be visited, further increasing efficiency.

particular set of tiles can be parameterized by the current view's corners and the zoom level.

The same way that images can be represented in a pyramidal structure, point data can also be stored in efficient data types for fast querying. Suppose we want to find the location and identity of points inside a particular region or view, the naïve way to do it is with a linear search. We could go through each of the millions of points and ask if they are located within the region of interest. This is extremely inefficient and would make it prohibitively expensive to visualize anything. To do searches of this kind, tree data structures are useful. To quickly find neighbors, kd-trees, where kd refers to k dimensions, are the solution. To find points within regions, quad-trees are used in 2D and octrees are used in 3D. Quad-trees are quick to create and only have to be created once, as opposed to a linear search where every point would be visited in every query. Quad-trees separate the total space spanning the data into four quadrants or equally sized areas. If any data point lies within each of the newly created quadrants, such quadrant will be divided again. This process is repeated until there is a single point in a quadrant (or very few of them). When there is a need to find points within a squared region (such as a view in the screen) we only need to know where the four corners of the view are located within the quad-tree and that automatically provides a list containing only the points in the desired region. In paper I, I

used quad-trees to store the points and query them, both for visualization and for quantification within regions that the user can input (manually or by code). Unlike an image pyramid, quad-trees only store pointers to the next level if the next level has a higher resolution. This way, areas without data points don't have more levels while dense areas with many data points have more levels.

When data consists of millions of points it is good to find optimizations to find which points lie within non-convex regions, such as polygons marking regions of interest by a user. In order to determine the points inside an irregular polygon, borrowing techniques from game development and computational geometry and using the quad-trees, I first found the points inside a rectangular region that contains the user's polygon and then used the Lagidse [84] method to quickly find points inside the polygons.

With these techniques one can produce a map of high resolution data that displays the data. It avoids distorting what the data has to say, makes a large datset coherent, encourages the eye to compare different pieces of data, and reveals the data at several levels of detail.

5.2 Digital color

As mentioned briefly in section 3.4 there are multiple color representations. The most common is RGB due to its use in images from cameras and the internet. It is also inspired by the cone cells present in the human eyes that enable us to observe different wavelengths in the visible spectrum. However the composition of any given color into its R, G and B components is not intuitive. Red, green and blue do not mix digitally the same way they do in the physical world. Because of this, specialists in design, cognitive scientists, physicists, and computer scientists have developed other color representations, such as HSV, HSL, YCbCr and several more. While RGB is a cubic square space, HSL and HSV use cylindrical coordinates. In both, H stands for Hue and S stands for saturation. Hue is the color, anyone from a continuum in the range of the visible spectrum such as the ones we see in the rainbow. The saturation is the strength of this color, or a measure of the amount of color. A desaturated image would look to us as a black and white picture. V and L, value and luminance, correspond to the whiteness of the color. A value of one means the color is white, and a value of 0 means the color is black. YCbCr uses two axes to represent blue and red in the four quadrants of a Cartesian space. Positive values along the axis represent blue and red while negative values along the axis represent green and orange. The y value represents the luminosity. This color space is used in digital video and photography. In Figure 5.2 RGB, YCbCr and HSV are illustrated[1].

[1] printed versions of this document might not fully reproduce the colors.

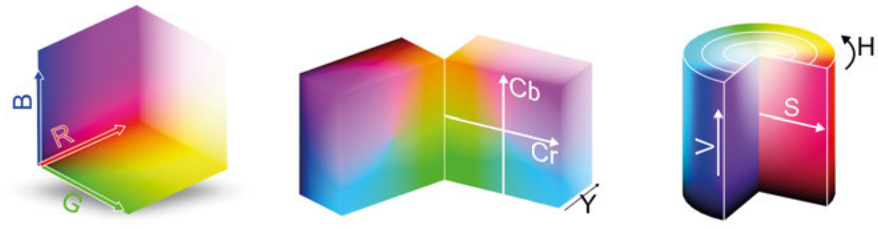

Figure 5.2. Representation of color spaces RB, YCbCr and HSV. The YCbCr block is divided into 2 to show the two halves of the cube.

Here I would like to throw in a word of caution, and it is a mistake I have seen constantly be made by people without experience in design, which is the assumption that the number of dimensions in their color space is the limits of categories that can be represented. For example assuming that there is a limit of three categories when using red, green and blue. But the worst of all is the one that assumes that CMYK is a good color space for representing four categories. CMYK (cyan, magenta, yellow and black) is not a digital space per se. It is not a palette used in design. It is a color space that was designed to convert any color into a value that could be translated into the ink toners of a printer. This color space was not intended to be used directly on the design but only as a means to print on paper. Cyan and yellow are two colors with high values (luminosity) associated. In the screen, these colors can be almost indistinguishable. Magenta is more distinguishable but still has high luminosity. In stark contrast, the black color K is the absolute 0 of luminosity, and both in computers and in physical life, black consumes all the colors. Any combination of black with another color will still be very dark. By using black as a "color" and cyan/yellow in a figure, the cyan/yellow objects will *disappear* and the black will attract all the attention. The result would be a figure that looks like a blot of ink and half of the information cannot be observed, breaking the two first items in the Tufte guidelines, the most obvious one, "show your data" and "avoid distorting what the data has to say".

Using HSL or HSV one can represent an infinite amount of categories. Since H is an angle in the unit circle, as seen in figure 5.2, it suffices to divide the circle into the number of categories to get n different colors. This can be specially convenient in clustering visualization where there is a large amount of clusters.

One interesting use of these color spaces and techniques is the use of dimensionality reduction to reduce from a high-dimensional space to two or three dimensions and to use these color spaces to immediately separate clusters via the different colors that appear. An example of this can be seen in paper IV which I will explain in the next section, and paper R1.

5.3 From data to visualization

One of the things you need to know about your data is whether it is numeric, categorical, or both. Or perhaps if it is a graph with nodes and edges, or if it is a map with spatial information or if it is a time series. Numeric data means that each point is one in a continuum (though it can be discrete), this means that points have an inherent scale and ranking and it is possible to know if a point comes before or after another. For example, when quantifying the amount of cells in a sample, 100 cells are less than 1000. In categorical data the available classes do not have an inherent order or rank. For example, shape and color is categorical data; there is no order in it being blue or pink or round or stretched. Cells and their neighbors can be represented by networks and graphs connected by edges, as seen in chapter 4. Nodes may or may not be associated with a coordinate, and may or may not have numerical and categorical data associated to them. In time series, points are organized by time and they may or may not have numerical categorical data associated.

When visualizing, one has to answer the question: What do we need to show? The D3.js graphical Library [85] has a good division of the possible graphs that are used in data visualization depending on the goal:

- *Distribution*: Violin, density, histogram, boxplot, and ridgeline.
- *Correlation*: Scatter, heatmap, correlogram, bubble, connected scatter, density.
- *Ranking*: Barplot, spider/radar, wordcloud, parallel, lollipop, circular barplot.
- *Part of a whole*: Treemap, donut, pie chart, dendrogram, circular packing.
- *Evolution*: Line plot, area, stacked area, stream chart.
- *Maps*: Map, chloropleth, hexbin map, cartogram, connection, bubble map.
- *Flows*: Chord diagram, network, sankey, arc diagram, edge bundling.

To exemplify the use of some of the graph, let us break down figure S3 in paper IV which is referenced here as Figure **??**. Let us divid it into three main areas, two on the top and one in the bottom. The first part in the top contains a histogram that shows the amount of cells that have a specific neighborhood representation. This neighborhood representation is a 2D point that lies in a circle due to contrastive learning and normalization. The location along this circle is an indication of the kind of neighborhood a cell resides in. The cosine distance between two points can tell how similar the neighborhoods of two cells are. Since the points lie in a circle, I mapped the angle of each point to the H from HSV to get an instinctive separation of different neighborhoods and a similar color between similar neighborhoods. Below the histogram, is a heatmap where the colors in the rows represent the different classes of cells in the dataset. The saturation of the color indicates the percentage of cells of a particular class that are found in such kind of neighborhood with respect

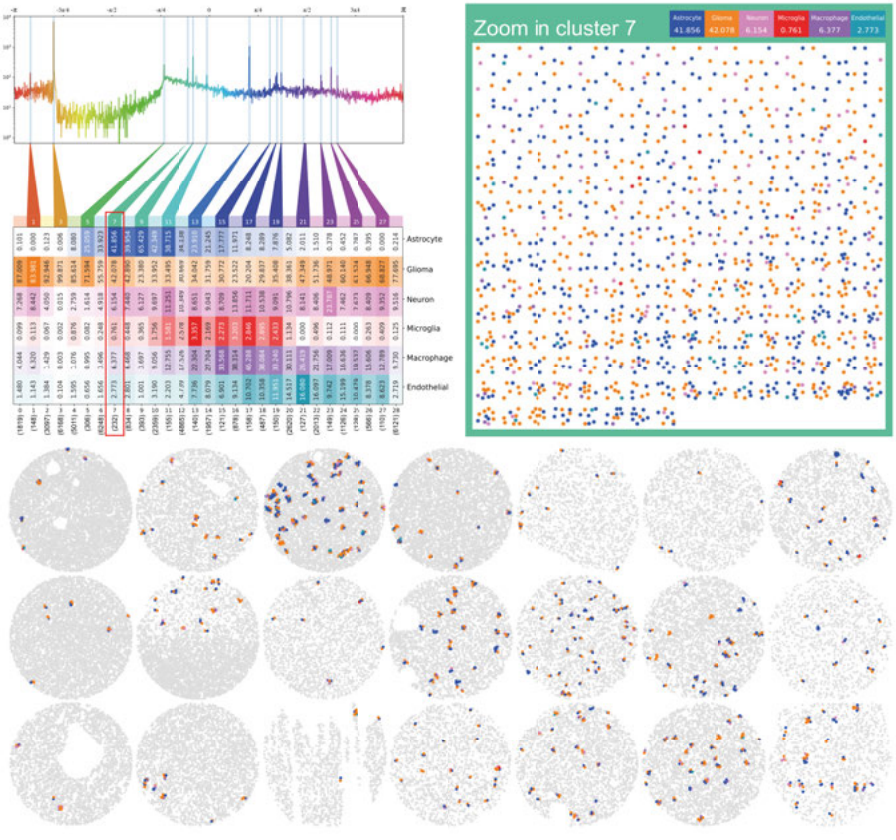

Figure 5.3. Figure S3 of paper IV: "Machine learning for cell classification and neighborhood analysis in glioma tissue", as an example of visualization. Bear in mind it was designed for a longer page width and possibly to be viewed more online than printed in paper. As a consequence, some labels might be too small. The histogram's horizontal axis is the angle in radians and the vertical axis is the cell count.

to the whole dataset. The numbers in the heatmap represent the percentage of cells of a particular class within that specific kind of neighborhood. The labels identifying the clusters are also colored in the top of the heatmap, such that those with low saturation are very heterogeneous therefore information in the column in the heatmap is not as representative of the neighborhood as those whose labels are more saturated. The amount of cells with a kind of neighborhood are displayed in the bottom of the heatmap and the names of the classes are shown at the right.

The second part of the top row of the figure shows each individual neighborhood of a cell in one specific kind of neighborhood which is marked by its color and by its number. The cells are color coded by their class and the corresponding column is shown again. This neighborhood is represented

mainly by the presence of two classes of cells (glioma/orange, astrocyte/blue) and this becomes apparent in this figure.

The bottom part of the figure shows the neighborhoods in their original positions in the tissue cores, allowing to visibly detect which cores contain the neighborhoods and which don't, in what amounts and with which compositions.

In figure 4, the cells are color coded with the H as explained previously and is an example of using differences in color to map differences in embedding or vector representation.

5.4 Open source software libraries

Several libraries for visualization exist in python. Matplotlib [86], Seaborn [87], Bokeh [88] and various others. These libraries offer a wide range of functions to do all the kinds of plots discussed in the previous section. Figure S3 of paper IV was done mostly using Seaborn. Matplotlib is perhaps the most common one as it was one of the first to exist and is inspired in Matlab's visualization library, but can have limitations in handling large datasets and in interaction. Seaborn built on top of matplotlib and has very handy functions to make plots directly from Pandas[89] dataframes. Bokeh is a very complete library that shines in the interaction field. It has two major components; a python backend and a TypeScript (JavaScript) client-side library. Bokeh combines python's power for data analysis and the power of JavaScript for web based interactive experiences.

D3 (Data Driven Documents)[90] is a very powerful JavaScript library. While it has no backend, it is powerful enough to perform data analysis in the browser. I used D3 in paper I to draw all the svg points and even used their quad-tree implementation. D3 is almost ubiquitous in the web for creating interactive data analysis and graphs in 2D.

For 3D visualization, there is arguably no better software than Blender[91]. For the last 20 years it has been used by millions of artists, scientists and specialists in computer graphics to create simulations of millions of particles, realistic renders, shader programming, video editing and many more. It uses C and C++ for heavy computations and Python for the user interface and API. Its large base of users have created additional features such as point cloud visualization which I used during the course of this thesis to explore 3D UMAP representations of thousands of cells. I also used it to design the cover.

However, in terms of creating interactive experiences on the web in 3D the main library is WebGL [92] and libraries made on top of it such as Three.js.

6. Contributions / Methods and applications / per paper

The scope of this thesis spans through a variety of projects and is always in the bridge gaping the knowledge between computer science and biology. The main contributions of this thesis is the creation of necessary tools and methods for the analysis of tissue and the development of visualization software and techniques.

When we want to visit a new city, we use maps. We determine important locations and the fastest way to travel between them and we give priorities to places depending how far they are from the main location of our visit. In today's world, arguably the best tool for such use of maps is GoogleMaps, where people can find things at a glance or draw conclusions after interacting with the map. Amongst the goals of the work in this thesis was not only to develop methods and algorithms but also the software tools to efficiently study tissue as a map, thus we created TissUUmaps, combining Tissue, Uppsala University and maps.

When a question arises and samples and images are acquired, the first natural step is to observe the data. Scientists need to be able to open and observe digital images in an easy and convenient ways, we need to be able to demarcate regions of interest and be able to store them. We also need to be able to observe different images together to come up with new hypotheses based on observations of data. Data does not come only in the form of images, but also as data points originating e.g. from earlier image analysis steps, such as decoding of gene expression, or extraction of cell features. To be able to do all this, in Paper I we presented the design and implementation of TissUUmaps: a visualization software tool that works easily with or without a server, locally and remotely via a web interface to enable the observation of one or several whole slide images (WSIs) or tissue micro arrays (TMAs). More information can be found at tissuumaps.research.it.uu.se/howto.html where several different projects are on display.

Biological processes are regulated by a multitude of factors, therefore multiple points of view help to understand an event and characterize a disease. All such factors are visible only under certain conditions and can only be evidenced and captured in images using different techniques which result in separate pieces of knowledge. To bring this information together in terms of image analysis, images coming from different samples or from

physically different pieces of tissue need to be spatially aligned by means of registration. In papers II and III registration is used to find the colocalization of two proteins coming from consecutive slices of tissue.

Tissue is composed of cells, each cell has a function so that the tissue can function. The identity of cells is one necessary component to understand the mechanisms of a disease. In paper IV we used machine learning to both determine the identity of the cells and to describe cell interactions by their neighborhoods using graph neural networks.

6.1 Paper I - TissUUmaps: Interactive visualization of large-scale spatial gene expression and tissue morphology data

Working with large amounts of various kinds of data and putting it together to draw a faithful picture of a tissue sample comes with numerous challenges as explained in section 2.2. Software for this purpose requires an architecture and a design based on constraints and requirements.

In terms of software development, for TissUUmaps an original non-functional requirement was to work online or offline, to not require a server but have the ability of using one if necessary. It should also be easy to install and scale.

Functional requirements included the ability to query and load tiles to display as a full image in a viewer and be able to overlay data on it that would zoom and pan along with the image. It should allow for the choice of data to be displayed and such data can have properties such as color and size. The viewer must support the visualization of millions of data points extracted from images, and have the ability to quantify points within regions given as input by the user.

All big software projects come with difficulties and this was no exception. Having no access to a server allows software to run locally or via the web at the cost of having the image tiles and data computed outside of TissUUmaps. Thus becoming more of a front-end software which still needed to perform requests for data. For this purpose any kind of web server can still be installed locally, either industrial scale like Apache or small lightweight like Flask, or temporal like Chrome web server add-on, but it has to be able to accept requests for images and data.

TissUUmaps was built as a set of modules, each in charge of different aspects, like creating and interacting with the user interface, or loading data, or computing geometrical data structures to later query data more efficiently. All of this was done in vanilla JavaScript with the help of D3 for visualization and OpenSeadragon for querying the correct image tiles for the view.

Although the concept of a viewer for WSIs and microscopy data is not new, our software is the first to offer a convenient way for visualising millions of data points, along with the microscopy image data via a web interface. Page limitations prevented us from explaining software development techniques and data management in the paper, however, TissUUmaps is open source and documented and can be found in the repository of the Wählby-lab research group in github. It is currently used to display several projects and has an interesting future.

6.2 Paper II - Towards automatic protein co-expression quantification in immunohistochemical TMA slides

To study a disease such as gastric cancer it is necessary to locate and relate proteins that play roles in tumour progression. As explained in section 2.1.2 IHC allows for the visualization of proteins within their context in the tissue.

In this paper we wanted to study two proteins in samples arranged in TMA slides. We presented a methodology for revealing the protein information in the image, represented by the DAB stain, and the general scaffold of the tissue, represented by the H stain. Consecutive tissue slices are expected to have a similar scaffold, which we used to guide the image registration which finds a transformation that we used to align the two TMA slides. Once aligned, the two proteins can be observed together in one image as opposed to a visual assessment using two separate images.

In parallel, using RF and image features we segmented tumour regions to mark the locations in which protein co-expression needed to be quantified. Combining the protein co-expression image, and the tumour segmentation, we were able to automatically quantify protein co-expression in tumour regions.

The contribution of this paper is to work with large images and bring together all the steps necessary for protein co-expression quantification, such as visualization, annotation, color unmixing and registration and exploring different methods for them and their parameters. We put them together into a pipeline which allowed the co-expression quantification of E-cadherin and CD44v6 in several TMA slides. The software is available at github.com/wahlby-lab/TMA-studies

6.3 Paper III - Whole slide image registration for the study of tumor heterogeneity

This paper is closely related to paper II. The main contributions were to develop a pipeline for piece-wise alignment of WSI using manual annotations. We developed an interface for synchronized annotation where

corresponding points were selected so that an affine transformation could be found between three WSI of consecutive tissue slices. This allowed for the exploration of three proteins and their combinations within tumour regions. We also showed a comparison with alignment based on SIFT features found on the corresponding H stains

6.4 Paper IV - Machine learning for cell classification and neighborhood analysis in glioma tissue

Tumor cells have intricate mechanisms and relationships with the environment that surrounds them, namely the Tumor Microenvironment (TME). Cells communicate with neighboring cells of different types to accomplish the goal of sustaining or fighting the tumor. To study the TME in glioma, in this paper we identified and classified glioma cells based on their expression profiles quantified in TMAs acquired using multiplexed immunofluorescence. Once cells were identified we did an exploratory analysis of the kinds of neighborhoods that appear, their quantity and location.

Classification and neighborhood analysis was done using ML models. For classification we used an ensemble of FNN and for neighborhood analysis we used GNNs.

There are multiple contributions of this paper. We introduced a marker intensity feature which is insensitive to variability amongst the cores in the TMA. We called it D90 since it is the difference between 90th percentile of the marker intensities quantified within a cell. We compared the accuracy of ensembles of two popular ML/DL architectures, RF and FNN. We showed how both architectures work and how they can both have advantageous features. We compared results with and without the use of D90s. We observed that the ensemble of FCNN performed better.

Followed by the classification done with the FNN ensemble we created a graph per TMA core where each cell corresponds to a node. This graph was used to train a GNN to find node embeddings that represent their neighbors. To do this in an unsupervised fashion we used contrastive learning, where one assumes that nearby cells have more similar neighbors than a random cell far away. We show the network a positive and a negative example and this allows the GNN to learn without manual annotation. We developed a technique for visualizing these neighborhoods and organizing them using a single value that identifies them.

7. Conclusions and future work

This thesis comprises a broad range of topics that are all part of big projects that are intended to answer biological questions. The insight brought by these answers will allow the understanding of mechanisms of disease and the improvement of treatment of patients and quality of life.

It is my pleasure to say that the development of all this work is ongoing and it will keep on being used.

The work in paper IV is now the base of what will become a large scale study and the methods will be applied to hundreds of TMA cores. Cells will be classified and their neighborhoods will be explored and analyzed with respect to patient data. The GNN methods will be further developed to understand the neighborhoods as they were originally developed for gene expression analysis [66] and not for cell neighborhoods. However they already show promising results. Additional work has to be done in the visualization and interpretation of the neighborhoods. In this thesis a new way of visualizing the cell neighborhood as a continuum is presented but it is still noisy in terms of the normalized amount of neighbours per class. Having a measure of trust on the cell classes and a clearer view of the neighborhood representation will allow us to not only relate pathologist knowledge with clinical data associated to patients but also to discover possibly unknown differences between Astrocytoma and Oligodendroglioma.

The conscientious architecture and documentation of TissUUmaps has already enabled its extension and the creation of spin-off tools. A "FlaskTissUUmaps" version is now available which removed the need for the pre generation of tiles of images and uses python's Flask[93] and OpenSlide [20] to do so. This at the cost is having to install python and an environment specialized for the software which was contrary to one of the initial requirements of TissUUmaps. However, due to the increased interest and wishes from the users, the FlaskTissUUmaps version improves greatly in usability and new features.

My personal favorite extension of TissUUmaps is the newly added WebGL[92] renderer that uses the HTML canvas to quickly render millions of points. It improves over my initial design of adding markers as SVG elements using an efficient search depending on the user's view. WebGL takes advantage natively of the GPU or integrated graphics to accelerate the rendering of the markers. TissUUmaps allows the use of the browser console as a data analysis tool by making all the libraries and modules available directly in the command line, this includes all the data that is loaded and that

is displayed. In the interface the user selects the columns that are to be displayed, namely spatial coordinates, gene expression information and morphological information (data need not be of gene expression but it's the main use case). While not readily accessible via the interface, all the additional data such as quality of decoding, embedding coordinates, colors, etc, can be accessed via the browser console. However the less tech-savvy user will benefit from the additional interaction with the measure of quality of the decoding directly in the interface.

CytoBrowser [94] forked[1] TissUUmaps and converted it to a Node.js application for the study of cytological data such as that resulting from smears and swabs. The images used in cytobrowser are a stack of RGB images of different focus levels, such images are challeging to load interactively and the authors opted for using the original TissUUmaps mechanism of pre generating the pyramidal images for all focus levels using VIPS[23].

[1] Term introduced by Github to mean a full copy of a software repository that is developed in its own independent project.

Summary in Swedish

Att se på biologisk vävnad genom ett mikroskop är som att se på en karta över ett land. Det finns städer, vägar som förbinder dem, och det finns speciella byggnader med specifika funktioner såsom sjukhus, en polisstation, en brandstation, ett snabbköp och ett riksdagshus. Människor rör sig mellan platser och utför funktioner som får samhället att fungera. I en vävnadskarta är människorna celler.

Alla levande varelser är byggda av celler. Små, invecklade maskiner vars uppgift är att hålla oss vid liv. Varje cell har sin egen identitet, särdrag och funktioner. Vissa celler kan ge struktur och stöd åt vävnadsarkitekturen, vissa transporterar näring dit den behövs medan andra tar hand om mikroskopiska inkräktare och skyddar oss. När dessa celler slutar att utföra de funktioner de är byggda för blir vi sjuka; till exempel bryts cellstrukturer sönder och resurser avleds till att utfodra och syresätta tumörer.

Till skillnad från stora organismer som människor, har celler inte ögon eller munnar för att kommunicera. De kommunicerar genom kemiska signaler och speciella instruktioner i den genetiska koden. Under normalt beteende, tar celler små molekyler och arrangerar dem i större, komplexa strukturer enligt instruktionerna i sitt program. Dessa större strukturer kallas proteiner och de kan hjälpa till att bygga strukturer, till exempel en barriär i våra magar som åtskiljer näring från skräp. De kan även skicka och ta emot meddelanden, som en instruktion att flytta sig till en viss plats, leverera ett specifikt näringsämne, eller att identifiera och attackera en inkräktare.

I själva verket är det få delar av den genetiska koden som består av instruktioner till proteinerna. För att veta vem de är och vad de ska göra med sin tid, tittar cellerna på sin genetiska kod. Koden är densamma för alla celler i vår kropp, med skillnad i vilka instruktioner som är på- eller avslagna. Celler kommer endast att utföra de instruktioner som är påslagna. Dessa instruktioner kallas gener. De gener som en cell har påslagna, kan avslöja för oss dess identitet, funktion samt vilket organ den kom från.

För att kunna hjälpa sjuka människor måste vi först förstå vad cellerna gör. Likaså, för att förstå hur celler fungerar skapar vi profiler baserade på deras meddelanden, proteiner och gener. När celler är sjuka kan deras profiler ändras, vilket gör att vi kan särskilja dem från friska celler. Alla funktioner som främjar sjukdom kallas sjukdomsmekanismer, och för att studera dessa behöver vi mikroskop. Precis som med en digitalkamera kan vi ta bilder på celler. Då celler är så gott som genomskinliga måste vi färga dem. Vissa färgämnen syns då vi lyser på dem med blixten från en kamera. Andra är

flourescerande, vilket betyder att de reagerar på ljus av en viss våglängd och syns endast då de blir belysta med sådant ljus. Den första sortens färgämne kan granskas med ljusfältsmikroskopi och den andra med flourescensmikroskopi.

Att generera ny kunskap för att förstå sjukdomsmekanismer kräver ofta integration av flera olika källor av bildinformation från olika avbildningstekniker och olika förstoringar. Genom att använda rumslig information kan vi skapa kartor av celler och vävnader, och mäta t.ex. hur många celler, vilken typ av celler, och vilka gener som är aktiva och vilka proteiner som finns på olika ställen i vävnaden. Den här sortens kartor ger oss information om vävnaden, som t.ex. cellernas ursprung, densitet, organisation, identitet, aktivitet, och samspel med andra celler och grupper av celler i vävnaden. Den informationen kan i sin tur kopplas till sjukdomsprognos och överlevnad samt effektivitet av olika läkemedel. Den här tvärvetenskapliga avhandlingen beskriver en mängd metoder för bild- och dataanalys som har som mål att få ut denna typ av information från mikroskopidata, tillämpade på bilder från fluorescensmikroskop och ljusfältsmikroskop.

Genom att använda mikroskopibilder av biomarkörer tillsammans med avancerade maskininlärningsmetoder, kan vi identifiera celler och deras funktion. I ljusfältsmikroskopibilder identifierade vi olika proteiner i parallella vävnadssnitt och matchade snitten mot varandra för att mäta hur proteinerna samverkade i samband med magcancer. I en annan tillämpning med bilder från avancerad multi-kanalsflourescensmikroskop använde vi neurala nätverk för att bestämma cellidentiteter. Dessa identiteter använde vi sedan vidare för att analysera den lokala cellomgivningen i hjärntumörer för att försöka hitta cellinteraktioner och områden som kan kopplas till specifika sjukdomsmekanismer och prognos. Vi använde ensembler av maskininlärningsmodeller för klassificeringen av cellerna, både för att få bättre resultat men också för att få ett konfidensmått på klassificeringarna.

Avhandlingen inkluderar också skapandet av ett visualiseringsverktyg, TissUUmaps, som gör det möjligt att utforska vävnadsbilder från mikroskop med olika typer av information från bildanalys pålagda som lager eller markeringar. Bilderna är enorma vilket medför utmaningar som kräver avancerade datahanteringsmetoder för bearbetning och visualisering. Genom att utveckla TissUUmaps har vi gjort det möjligt att t.ex. se miljoner av genuttryck överlagrade med originalbilden. Detta tillåter kontextuella studier av vävnad i cellulär och subcellulär upplösning. Utöver de tidigare nämnda tillämpningarna användes verktyget också för visualisering av vävnad och motsvarande cancergraderingar beräknade med djupinlärningsmetoder, samt för att annotera par av konsekutiva vävnadssnitt och hitta specifika områden av intresse i matchade bilder.

Metoderna och verktygen lägger tillsammans grunden för att analysera och visualisera stora och komplexa rumsliga vävnadsstrukturer. Att förstå rumslig information om vävnad i olika sjukdomar banar vägen för nya upptäckter och förbättrad behandling för patienterna.

Summary in Spanish

Ver tejido biológico bajo el microscopio es como ver un conjunto de mapas, de diversa índole y entretejidos de múltiples formas. Hay ciudades, vías que las conectan, fronteras, tipos de suelo, recursos, bienes inmuebles con objetivos específicos, entre otros.. En ellos, siguiendo el símil, se encuentran las personas, que se mueven de un lugar para otro y se articulan con otras, realizando actividades que permiten observar el funcionamiento de una sociedad. En un mapa de tejido biológico, estos individuos son representados por las células.

Nosotros los seres humanos estamos compuestos por células. Pequeñas y complejas máquinas que trabajan armónicamente con el principal objetivo es mantenernos con vida. Cada célula tiene su propia identidad, características, objetivos y funciones. Algunas células pueden dar estructura y soporte al tejido mientras que otras pueden transportar nutrientes a donde sea necesario. Así mismo, existen otras células que se encargan de identificar intrusos y protegernos de ellos. Cuando las células dejan de cumplir sus funciones, el cuerpo del individuo empieza a presentar diversas anomalías, que pueden desembocar en una gran variedad enfermedades e incluso la muerte . Por ejemplo, los tejidos se deshacen al perder su soporte estructural o por ejemplo los recursos vitales que requieren las células para cumplir con sus funciones pueden empezar a ser malgastados en alimentar y oxigenar un posible tumor.

Pero a diferencia de nosotros los humanos que tenemos órganos sensoriales para comunicarnos, como los ojos, las manos y la boca, las células se comunican a través de señales químicas e instrucciones especiales en el código genético, permitiendo así el envío y recepción de mensajes, tales como una instrucción de moverse a un lugar específico, entregar un nutriente en algún destino, o identificar y atacar un intruso. En condiciones normales, las células toman pequeñas moléculas y las transforman en entidades más grandes y complejas llamadas proteínas. Las proteínas, entre otros, ayudan a construir estructuras como por ejemplo, la barrera intestinal, que permite dar acceso a diversos nutrientes evitando el paso de otros elementos que impidan el correcto funcionamiento del intestino.

Con el fin de conocer su propia identidad y sus funciones, las células encuentran en el código genético las instrucciones que la definen . Aunque este código es el mismo en todas las células de nuestro cuerpo, en cada célula se puede diferenciar qué instrucciones están encendidas y cuales están apagadas. Por ende, las células sólo realizan las instrucciones que están encendidas, a estas instrucciones se les conoce como genes. Los genes que

una célula están encendidos proveen información acerca de su identidad, su función e incluso el órgano del que provienen.

Para ayudar a quienes presentan algún tipo de enfermedad se necesita entender de manera íntegra a las células. Para ello creamos perfiles basados en sus mensajes, proteínas y genes, así, cuando los perfiles cambian, podemos diferenciar una célula enferma de una saludable.

Todas las funciones que favorecen a una enfermedad se les denomina mecanismos de enfermedad, y para estudiarlos se usan diversas herramientas, principalmente microscopios. De la misma forma que una cámara digital toma fotografías, con un microscopio podemos capturar imágenes de células y tejidos. Sin embargo, las células son transparentes, así que para observarlas en su ambiente natural necesitamos teñirlas. Algunas tinciones aparecen fácilmente cuando se les ilumina con una luz similar al flash de una cámara fotográfica. Otras tinciones son fluorescentes lo que significa que sólo pueden ser observadas si se iluminan con una longitud de onda específica. El primer tipo de microscopía se denomina microscopía de campo claro y la segunda se denomina microscopía de fluorescencia.

Para entender los mecanismos de las enfermedades, la extracción de datos requiere la integración de múltiples fuentes de información en varias magnificaciones y técnicas de adquisición y procesamiento de imágenes. Usando información espacial podemos construir mapas de tejido biológico y células en los que es posible cuantificar perfiles proteínicos y perfiles genéticos directamente sobre la imagen. Estos perfiles revelan información acerca de las células, tales como su origen, densidad, organización estructural, identidad, actividad, interacciones y comunidades. Esta información puede relacionarse con la supervivencia y la efectividad de posibles tratamientos.

Esta tesis recopila un trabajo multidisciplinario que incluye una variedad de métodos para análisis y tratamiento de imagen aplicado a aquellas que provienen de microscopía de fluorescencia y microscopía de campo claro.

Usando métodos avanzados de aprendizaje de máquina se pueden cuantificar biomarcadores que son usados para encontrar perfiles de expresión que se relacionan con la identidad y función de la célula o el tejido. Usando imágenes de campo claro hemos identificado la expresión proteínica proveniente de secciones consecutivas de tejido y los hemos fusionado por medio de registro de imágenes para cuantificar su co-expresión en el área del tumor en muestras de cáncer gástrico. En otra aplicación hemos usado la microscopía de fluorescencia para asociar la expresión de marcadores con la identidad de la célula. Esta identidad, es luego usada para analizar el vecindario celular, el cual puede ser asociado a mecanismos específicos de enfermedades, en este caso, glioma. Para ello usamos ensambles de modelos de aprendizaje de máquina para realizar la clasificación de las células, incrementando la eficiencia con respecto a un modelo simple y para obtener una medida de confianza sobre las predicciones.

De igual manera, en esta tesis se incluye la creación de una herramienta de visualización que permite, además de la exploración de imágenes provenientes de microscopios, la superposición de información espacial extraída de ellas. Estas imágenes son de gran tamaño (más de 10000x10000 px) y generan retos tecnológicos que requieren métodos avanzados en procesamiento y visualización de imágenes. Esta herramienta permite la visualización de millones de puntos de expresión genética superpuestos sobre las imágenes originales lo que permite el estudio contextual del tejido a nivel celular y subcelular. Esta herramienta fue utilizada para la visualización de tejido de cáncer de próstata y los correspondientes grados de agresividad, resultado de métodos de aprendizaje profundo. También para realizar anotaciones sincronizadas en imágenes consecutivas de tejido para encontrar regiones de interés correspondientes.

En conclusión, los métodos y herramientas presentados proveen el marco de referencia para analizar y visualizar vastos complejos estructurales en el tejido. Estos desarrollos en el entendimiento de la información espacial en el tejido de diferentes enfermedades pavimentará el camino hacia nuevos descubrimientos y la mejora de los tratamientos de los pacientes.

Acknowledgements

I would not be here today had it not been for the support of a huge number of people. So many, that I apologize if I can't name you all and if you think I left you out, do tell me, and I shall invite you to fika!

I thank my role model, my supervisor Carolina Wählby for believing in me and giving me the opportunity of a lifetime, working with you has been a true honor and pleasure, thank you for your patience, guidance and lessons on how to be a good and honest researcher. My time with you and the micro-team and Vi2 has been the most important of my life.

Life is a roller coaster and I thank lady luck for putting us together in the same time and place, Kristina Lidayova (Kika) so that one day you would send this opportunity my way.

I would like to thank Nataša Sladoje and Joakim Lindblad for being great professors of the more theoretical aspects of image analysis and machine learning and for hiring Johan Öfverstedt. Without knowing, you brought to me my kindred spirit, my partner in crime. My love, I do not know what would've been of me without your support in so many ways, you've shown me love, happiness and *formality*. Thank you for the music, the books, the movies, the series, the papers, the deep conversations.

To all my fellow PhDs, the clan, Gabriele Partel, Nadezhda Koriakina, Raphaela Heil, Håkan Wieslander, Ankit Gupta, Eva Breznik, Anindya Gupta, Karl Bengtsson Bernander, Elisabeth Wetzer, Eduard Chelebian, Erik Hallström, Nicolas Pielawski master of profound learning, Kimmo Kartasalo digital pathology extraordinaire, Axel Andersson fellow gamer, Andrea Beháňová you're a beautiful soul, Natalia Calvo we are like yin and yang, thank you for your friendship, Fredrik Nysjö you're my CG hero, Teo Asplund, Anders Persson, Maike Paetzel, Kalyan Ram, Ekta Vats. From other wonderful places in the world, Mariëlle Jansen from Utrecht, Daniela Vorkel in Dresden. Valentyna Zinchenko in Heidelberg. Also my friends from Geocentrum Chiara Költringer, Claudia Abril and Tatiana Pertuz. Thank you all for insightful, fun and interesting conversations, it's been fun to play with your data as well.

Dear professors who have been an inspiration in different ways, Gunilla Borgefors, who not only taught us rigor and formality, also has taught us lessons, in art, geometry, music, journalism, literature, and gender equality. My cosupervisors: Ida-Maria Sintorn thank you for your lessons on electron microscopy, thank you Petter Ranefall for your lessons in software and research. For your guidance during the PhD: Ingela Nyström , Robin Strand.

Filip Malmberg, Ewert Bengtsson, Anders Hast, Mikael Laaksoharju. Seniors and post-docs Maxime Bombrun, Chirstophe Avenel, Amin Allalou, Hanqing Zhang. Thank you Anna Klemm for your friendship, kindness, leadership and lessons. Thank you Fred Hamprecht for opening the doors to all the group in Heidelberg giving us all a great opportunity to meet researchers in EMBL.

To my collaborators Arne Östman and Lina Wik, for your patience, explanations and interesting work. To all the collaborators I met from the I3S in Porto. Carla Pereira for our work and time together.

Thanks to Åsa Cajander, Ginevra Castellano and the whole equal opportunities and gender equality group. Thank you to Astrid Raidl you are an inspiring Linux Administrator and creative spirit.

Thank you to Anna-Lena Forsberg and Elisabeth Lindqvist for all your support about human resources, and documentation. I don't know where I would be without all that knowledge. Thank you to Ulrik Ryberg from Rullan for keeping us fed!

Life outside of university might seem impossible but thanks to serendipitous coincidences I managed to procure the most wonderful auntie, directly from Argentina, Sara España has been a great friend and guide. Thank you for taking care of me, making me part of your family and introducing me to Jadwiga L.K, who has been a nice friend and sculpting buddy. Thank you for introducing me to Josefine Kron who devoted week after week to teach me Swedish. Thanks to Li Hedenmalm and Saga Samuelsson and the syjunta group. Thank you to my musical friend, Ricardo Gonzalez for rushing to my aid and translating to swedish. Thank you Juan Pablo Siza for correcting my spanish summary. Thanks to the choir for all the music that reminds me of Latin America.

Thank you to all my friends all over the world who kept cheering me on and have always been there to listen, through the good ones and the bad ones. Thanks to my in-laws, my adoptive Swedish family. To my family back in Colombia, my father, the kindest soul I have ever met, thank you for encouraging me and telling me you're proud of me.

References

[1] Erik Meijering, Anne E Carpenter, Hanchuan Peng, Fred A Hamprecht, and Jean-Christophe Olivo-Marin. Imagining the future of bioimage analysis. *Nature Biotechnology*, 34(12):1250–1255, dec 2016.

[2] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[3] Edward R. Tufte. *The Visual Display of Quantitative Information*. ISBN 096139210X. Graphics Press, USA, 1986.

[4] Bernard Valeur and Mário Nuno Berberan Santos. *Fluorescence Microscopy*, chapter 11, pages 327–348. John Wiley & Sons, Ltd, 2012.

[5] Jan Kapuscinski. Dapi: a DNA-specific fluorescent probe. *Biotechnic & Histochemistry*, 70(5):220–233, 1995.

[6] Wei Chang Colin Tan, Sanjna Nilesh Nerurkar, Hai Yun Cai, Harry Ho Man Ng, Duoduo Wu, Yu Ting Felicia Wee, Jeffrey Chun Tatt Lim, Joe Yeong, and Tony Kiat Hon Lim. Overview of multiplex immunohistochemistry / immunofluorescence techniques in the era of cancer immunotherapy. *Cancer Communications*, 40(4):135–153, apr 2020.

[7] Edwin Parra, Alejandro Francisco-Cruz, and Ignacio Wistuba. State-of-the-art of profiling immune contexture in the era of multiplexed staining and digital analysis to study paraffin tumor tissues. *Cancers*, 11(2):247, feb 2019.

[8] Yury Goltsev, Nikolay Samusik, et al. Deep profiling of mouse splenic architecture with CODEX multiplexed imaging. *Cell*, 174(4), 2018.

[9] Rongqin Ke, Marco Mignardi, Alexandra Pacureanu, Jessica Svedlund, Johan Botling, Carolina Wählby, and Mats Nilsson. In situ sequencing for RNA analysis in preserved tissue and cells. *Nature Methods*, 10(9):857–860, jul 2013.

[10] Veronique Kiermer. Antibodypedia. *Nature Methods*, 5(10):860–860, oct 2008.

[11] Famke Aeffner, MarkD Zarella, Nathan Buchbinder, MarilynM Bui, MatthewR Goodman, DouglasJ Hartman, GiovanniM Lujan, MariamA Molani, AnilV Parwani, Kate Lillard, OliverC Turner, VenkataN P Vemuri, AnaG Yuil-Valdes, and Douglas Bowman. Introduction to digital image analysis in whole-slide imaging: A white paper from the digital pathology association. *Journal of Pathology Informatics*, 10(1):9, 2019.

[12] M.N. Gurcan, L.E. Boucheron, A. Can, A. Madabhushi, N.M. Rajpoot, and B. Yener. Histopathological image analysis: A review. *IEEE Reviews in Biomedical Engineering*, 2:147–171, 2009.

[13] Jia-Ren Lin, Benjamin Izar, Shu Wang, Clarence Yapp, Shaolin Mei, Parin M Shah, Sandro Santagata, and Peter K Sorger. Highly multiplexed immunofluorescence imaging of human tissues and tumors using t-CyCIF and conventional optical microscopes. *eLife*, 7, jul 2018.

[14] Erik A. Burlingame, Mary McDonnell, Geoffrey F. Schau, Guillaume Thibault, Christian Lanciault, Terry Morgan, Brett E. Johnson, Christopher Corless,

Joe W. Gray, and Young Hwan Chang. SHIFT: speedy histological-to-immunofluorescent translation of a tumor signature enabled by deep learning. *Scientific Reports*, 10(1), oct 2020.

[15] Robert Serafin, Weisi Xie, Adam K. Glaser, and Jonathan T. C. Liu. FalseColor-python: A rapid intensity-leveling and digital-staining package for fluorescence-based slide-free digital pathology. *PLOS ONE*, 15(10), 2020.

[16] Christel Daniel, François Macary, Marcial García Rojo, Jacques Klossa, Arvydas Laurinavičius, Bruce A. Beckwith, and Vincenzo Della Mea. Recent advances in standards for collaborative digital anatomic pathology. *Diagnostic Pathology*, 6(1):S17, Mar 2011.

[17] Albert Cardona and Pavel Tomancak. Current challenges in open-source bioimage informatics. *Nature Methods*, 9(7):661–665, jun 2012.

[18] Melissa Linkert, Curtis T. Rueden, Chris Allan, Jean-Marie Burel, Will Moore, Andrew Patterson, Brian Loranger, Josh Moore, Carlos Neves, Donald MacDonald, Aleksandra Tarkowska, Caitlin Sticco, Emma Hill, Mike Rossner, Kevin W. Eliceiri, and Jason R. Swedlow. Metadata matters: access to image data in the real world. *Journal of Cell Biology*, 189(5):777–782, may 2010.

[19] Sébastien Besson, Roger Leigh, Melissa Linkert, Chris Allan, Jean-Marie Burel, Mark Carroll, David Gault, Riad Gozim, Simon Li, Dominik Lindner, Josh Moore, Will Moore, Petr Walczysko, Frances Wong, and Jason R. Swedlow. Bringing open data to whole slide imaging. In Constantino Carlos Reyes-Aldasoro, Andrew Janowczyk, Mitko Veta, Peter Bankhead, and Korsuk Sirinukunwattana, editors, *Digital Pathology*, pages 3–10. Springer International Publishing, 2019.

[20] Mahadev Satyanarayanan, Adam Goode, Benjamin Gilbert, Jan Harkes, and Drazen Jukic. OpenSlide: A vendor-neutral software foundation for digital pathology. *Journal of Pathology Informatics*, 4(1):27, 2013.

[21] Ilya G Goldberg, Chris Allan, Jean-Marie Burel, Doug Creager, Andrea Falconi, Harry Hochheiser, Josiah Johnston, Jeff Mellen, Peter K Sorger, and Jason R Swedlow. The open microscopy environment (OME) data model and xml file: open tools for informatics and quantitative analysis in biological imaging. *Genome Biology*, 6(5):R47, 2005.

[22] Stanislav Melnikov. Slideio, a python module for the reading of medical images.

[23] Kirk Martinez and John Cupitt. Vips - a highly tuned image processing software architecture. In *IEEE International Conference on Image Processing 2005*, volume 2, pages II–574, 2005.

[24] Michael Appleby, Tom Crane, Robert Sanderson, Jon Stroop, and Simeon Warner. IIIF image API 3.0.

[25] Ian Gilman. An open-source, web-based viewer for high-resolution images.

[26] David Urbanic. Zoom-and-pan for fast, interactive viewing on the web with html, jpegs, and javascript.

[27] Tim Schaub, Andreas Hocevar, Tom Payne, and Frédéric Junod. A high-performance, feature-packed library for all your mapping needs.

[28] Carolina Wählby. Image segmentation, processing and analysis in microscopy and life science. In *Mathematical Models in Biology*, pages 1–16. Springer International Publishing, 2015.

[29] J. Morgan and J. Sonquist. Problems in the analysis of survey data, and a

proposal. *Journal of the American Statistical Association*, 58:415–434, 1963.
[30] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science*, pages 234–241. Springer International Publishing, 2015.
[31] Simon Jégou, Michal Drozdzal, David Vazquez, Adriana Romero, and Yoshua Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation, 2017.
[32] David Marr and Ellen Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 1980.
[33] Peter Bankhead, Maurice B. Loughrey, José A. Fernández, Yvonne Dombrowski, Darragh G. McArt, Philip D. Dunne, Stephen McQuaid, Ronan T. Gray, Liam J. Murray, Helen G. Coleman, Jacqueline A. James, Manuel Salto-Tellez, and Peter W. Hamilton. QuPath: Open source software for digital pathology image analysis. *Scientific Reports*, 7(1), dec 2017.
[34] Anna Kreshuk and Chong Zhang. Machine learning: Advanced image segmentation using ilastik. In *Computer Optimized Microscopy*, pages 449–463. Springer New York, 2019.
[35] Barbara Zitová and Jan Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977–1000, 2003.
[36] Max A. Viergever, J.B. Antoine Maintz, Stefan Klein, Keelin Murphy, Marius Staring, and Josien P.W. Pluim. A survey of medical image registration â under review. *Medical Image Analysis*, 33:140–144, 2016. 20th anniversary of the Medical Image Analysis journal (MedIA).
[37] Johan Öfverstedt, Joakim Lindblad, and Natasa Sladoje. Inspire: Intensity and spatial information-based deformable image registration, 2020.
[38] J. Öfverstedt, J. Lindblad, and N. Sladoje. Fast and robust symmetric image registration based on distances combining intensity and spatial information. *IEEE Transactions on Image Processing*, 28(7):3584–3597, 2019.
[39] David. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999.
[40] Thomas de Bel, John-Melle Bokhorst, Jeroen van der Laak, and Geert Litjens. Residual cyclegan for robust domain transformation of histopathological tissue slides. *Medical Image Analysis*, 70:102004, 2021.
[41] Wouter Bulten, Péter Bándi, Jeffrey Hoven, Rob van de Loo, Johannes Lotz, Nick Weiss, Jeroen van der Laak, Bram van Ginneken, Christina Hulsbergen van de Kaa, and Geert Litjens. Epithelium segmentation using deep learning in H&E-stained prostate specimens with immunohistochemistry as reference standard. *Scientific Reports*, 9(1), jan 2019.
[42] Matthew McCormick, Xiaoxiao Liu, Luis Ibanez, Julien Jomier, and Charles Marion. Itk: enabling reproducible research and open science. *Frontiers in Neuroinformatics*, 8:13, 2014.
[43] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.
[44] Stefan Klein, Marius Staring, Keelin Murphy, Max A. Viergever, and Josien P. W. Pluim. elastix: A toolbox for intensity-based medical image registration. *IEEE Transactions on Medical Imaging*, 29(1):196–205, 2010.

[45] Caroline A. Schneider, Wayne S. Rasband, and Kevin W. Eliceiri. Nih image to imagej: 25 years of image analysis. *Nature Methods*, 9(7):671–675, Jul 2012.

[46] Caroline A. Schneider, Wayne S. Rasband, Kevin W. Eliceiri, Johannes Schindelin, Ignacio Arganda-Carreras, Erwin Frise, Verena Kaynig, Mark Longair, Tobias Pietzsch, Stephan Preibisch, Curtis Rueden, Stephan Saalfeld, Benjamin Schmid, Jean-Yves Tinevez, Daniel James White, Volker Hartenstein, Kevin Eliceiri, Pavel Tomancak, and Albert Cardona. Nih image to imagej: 25 years of image analysis. *Nature Methods*, 9(7):671–675aTY – JOUR, Jul 2012.

[47] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with numpy. *Nature*, 585(7825):357–362, Sep 2020.

[48] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

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

[50] Claire McQuin, Allen Goodman, Vasiliy Chernyshev, Lee Kamentsky, Beth A. Cimini, Kyle W. Karhohs, Minh Doan, Liya Ding, Susanne M. Rafelski, Derek Thirstrup, Winfried Wiegraebe, Shantanu Singh, Tim Becker, Juan C. Caicedo, and Anne E. Carpenter. CellProfiler 3.0: Next-generation image processing for biology. *PLOS Biology*, 16(7):e2005970, jul 2018.

[51] Thouis R. Jones, In Han Kang, Douglas B. Wheeler, Robert A. Lindquist, Adam Papallo, David M. Sabatini, Polina Golland, and Anne E. Carpenter. CellProfiler Analyst: data exploration and analysis software for complex image-based screens. *BMC Bioinformatics*, 9(1), Nov 2008.

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

[53] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[54] Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with PyTorch Geometric, 2019.

[55] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[56] David Cournapeau and Matthieu Brucher. Scikit-learn documentation. 2.3

clustering.
[57] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2):107–145, Dec 2001.
[58] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. The Wadsworth and Brooks/Cole statistics/probability series. Wadsworth and Brooks/Cole Advanced Books and Software, Monterey, CA, 1984.
[59] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. AAAI Press, 1996.
[60] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.
[61] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.
[62] Steve Harenberg, Gonzalo Bello, L. Gjeltema, Stephen Ranshous, Jitendra Harlalka, Ramona Seay, Kanchana Padmanabhan, and Nagiza Samatova. Community detection in large-scale networks: a survey and empirical evaluation. *WIREs Computational Statistics*, 6(6):426–439, 2014.
[63] V. A. Traag, L. Waltman, and N. J. van Eck. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports*, 2019.
[64] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008.
[65] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015.
[66] Gabriele Partel and Carolina Wählby. Spage2vec: Unsupervised representation of localized spatial gene expression signatures. *The FEBS Journal*, 2020.
[67] Erich Novak. High dimensional numerical problems. *Nonlinear Analysis: Theory, Methods and Applications*, 30(3):1439–1446, 1997.
[68] Kevin S. Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is ”nearest neighbor” meaningful? ICDT ’99, page 217â235, Berlin, Heidelberg, 1999. Springer-Verlag.
[69] M. Espadoto, R. M. Martins, A. Kerren, N. S. T. Hirata, and A. C. Telea. Toward a quantitative survey of dimension reduction techniques. *IEEE Transactions on Visualization and Computer Graphics*, 27(3):2153–2173, 2021.
[70] Leland McInnes, John Healy, and James Melville. UMAP: Uniform manifold approximation and projection for dimension reduction, 2020.
[71] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 1117(9):2579–2605, 2019.
[72] Martin Wattenberg, Fernanda Viégas, and Ian Johnson. How to use t-SNE effectively. *Distill*, 2016.
[73] Ding-Zhu Du, Panos M. Pardalos, and Weili Wu. *History of optimizationHistory of Optimization*, pages 1538–1542. Springer US, Boston, MA, 2009.

[74] Jeffrey M. Stanton. Galton, Pearson, and the peas: A brief history of linear regression for statistics instructors. *Journal of Statistics Education*, 2001.
[75] Wei-Yin Loh. Fifty years of classification and regression trees. *International Statistical Review*, 82(3):329–348, 2014.
[76] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754, 2016.
[77] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.
[78] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry, 2017.
[79] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs, 2018.
[80] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.
[81] Tyler Vigen. *Spurious Correlations*. Hachette Books, 2015.
[82] Alexandru C. Telea. *Data Visualization: Principles and Practice*. ISBN 978-1466585263. CRC PR INC, 09 2014.
[83] Marco Mignardi, Omer Ishaq, Xiaoyan Qian, and Carolina Wahlby. Bridging histology and bioinformatics—computational analysis of spatially resolved transcriptomics. *Proceedings of the IEEE*, pages 1–12, 2016.
[84] Darel Rex Finley. Determining whether a point is inside a complex polygon.
[85] Yan Holtz. The d3.js graph gallery: a collection of simple charts made with data driven documents (d3).
[86] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.
[87] Michael Waskom and the seaborn development team. Seaborn: statistical data visualization, September 2020.
[88] Bokeh Development Team and sponsors. Bokeh: Python library for interactive visualization, 2018.
[89] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.
[90] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3 Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011.
[91] Ton Roosendaal and Blender Community. Blender - open source 3D creation. free to use for any purpose, forever, 2021.
[92] Jon Leech and Benj Lipchak. Webgl API. a low-level, shader based, 3d graphics API based on opengl es 2.0 that renders directly into an html5 canvas element.
[93] Armin Ronacher. Flask. A lightweight WSGI web application framework.
[94] Christopher Rydell and Joakim Lindblad. CytoBrowser: a browser-based collaborative annotation platform for whole slide images. *F1000Research*, 10(226), 2021.

Acta Universitatis Upsaliensis

Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology 2025

Editor: The Dean of the Faculty of Science and Technology

A doctoral dissertation from the Faculty of Science and Technology, Uppsala University, is usually a summary of a number of papers. A few copies of the complete dissertation are kept at major Swedish research libraries, while the summary alone is distributed internationally through the series Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology. (Prior to January, 2005, the series was published under the title "Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology".)

Distribution: publications.uu.se
urn:nbn:se:uu:diva-438775

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2021