Aspects of the analysis of cell imagery: from shape to understanding

Li, C.

Aspects of the Analysis of Cell Imagery: from Shape to Understanding

Proefschrift

ter verkrijging van
de graad van doctor aan de Universiteit Leiden,
op gezag van rector magnificus prof.dr.ir. H. Bijl,
volgens besluit van het college voor promoties
te verdedigen op donderdag 27 juni 2024
klokke 12:30 uur

door

Chen Li

geboren te Shandong, China

in 1992

Promotor:
Prof. Dr. Ir. F.J. Verbeek

Co-promotor:
Dr. L. Cao

Promotiecommissie:
Prof. Dr. M.M. Bonsangue
Prof. Dr. T.H.W. Bäck
Prof. Dr. K.J. Batenburg
Prof. Dr. B. Gravendeel (Radboud universiteit)
Dr. M. Frazão Muzitano (Universidade Federal do Rio de Janeiro)

ISBN: 978-94-6506-136-8
Cover designed by Chen Li.
Printed by Ridderprint, the Netherlands.

The research presented in this dissertation was financially supported by the China Scholarship Council (CSC), Grant No. 201806350256.

Contents

Chapter 1

Introduction

Bioimaging is the study of making images from biological samples. The biological samples that are investigated basically vary from cells, tissues, organs, etc, with a resolution from micrometers to centimeters. To make images of those small samples, imaging instruments, i.e. microscopes, are required. A huge range of different microscopes and techniques have become available over time [1]. Hence, the selection of a microscope technique depends on the level of details that you want to obtain from small samples.

In this thesis, we intend to study cell image data with multi-dimensional characteristics. In order to guarantee an accurate analysis subsequently with computational tools, data acquisition techniques should be good enough to acquire as many details as possible. Several requirements need to be taken into account and for computational analysis, the volume of the data set is very important. A sufficient amount of data can provide diverse information that advances the computational procedures. Moreover, image resolution/quality determines if we can gain sufficient details from biological samples. With these requirements, we start our work.

The data we are considering is not just flat cells. Other than *in vitro* imaging, we look at *in vivo* cell data and try to find a microscopy technique that fits them to capture images. We have two particular classes of cell data. One is pollen grains, which are imaged with a brightfield microscope. The other one is the immune cells, in particular neutrophils, and these are imaged by a confocal microscope.

So we study a gamete cell and a cell that dynamically moves in the body. In order to analyse these cells we need 3D images or 3D images over time. So, the subjects of research are captured in multi-dimensional images.

Pollen types are as diverse as the plants that produce them. We specifically look at airborne pollen. Airborne pollen encompass the reproductive cells, gametes, generated from flowering plants. Some of these are causing issues, i.e. health problems for individuals

suffering from hay fever, leading to decreased productivity and increased healthcare costs [2]. In recent decades, the incidence of hay fever patients is rising worldwide, which has raised awareness of some institutions towards problems. Therefore, monitoring airborne pollen and identifying pollen taxa is becoming more and more important. For medical research in hay fever, this has quite an impact. It contributes in forecasting allergies and providing potential medicine treatments to alleviate allergy numbers. Pollen category identification based on distinguished family/genus levels is possible using microscopic methods. However, for pollen with quite similar morphology within the same species even genera, computational algorithms and strategies are essential, specifically to identify the allergenic pollen in a sample. In this thesis, we elaborate on a case study for a particular plant family. Pollen grains are such a nice object for study because they are isolated cells, one just has to identify each pollen grain from a sample.

Contrary to pollen grains, we study neutrophils *in vivo* in the body. In order to be able to do that we need a model system, and for that, zebrafish is very suitable. Neutrophils are a type of innate immune cell and play a role in the protection of organisms. They are involved in combating infections caused by bacteria, viruses, and other pathogens. The study of neutrophils in zebrafish aims to unravel the mechanisms behind various diseases, including autoimmune disorders, cancer, and infectious diseases [3][4][5]. We study the neutrophils as part of the innate immune system, therefore, we look at the early developmental stages of zebrafish; i.e. larvae stages. In this stage of development the zebrafish are transparent. This transparency allows for high-resolution imaging, providing detailed insights into the entire organism's internal structures, including the neutrophils. If this event that challenges the integrity of the organism occurs, then the neutrophils are triggered to play a role in maintaining the integrity. Thus neutrophils will move to where they are necessary to play a role in neutralizing intruders. In order to visualize this movement in the body, we need to capture 3D images over time. Subsequent to studying the behavior of neutrophils, cell tracking techniques, are required.

In this thesis, we intend to find the solutions for both problems. It holds that we are to find specific classes in the data. These are two typical cases, for which we acquire images with a microscope and get sufficient images to find the pattern that we are interested in.

1.1 Microscopy Imaging

Microscopes are the "eyes" that can observe the things that are not visible to the naked eye [6]. With the help of a microscope, we can see the structure of various proteins, viruses,

bacteria, cells, tissues and organs. It helps us to study the function of organisms so as to explore the mysteries in the world of living organisms.

In order to view the organisms at different scales, many types of microscopes are designed. The most widely used types of microscopes are optical microscopes and electron microscopes [7]. Each kind of these microscopes can see organisms in the range from nanometer to micrometer or millimeter [6][8], depending on the configurations being set. Fig. 1.1 shows the size scale of the different organisms and the corresponding applicable microscope.

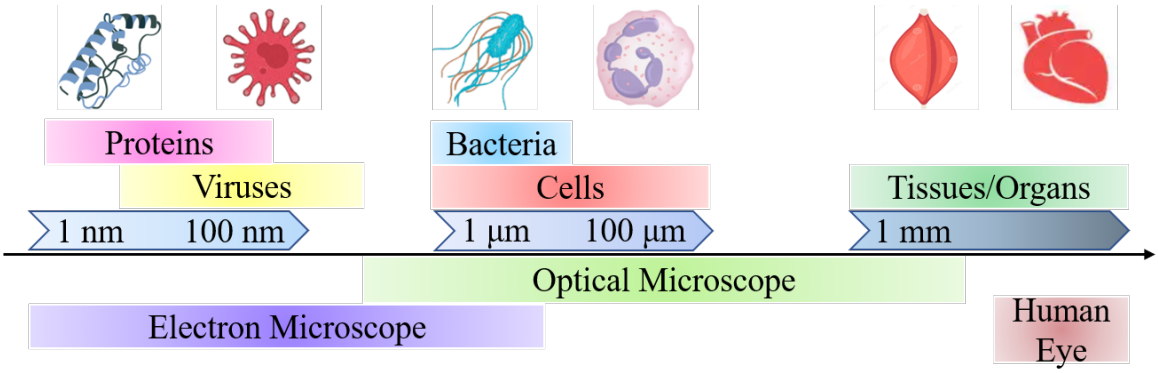


Fig. 1.1 The size scale of the organisms and the corresponding microscope applicable.

Optical microscopes, use visible light to illuminate the specimen and magnify the small samples. It is the most commonly used microscopy type due to the large observation range from nanometers to millimeters. Normally, optical microscopes are focused on studies of cellular-level samples. In the research mentioned in this thesis, we aim to seek computational approaches to analyze cellular-level microscopy images in multi-dimensions, from cells of pollen grain and neutrophil.

The isolated pollen cell is suitable to be imaged with a brightfield imaging technique using a light microscope. Because it uses visible light at wavelengths from approximately 400 to 700 nanometers and pollen can be seen exactly within this wavelength range. Fig. 1.2 (a) shows the light microscope system that was used in capturing our pollen images. For our research project, pollen grains are on a slide and that is scanned. On each slide, a list of XY positions of pollen is given, which is shown in Fig. 1.2 (b); it indicates where a pollen grain is on the slide. Given the positions with an automated stage, at each position a high-resolution image acquisition is realized. In our set-up this is done with the Zeiss software Zen BLUE. For each pollen grain, focusing through different focal levels automatically produces images per focal plane. In this manner, a Z-stack image representing the 3D structure is captured.

In contrast to a static pollen grain, neutrophils move through the zebrafish body. Moreover, neutrophils are fluorescently labeled through a green fluorescent protein (GFP). It needs to

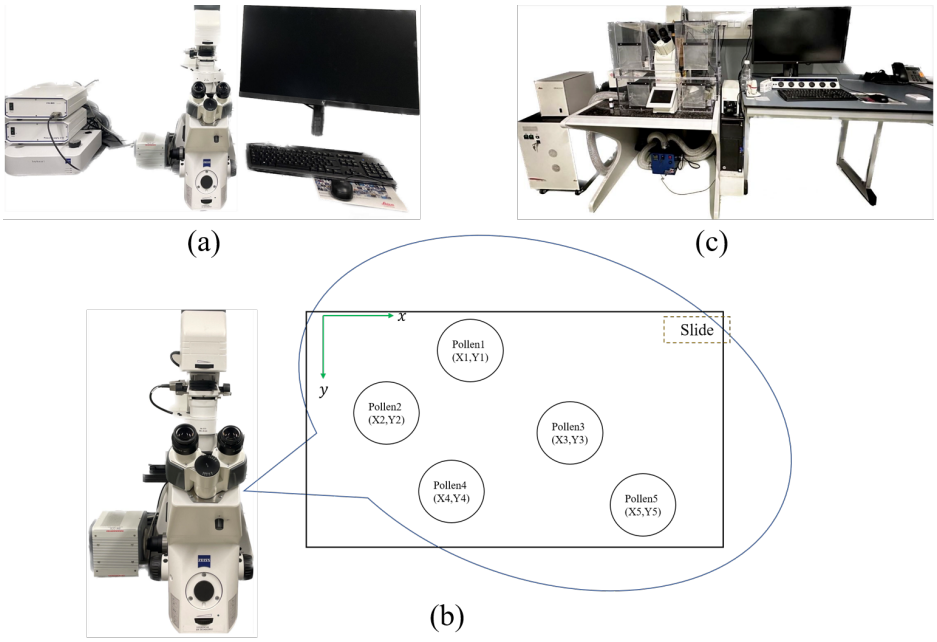


Fig. 1.2 Microscopes. (a) A light microscope system that is used to capture pollen images; (b) An example of the XY positions of each pollen grain is given on a slide with a light microscope; (c) A confocal microscope system that is used to capture neutrophil images.

be excited with blue light and emitted green because of the GFP. For this specific light data, a confocal microscope can utilize specific wavelengths to highlight GFP-labeled neutrophils in zebrafish. Other than a conventional microscope, confocal laser scanning microscopy (CLSM) is scanning a sample with a point source (a laser) that excites the sample locally and then registers the emitted light. The confocal principle realizes a thin optical section and, by moving a focal plane, a stack of optical sections is obtained. Therefore, in this thesis, we utilize a confocal microscope to capture 3D images of neutrophils in the zebrafish over time. The confocal microscope system is shown in Fig. 1.2 (c).

A high resolution/quality image ensures to provide sufficient detail and these selected microscopes fulfill the requirements to make these images. It subsequently helps us look for patterns in these collected image data.

1.2 Data Patterns

The pollen grains investigated in our studies are from a particular plant family named Urticaceae. They are typically small and spherical to ovoid in shape. The outer surface of the pollen grain, called the exine, can vary in texture and ornamentation between different genus/species within the Urticaceae family. This variation, however, is not easy to distinguish

even under a microscope due to its highly similar characteristics. In the research presented in this thesis, we intend to identify pollen patterns from pollen images, including their size, shape, and surface patterns, so that we can categorize each image into different classes accurately.

Neutrophils have irregular deformation and move in the body. To track the trajectory of each moving cell over time, the representative patterns that can distinguish the difference between a tracked cell and candidate cells should be identified, including cell morphological changes, cell migrated direction, cell moving speed, etc. Identifying these patterns of neutrophils from 3D images over time helps to localize the position of each cell and link them to a trajectory.

Based on the patterns of two classes of data, the design of machine learning techniques plays an important role to be able to correctly and automatically analyze, interpret and understand these cell data.

1.3 Microscope Image Analysis with Machine Learning

With respect to understanding the patterns from these cell images, not only qualitative but also quantitative measurements are essential. Just depending on human input to analyze images is not feasible due to large amounts of data, we have to develop techniques that can perform image analysis tasks automatically and very accurately [9]. The study of algorithms and methods to train computers to analyze, interpret, and understand visual data is referred to as computer vision. It aims to simulate part of the complexity of the human visual system and processes visual data in similar fashion to humans. Thanks to the progresses in the field of computer vision, the use of advanced computational methodologies to analyze and extract meaningful information from microscopy image data has been successfully applied [10][11][12][13][14]. In general, the main tasks that can be conducted with those data focus on the sub-domains that include, but are not restricted to, object detection, tracking, reconstruction, motion estimation, modeling, etc [15]. The field of bio-image analysis has been able to take big steps using wide-spread and common machine learning approaches; nowadays especially the innovations in deep learning [16][17].

Machine learning approaches include traditional machine learning techniques and deep learning convolutional neural networks (CNNs), [18] which is illustrated in Fig. 1.3 (a). Traditional machine learning techniques are algorithms that typically rely on handcrafted features and require extensive domain experience and expertise. These algorithms are based on explicit mathematical models so that they are better interpretable. They require a relatively small amounts of data and can perform well with those limited data. As a comparison, deep

learning CNNs consist of numerous interconnected layers with a huge number of parameters and are considered as "black boxes". From this structure, representative features can be directly learned from raw data during the training process, thus reducing the need for explicit feature extraction. However, the "black boxes" are less interpretable, and understanding the exact theoretical process can be challenging. Moreover, in order to be efficient, training deep learning CNNs requires huge amounts of data to achieve satisfactory performance, which has been demonstrated on various tasks [19]. Therefore, the selection of the two approaches often depends on the specific problem, available data, interpretability, and complexity requirements.

In this research, the applicability of machine learning strategies for bioimaging needs to be studied further; especially deep learning as compared to conventional machine learning. The most important applications we recognize are segmentation, basically a pixel classification problem, and classification over features obtained from images. In bio-imaging, images of cells are a dominant field for analysis. Therefore, investigating how machine learning can be integrated in workflows for analysis of these images is important for the understanding of analysis; this holds for both 2D and 3D image analysis.

In our studies we have chosen to work with images of cells that are known to be analyzed with 2D images whereas there is a 3D counterpart that can be addressed as well. Moreover, we might even consider images with a time-dimension for the analysis. As we will explain further, we have selected two classes of cells that can be studied and analyzed in 2D and 3D. These are pollen grains and immune cells.

In short, the methodology research in this thesis focuses on both traditional machine learning techniques and deep learning CNNs, and how CNNs can be efficiently applied to our kind of microscope images. The combination allows for a comprehensive analysis of the acquired cell data. Two applications are designed and implemented, they are the classification of pollen from images and neutrophil tracking (segmentation included) in time-lapse sequences. Fig. 1.3 (b) presents a brief diagram of the themes related to microscope image analysis that are addressed in this thesis. We expect to achieve accurate and reliable results, providing valuable insights into cell patterns, behaviors, and interactions.

1.3.1 Ground Truth Data

Ground truth data is essential for learning patterns using machine learning techniques as shown in Fig. 1.3 (b). It is defined as accurate and reliable information about the target variable or outcome that a model is trying to predict, which is considered to be real or true. It is often used to train, validate, and test a learning-based model, as well as evaluate the performance of a model [20]. In practice, the ground truth data is annotated manually controlled by experts based on their expertise. A convincing ground truth data is based on the

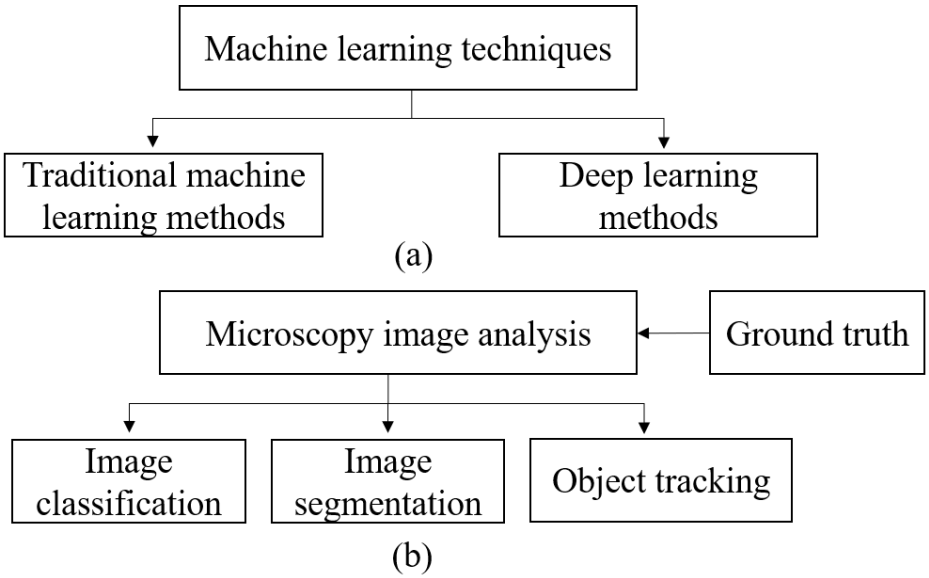


Fig. 1.3 The diagram of (a) components of machine learning techniques and (b) investigated themes in microscopy image analysis in this thesis.

annotation from different experts with several rounds of repetition. The annotation process is a laborious and time-consuming endeavor, especially for complex data. At present, some biomedical research work focuses on developing algorithms from public datasets which are already annotated and thus represent ground truth. These data sets are often employed in competitions [20][21][22]. For instance, the cell tracking challenge [23] which is held under the auspices of the IEEE International Symposium on Biomedical Imaging (ISBI), provides ground truth data for both segmentation and tracking of real cell data as well as synthetic data. It enables the evaluation of each state-of-art based on the same benchmark and allows the researchers to advance their studies conveniently and straightforwardly. However, in many cases, researchers are dedicated to designing a computational algorithm for a dataset with a special condition and experimental design. It requires additional labeling of ground truth data. In fact, the difficulties of annotating the ground truth data differ from various microscopy image analysis tasks. With respect to a classification task, the annotation is to decide which category each image or parts thereof belongs to. It is a relatively easy annotation. While a segmentation annotation is to identify the object(s) of interest and separately outline or mask each object in the image. For a complex study like 2D object tracking, a continuous trajectory of the object through a whole time-lapse sequence is required, and the positional information of each object on the trajectory needs to be recorded. It is particularly hard to conduct annotation in a 3D time-lapse sequence because annotating in the (X, Y, Z, T) axis requires visualization of 3D + Time space first, so as to localize the position of each

3D object through time frames. Due to the necessity of annotating the ground truth data, some annotation tools such as ImageJ/Fiji plugins [24] are developed as auxiliaries to reduce the burden and support labeling image segmentation masks or 2D tracking trajectories. To the best of our knowledge, there is still a lack of open-source tools to label 3D time-lapse data. Facing the potential challenge, we prepare the ground truth data in this thesis, with the assistance of different strategies and tools, to achieve an efficient annotation process. The details are given in each chapter.

1.3.2 Image Classification

Image classification aims to assign a label or category to an image based on its visual features [25]. Feature extraction is one of the essential steps to collect a pool of quantified features from the images, which are used as representations to distinguish between categories. Normally, the most relevant features are selected and the redundant features are removed, through a process of feature selection or dimensional reduction. A classifier is applied to calculate the probability that a certain image belongs to each class based on its representative features and determine the most likely category that an object from the image should be assigned. In the past years, both traditional machine learning methods and innovative deep learning methods have achieved great accomplishments in image classification [26][27]. The traditional machine learning methods treat the classification with feature selection and classifier parts separately. It extracts handcrafted features based on expertise first and chooses a good classifier in order to maximize the classification performance. In contrast, deep learning methods conduct the two parts sequentially and automatically through a structure of neural networks, which saves lots of human effort and often gains superior performance [28].

A special case of classification is addressed in this thesis on our microscopy images of pollen grains to explore the insights of the pollen categories.

1.3.3 Image Segmentation

The goal of image segmentation is to partition the image into multiple components, normally the regions of interest and background, which is referred to as binary classification. Microscopy images contain more fine-grained details and structures that need to be segmented accurately. Besides, the image-capturing process introduces noise to the images. This noise is caused by factors such as illumination, photon, and sensor noise [29]. Noise may influence segmentation accuracy, due to the pixel-wise-based prediction of segmentation [30]. Therefore, advanced image segmentation methods are needed for microscopy images that consider the diverse characteristics of these images. In particular, those cell segmentation methods

take various cell properties into account, such as size, and morphology, in order to recognize cellular processes and mechanisms. In the literature, the proposed approaches mostly fall into the following aspects which include thresholding, edge-based methods, region-based methods, and machine learning-based methods [31][32]. Thresholding requires a threshold value determined manually or automatically to segment foreground and background, while edge-based methods detect the edges or boundaries of cells to segment them. Region-based methods divide the image into regions based on their similarity and use these regions to represent cells. Machine learning-based methods especially deep learning convolutional neural networks are used to learn features from the input image and classify each pixel as belonging to a cell or not. The selection of a segmentation method is important to precisely localize the position of the cell and perform in a robust manner in the presence of noise.

1.3.4 Object Tracking

In conjunction with image segmentation, object tracking is used subsequently to identify and track the movement of objects across frames in a sequence. This typically involves using algorithms to identify features or key points of the object and match them across frames to determine the object's movement [33]. With respect to the biomedical field, cell tracking is the process of monitoring and tracing the movement and behavior of cells over time. Cell tracking uses various techniques such as microscopy, data processing, image analysis, and machine learning, to study the behavioral mechanisms of cells. At present, cell tracking is quite a challenging task due to factors such as morphological deformations and complex motion patterns, depending on the types of cells [34][35]. There is no single general tracking algorithm that can deal with diverse types of cell tracking tasks well. Therefore, designing a pipeline tailored for tracking specific cell types in biomedical research is essential. With the recent development of machine learning techniques, cell tracking tasks are transformed from manual annotation to automatic tracing. It tremendously diminishes the labor effort and achieves satisfactory accuracy. However, there is still room for improvement. Furthermore, for a learning-based method, annotating ground truth data is another challenge as we illustrated beforehand. Thus, in this thesis, we develop original strategies to perform cell tracking for a neutrophil migration study.

1.4 Research Questions and Main Contributions

In section 1.2, we mentioned that two classes of multidimensional microscopy image datasets are acquired: (1) 3D images of individual pollen; (2) 3D images of neutrophil moving

over time. The main purpose of our work is to explore machine learning so that it can help understanding these cell images. To fulfill this, we focus on two applications: the classification of pollen from images and neutrophil tracking in time-lapse sequences.

1.4.1 Classification of Pollen from Images

The pollen investigated in this work is from two genera of the Urticaceae family, named *Parietaria* and *Urtica*. Why Urticaceae was selected as the research type of pollen and what is the challenge of this family? Urticaceae is one of the most common pollen types encountered in the Netherlands. In recent years, pollen seasons are prolonged due to climate changes. Hay fever allergies have strongly affected people's health and daily activities. Furthermore, the two genera of Urticaceae are morphologically highly similar but induce allergy at a very different level, which leads to difficulties in medical treatments. The pollen of *Urtica membranacea* is the only species that can be recognized easily within this family. Therefore, our goal is to find an accurate classification method to identify the pollen within three categories: *Parietaria*, *Urtica*, and *Urtica membranacea*. With the popularity and success of deep learning neural networks in various similar classification applications in recent years, we come to our first Research Question:

RQ 1: Can the existing deep learning-based classification models work with images from morphologically similar pollen grain of related species and what is the performance of the different models?

To answer the **RQ 1**, we have conducted three commonly used deep learning classification models; i.e. VGG16, MobileNet V1, and MobileNet V2 in **Chapter 2**. Deep learning models learn the image features automatically using the structure of convolutional neural networks (CNNs). CNNs acertain sufficient distinguished features to identify each category. In order to guarantee the accuracy of the classification model further, the first contribution from this work is:

C. 1: The data quality is improved by pre-processing the raw 3D pollen images into different 2D projection images.

The three projections are Standard Deviation (STD), Minimum Intensity (MIN), and Extend Focus (EXT). It aims to integrate as many representative pollen images as possible and use them as the input of the classification model, so as to increase the performance of the model. Another innovation is that the pollen grains that we used are unacetolyzed. Compared to acetolyzed pollen grains, in which all pollen materials are destroyed by acetolysis with the exception of sporopollenin that forms the outer pollen wall (a.k.a the exine), unacetolyzed

pollen keeps their original organic features which are less apparent. It leads to the second contribution:

C. 2: This is the first work to apply deep learning classification models and compare them for the analysis of the unacetolyzed Urticaceae pollen grains.

Except for the three mentioned deep learning models, traditional machine learning classification models such as Support Vector Machine (SVM) and Random Forest (RF) have been proven to achieve great results in pollen classification tasks as well. Thus, we come to the second Research Question:

RQ 2: How does the performance of the traditional machine learning-based classification models compare to that of deep learning-based models?

Instead of extracting features automatically like a deep learning model, we design and extract the handcrafted features manually and different classifiers are selected to perform the classification tasks on the pollen images. We also investigate more deep learning models such as AlexNet, VGG19, and ResNet50 as a comparison to extend our research.

In addition, training a deep learning model requires a large amount of data. Even for the traditional machine learning-based methods, the more data collected, the more raw variables that can be used as features, so as to improve the performance of each model. However, in most cases, collecting sufficient data in practice is difficult, especially for bio-medical data. Thus, the third research question is considered:

RQ 3: To what extent is it possible that both the traditional machine learning-based and the deep learning-based classification models perform well on a relatively small amount of data?

We investigate this question by implementing our classification models on two small-scale datasets. We are curious about how much accuracy can be achieved by traditional machine learning-based and deep learning-based methods, respectively.

Investigating **RQ 2** and **RQ 3** in **Chapter 3** has led us to have a thorough understanding of the mechanisms of different classification models, as well as explore the insights of different machine learning methods. It has materialized into our third contribution:

C. 3: A comprehensive and deep exploration of different strategies for pollen classification.

1.4.2 Neutrophil Tracking in Time-lapse Sequences

Contrary to the analysis of individual-cell image research such as pollen grains, neutrophils need to be investigated with a form of multicellular movement in a spatial-temporal domain.

Neutrophils act as the first line of defense against invading pathogens and protect organisms accordingly. In order to learn the migration patterns and analyze the behaviours and functional mechanisms of neutrophils, we formulate the fourth question:

RQ 4: To what extent is it possible to develop an automated algorithm that provide accurate support in the tracking of neutrophils from time-lapse sequences in the 2D spatial domain?

Through the fourth question, in **Chapter 4**, we investigate the whole pipeline to solve a tracking problem that contains three parts: cell segmentation, cell tracking, and trajectory linkage. To that end, we train and analyze several segmentation models that include both rule-based and deep learning-based approaches to localize the position of neutrophils first and attempt to learn the migration behaviours with a U-Net deep-learning model. Subsequently, We compare different linkage methods and propose an improved algorithm that tailors for the specific movement patterns of neutrophils. Correspondingly, we substantiate the fourth contribution:

C. 4: A thorough pipeline for the tracking of neutrophils in 2D time-lapse sequences. It solves the complex problems that occur in the migration of neutrophils.

Except for the analysis of the movement of neutrophils in time-lapse sequences of the 2D spatial domain, we continue to solve the tracking task in the 3D spatial domain which is de facto the real situation. Therefore, we move our focus to the fifth research question in **Chapter 5**:

RQ 5: To what extent is it possible to develop an automated algorithm that provide accurate support in the track of neutrophils from time-lapse sequences in the 3D spatial domain?

Compared with **RQ 4**, elaborating **RQ 5** with a strategy of deep learning-based method is more difficult. This is because a deep learning model requires large amounts of annotations. However, labeling cell trajectory in 3D space along with a time axis is complex and laborious work. Thus, an effective and efficient tracking algorithm is still essential. It gives rise to our last contribution:

C. 5: A feature-weighted tracking method for tracking neutrophils' movement in 3D time-lapse sequences.

1.5 Thesis Structure

The thesis is structured based on the research question presented in the previous paragraphs. A series of publications that have been peer-reviewed and published are listed to support our work in each chapter, respectively.

Chapter 1: "Introduction" a brief introduction on the research background of bioimaging, and microscopy technology is given. Besides, the types of cellular images that would be investigated in this work are being discussed. Moreover, the discussion of different image analysis tasks for microscopy such as image classification, segmentation, and object tracking are presented. Subsequently, the research questions are proposed and the main contributions are highlighted.

Chapter 2: "Neural Networks for Increased Accuracy of Allergenic Pollen Monitoring" introduces the background and motivation for classifying the morphologically similar unacetolyzed pollen of two common genera and a species in the Urticaceae family that have highly differing allergenic properties. A new pollen image dataset is collected and three deep learning neural networks are utilized and compared to distinguish each category of the pollen images. The model with the best performance is taken into account as a case study for the unseen data from aerobiological samples. **Chapter 2** answer the research question **RQ 1** and give the contributions **C. 1**, **C. 2** based on a journal paper:

- M, Polling., **C, Li.**, L, Cao., F, J. Verbeek., L, de Weger., J, Belmonte., C, De Linares., J, Willemse., H, de Boer., B, Gravendeel., Neural Networks for Increased Accuracy of Allergenic Pollen Monitoring. *Scientific Reports*. 2021.

Chapter 3: "Analysis of Automatic Image Classification Methods for Urticaceae Pollen Classification" aims to extend the content of Chapter 2. We investigate three more deep learning-based classification models to compare the performance of differently structured CNNs. The traditional machine learning classification models are adopted and compared subsequently. We intend to evaluate the performance of different approaches and analyze the possible reasons leading to the differences. Finally, we find the most accurate and effective classifier for our pollen images among all models reviewed and probed. Aiming to explore deeper insights into how the generalization and robustness of the different approaches on small-size datasets, we conduct extra experiments with two small-scale datasets and evaluate the performance. **Chapter 3** answers the corresponding research question **RQ 2** and **RQ 3**. The contribution **C. 3** is published in a journal paper:

- **C, Li.**, M, Polling., L, Cao., B, Gravendeel., F, J. Verbeek., Analysis of Automatic Image Classification Methods for Urticaceae Pollen Classification. *Neurocomputing*. Vol. 522, 2023. pp. 181-193.

Both **Chapter 2** and **Chapter 3** focus on the pollen classification task, which is based on two publications. In each publication, evaluation criteria are required to evaluate the performance of each classification model. Therefore, in the two chapters, the same evaluation formulas are presented since they deal with the same classification tasks.

Chapter 4: "An Automated Cell Tracking Pipeline for the Analysis of Neutrophil Dynamics" focuses on finding a pipeline to accurately track the migration of neutrophils forward a tail wound in the zebrafish from 2D time-lapse sequences. We build and train the segmentation models to automatically detect the position of each cell frame by frame. Subsequently, the segmentation performances are compared and evaluated. The model with the best performance will be considered for the first part of this pipeline. In addition, we employ a deep learning-based tracking model to learn the movement pattern from the ground truth tracking trajectories, which is the second part of the pipeline. Lastly, an extended Viterbi linkage method is tailored for linking the trajectories of each cell through the whole time-lapse sequence. Experiments with the proposed pipeline indicate the effectiveness and improvement compared with the other state-of-art. **Chapter 4** give the answer of the corresponding research question **RQ 4**. The contributions **C. 4** is based on the following journal paper:

- **C, Li.**, W, W.C. Yiu., W, Hu., L, Cao., H, P. Spaink., F, J. Verbeek., An Automated Cell Tracking Pipeline for the Analysis of Neutrophil Dynamics. **Prepared for submission**.
- W, Hu., L, van Steijn., **C, Li.**, F, J. Verbeek., L, Cao., R, M.H. Merks., H, P. Spaink., A Novel Function of TLR2 and MyD88 in the Regulation of Leukocyte Cell Migration Behavior During Wounding in Zebrafish Larvae. *Frontiers in Cell and Developmental Biology*. Vol. 9, 2021.

Chapter 5: "A Feature Weighted Tracking Method for 3D Neutrophils in Time-lapse Microscopy" explores the possibility of tracking the cells' migration in 3D time-lapse sequences. We present a rule-based feature-weighted method that extracts cell features manually, instead of an automatic deep learning-based method. This is to overcome the laborious labeling of training data for a deep learning model, which is rather time-consuming. Thus, several tailored features for the migration of neutrophils are designed and extracted, so

as to link the cell trajectories further based on feature similarities. Nevertheless, tracking neutrophils in 3D time-lapse sequences using a deep learning method is still challenging and limited by data annotation. **Chapter 5** answers the final research question **RQ 5**. The contribution **C. 5** is based on the following conference paper:

- **C, Li.**, W, W.C. Yiu., W, Hu., L, Cao., F, J. Verbeek., A Feature Weighted Tracking Method for 3D Neutrophils in Time-lapse Microscopy. *2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. 2022, pp. 2196-2202.

Both **Chapter 4** and **Chapter 5** deal with the cell tracking task, either in 2D or 3D. Similarly, in each publication, evaluation criteria are required. Thus, the same evaluation criteria are described which leads to part of content duplication.

Chapter 6: "Conclusion" summarizes the contribution of the presented methods for each application and elaborates the importance of our work. The limitations and possible solutions are illustrated in future work.

Chapter 2

Neural Networks for Increased Accuracy of Allergenic Pollen Monitoring

This chapter is based on the following publication:
M, Polling., **C, Li.**, L, Cao., F, J. Verbeek., L, de Weger., J, Belmonte., C, De Linares., J, Willemse., H, de Boer., B, Gravendeel., Neural Networks for Increased Accuracy of Allergenic Pollen Monitoring. *Scientific Reports*. 2021. (M, Polling. and **C, Li.** contributed equally)

Abstract:

Monitoring of airborne pollen concentrations provides an important source of information for the globally increasing number of hay fever patients. Airborne pollen is traditionally counted under the microscope, but with the latest developments in image recognition methods, automating this process has become feasible. A challenge that persists, however, is that many pollen grains cannot be distinguished beyond the genus or family level using a microscope. Here, we assess the use of Convolutional Neural Networks (CNNs) to increase taxonomic accuracy for airborne pollen. As a case study we use the nettle family (Urticaceae), which contains two main genera (*Urtica* and *Parietaria*) common in European landscapes which pollen cannot be separated by trained specialists. While pollen from *Urtica* species has very low allergenic relevance, pollen from several species of *Parietaria* is severely allergenic. We collect pollen from both fresh as well as from herbarium specimens and use these without the often used acetolysis step to train the CNN model. The models show that unacetolyzed Urticaceae pollen grains can be distinguished with > 98% accuracy. We then apply our model on before unseen Urticaceae pollen collected from aerobiological samples and show that the genera can be confidently distinguished, despite the more challenging input images that are often overlain by debris. Our method can also be applied to other pollen families in the future and will thus help to make allergenic pollen monitoring more specific.

2.1 Introduction

Pollen allergies are on the rise globally, with worldwide approximately 10–30% of adults and 40% of children affected [36][37]. For patients the symptoms include a runny nose, sneezing and itchy eyes, mouth or skin. Control measures and medication are readily available, but to alleviate the symptoms most efficiently, exposure to allergens should be kept to a minimum [38]. Therefore, for more and more people, fast and accurate monitoring of airborne pollen provides an essential early warning system [39][40]. Pollen concentrations in the air are monitored using samplers that collect airborne pollen on sticky tape, e.g. Hirst type samplers [41]. These tapes are microscopically inspected for their pollen content, a process that requires highly trained specialists. Moreover, although the allergenic pollen from some plants can be monitored at the species level (e.g. species of plantain, *Plantago* L. [42]), many other pollen grains cannot be accurately identified to this level. In many taxa, only a genus- or family-level identification is possible using current microscopic methods [43]. This is problematic since different species and even genera within the same family can possess very different allergenic profiles. An extra challenging factor in airborne pollen identification from Hirst samples is that they are collected directly from the air. In contrast to pollen grains that have been acetolyzed [44], these pollen grains still contain all organic material, and defining features are less apparent [45].

This identification challenge is exemplified in the case of the nettle family (Urticaceae). Pollen grains produced by all species from the genus *Urtica* L. (stinging nettles) have a low allergenic profile [46], while pollen from several species of *Parietaria* L. (pellitory) is a major cause of hay fever and asthma, in particular P. *judaica* L. and P. *officinalis* L. [47][48]. These pellitory species are native to the Mediterranean, but throughout the second half of the twentieth century, a range expansion occurred through north-eastern Europe, the Americas and Australia as a result of anthropogenic distribution and climate change [49][50]. *Parietaria* sensitization is highly different per geographic area, but has been reported to reach 80% in southern Italy while a value of 13% was found in the United Kingdom [51]. Species of *Parietaria* flower throughout the year but their main flowering peaks occur in May–June and August-October, which overlaps with the flowering season of *Urtica* species (June–October) [52]. Cross-reactivity is present between species of *Parietaria*, but is absent between the genera Urtica and Parietaria [46][53][54]. *Parietaria* pollen is microscopically indistinguishable from that of *Urtica* and their contribution to the total airborne pollen load is currently not assessed in either native or expanded range [55].

Pollen grains from *Urtica* and *Parietaria* species have a simple morphology: they are small ($\sim$ 11–20µm), rounded to slightly ellipsoidal tri-, tetra- or zonoporate with a psilate to scabrate surface ornament and small pores. Most species have an annulus around the pore,

i.e. a thickening of the otherwise very thin exine and a germination area called the oncus (lens-shaped body located in the apertural region) [42]. The only species of Urticaceae that can be distinguished in aerobiological samples is *Urtica membranacea* due to its small size ($\sim$ 10–12µm) and a high number of pores (usually more than six) [56]. The main difference between the pollen of *Urtica* and *Parietaria* are the slightly smaller size and coarser surface ornamentation of *Parietaria*, and a more angular outline and more pronounced annulus of *Urtica* [57].

Despite recent advances in innovative technologies, palynology is still largely an image-based discipline [58]. Therefore, automating this process currently receives a lot of attention. Automatic classification using manually selected pollen-specific features has typically resulted in relatively low classification success (see e.g. [59][60]). However, recent studies applying advances using deep learning have been very promising [61][62][63][64]. Neural networks have been used successfully to manage both the tasks of differentiating pollen from non-pollen debris as well as correctly identifying different taxa (for an overview please refer to [58]). Automatic image recognition can, however, also be used to improve identification of pollen taxa that are difficult to distinguish using traditional methods. Subtle variations in morphology that are not readily apparent through microscopic investigation may be consistently detected by neural networks. This has for example been shown for the highly similar pollen of black spruce (*Picea mariana* (Mill.) Britton, Sterns & Poggenb.) and white spruce (*Picea glauca* (Moench) Voss) using machine learning [65] and for pollen of ten species of the thistle genus *Onopordum* L. using an artificial neural network [66]. Recent advances have also been made in the field of aerobiological samples with for example the distinction of anomalous from normal pollen grains of common hazel (*Corylus avellana* L.) [67]. However, neural networks have so far not been tested for improvement of taxonomic resolution in unacetolyzed pollen in aerobiological samples.

Here we use Convolutional Neural Networks (CNNs) to distinguish morphologically similar, unacetolyzed pollen from the nettle family. We collect pollen from all species of Urticaceae present in the Netherlands (*Urtica dioica*, *U. membranacea*, *U. urens*, *Parietaria judaica* and *P. officinalis*). The pollen was collected from several sources for each species, freshly collected as well as from herbaria, and used to create a pollen image reference dataset. We compare the results of CNNs trained from scratch with those from pre-trained CNNs using transfer learning. Because of the limited size of the pollen image dataset, pre-training the CNN on a publicly available image database can help to recognize the distinguishing features of pollen grains such as pores, texture and shape.

We test both the deep CNN VGG16 and the faster CNNs MobileNetV1 and V2, and optimize the performance using data augmentation. The model is then applied to unknown

Urticaceae pollen from three aerobiological samples with high Urticaceae pollen counts. We use one sample from the Leiden University Medical Centre (LUMC), Leiden, the Netherlands as well as one sample each from Lleida and Vielha, Catalonia, Spain (ICTAUAB). In the Netherlands, stinging nettles (*Urtica*) are highly abundant and therefore it is expected that most Urticaceae pollen will be from this genus. *Urtica* is also expected to be dominant in Vielha, while in the direct surroundings of Lleida, *Parietaria* is very abundant.

The main objectives of this study are (1) to see whether a CNN model can distinguish morphologically similar unacetolyzed pollen of two common genera and a species in the Urticaceae family that have highly differing allergenic profiles; (2) to test whether the trained model can be successfully applied on aerobiological samples containing more complex and for the model before unseen input images.

2.2 Materials and Methods

A flowchart has been constructed to visualize all the steps in the Urticaceae pollen image classification process (Fig. 2.1). Details on the individual steps are described in this section.

2.2.1 Collection of Pollen

Pollen grains were collected from all five species of Urticaceae found in the Netherlands. In the genus *Urtica*, the native species *U. dioica* L. (common nettle) and *U. urens* L. (small nettle) are ubiquitous in nitrogen rich moist areas, ditches, woodlands, disturbed sites and roadsides. The exotic Mediterranean species *U. membranacea* is rarely encountered, though is included in this study since its range is expected to increase due to the effects of global warming. The genus *Parietaria* is represented in the Netherlands by the species *P. judaica* L. (pellitory of the wall) and *P. officinalis* L. (upright pellitory) that both occupy rocky substrates, mainly in the urban environment [50]. Moreover, *P. judaica* has shown a big increase in abundance over the past decades, e.g. in the Netherlands (Appendix A: Supplementary Fig. S1), but also in many other parts of the world.

Pollen from all Urticaceae species was either freshly obtained or collected from herbarium specimens (Naturalis Biodiversity Center). Fresh material was collected with the help of an experienced botanist (Barbara Gravendeel) in the direct surroundings of Leiden and The Hague during the nettle flowering seasons of 2018 and 2019. All newly collected plant specimens have been vouchered and were deposited in the herbarium of the Naturalis Biodiversity Center (L.3993376–L.3993387) (Appendix A: Supplementary Table S1). Original taxonomic assignments for the herbarium specimens were verified using identification keys

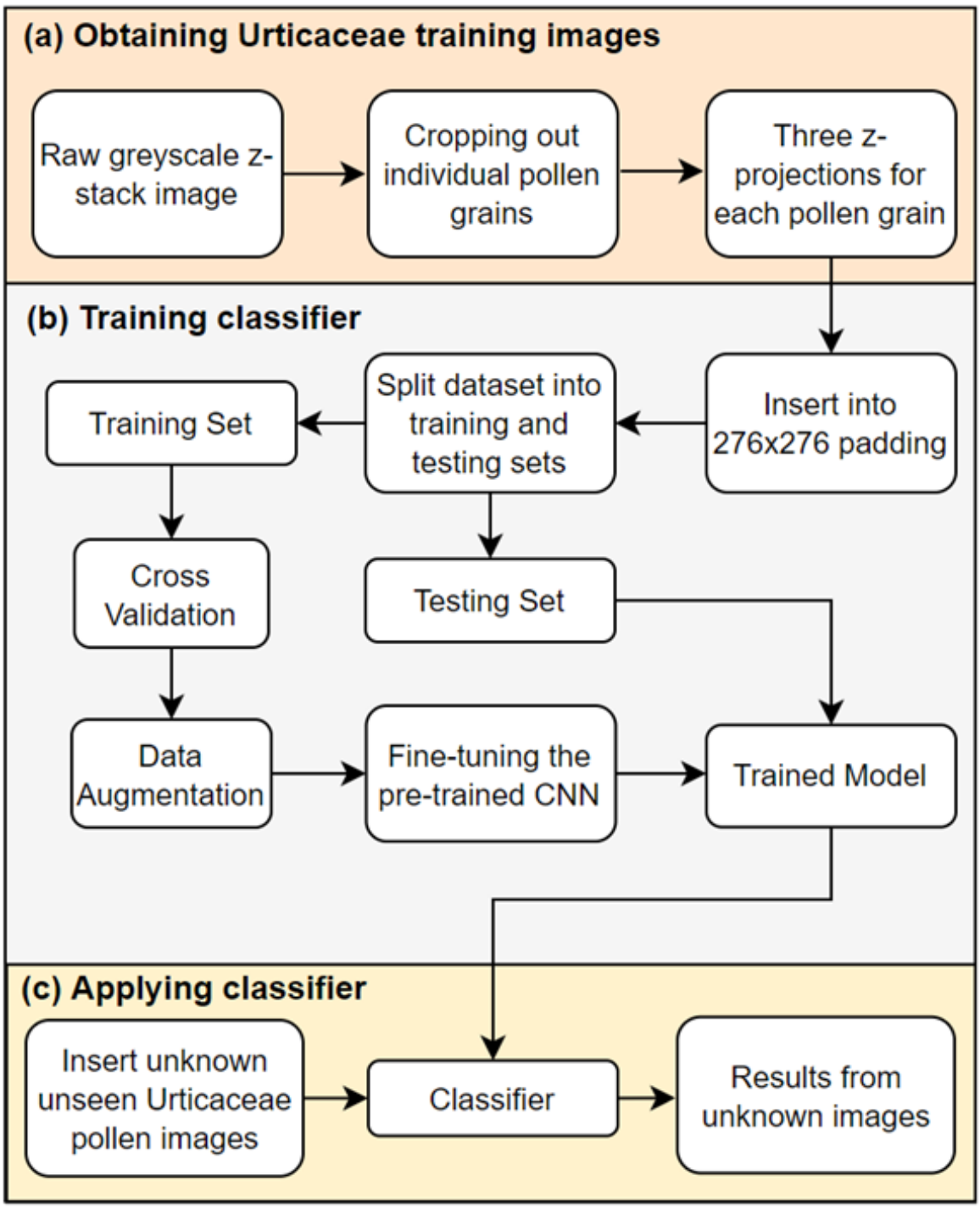


Fig. 2.1 Flowchart showing the pollen image classification process. (a) Reference pollen image capture using the custom Fiji macro Pollen_Projector. (b) Images were inserted into a fixed frame and split into training and testing sets. The training set was used for cross-validation and data augmentation (flip, brightness) so as to train the CNNs VGG16, MobileNetv1 and MobileNetv2. Results from the models trained from scratch are compared to results from transfer learning on pre-trained models. (c) Images from before unseen unknown Urticaceae pollen grains are fed to the resulting classifier. Created using https://app.diagrams.net/.

and descriptions [68]. A minimum of four different plants were sampled per species, from different geographical locations to cover as much of the phenotypic plasticity in the pollen grains as possible and reflect the diversity found on aerobiological slides.

To produce palynological reference slides, thecae of open flowers were carefully opened on a microscopic slide using tweezers. A stereo microscope was mounted in a fume hood to avoid inhalation of the severely allergenic pollen of *Parietaria* species. Non-pollen material was manually removed to obtain a clean slide. The pollen grains were mounted using a glycerin:water:gelatin (7:6:1) solution with 2% phenol and stained with Safranin (0.002%w/v). These represent the same conditions as used in airborne pollen analysis on pollen collected with a Hirst type sampler. Cover slips were secured with paraffin.

2.2.2 Pollen Image Capture

A total of 6472 individual pollen grains were scanned from the five different species of Urticaceae. The number of images for each species varied between 1055 and 1670 (Appendix A: Supplementary Table S1). The images were divided into three classes, namely *Urtica* (*U. dioica* + *U. urens*), *Parietaria* (*P. judaica* + *P. officinalis*) and *U. membranacea*. The system used for imaging was a Zeiss Observer Z1 (inverted microscope) linked to a Hamamatsu EM-CCD Digital Camera (C9100), located at the Institute of Biology Leiden (IBL). Grayscale images were used, since the pollen was stained to increase contrast and not for species recognition.

The imaging procedure was as follows: on each microscope reference slide containing only pollen of one species of Urticaceae, an area rich in pollen was identified by eye and this area was automatically scanned using multidimensional acquisition with the Zeiss software Zen BLUE. For areas that were very rich in pollen, a user-defined mosaic was created consisting of all the tiles to be scanned (e.g. 20 × 20 tiles), while a list of XY positions was used for microscopic slides less rich in pollen. Because pollen grains are 3-D shapes, catching all important features can only be achieved using different focal levels, so-called 'Z-stacks'. A total of 20 Z-stacks were used in this study with a step size of 1.8 µm. The settings used for scanning were a Plan Apochromat 100× (oil) objective and numerical aperture 0.55 with a brightfield contrast manager. To maintain similar conditions in the image collection process, the condenser was always set to 3.3 V with an exposure time of 28 ms.

2.2.3 Reference Pollen Image Library

All images were post-processed in ImageJ v1.52a (Fiji) [69] using the script Pollen_Projector (https://github.com/pollingmarcel/Pollen_Projector). The input for this script is a folder

containing all raw pollen images (including all Z-stacks), and the output is a set of projections for each individual pollen grain that are subsequently used as input for the deep learning model.

Pollen_Projector identifies all complete, non-overlapping pollen grains and extracts them as stacks from the raw Z-stack. This is achieved using binarization on the raw images to detect only those rounded objects with a circularity > 0.3 and a size larger than 5 μm. Out-of-focus images within each group of 20 Z-stack slices were removed using a threshold for minimum and maximum pixel values. The conventional input of a convolutional neural network is a three-channel image. In colour images RGB channels are commonly used, but since we use grayscale images, three different Z-stack projections were chosen to represent the three different channels. The projections used are Standard Deviation, Minimum Intensity and Extended Focus. Standard Deviation creates an image containing the standard deviation of the pixel intensities through the stack, where positions with large differences appear brighter in the final projection. Minimum intensity takes the minimum pixel value through the stack and uses that for the projection. Finally, the Extended Focus projection was created using the 'Extended_Depth_of_Field' ImageJ macro of Richard Wheeler (www.richardwheeler.net) [70]. This macro takes a stack of images with a range of focal depths and builds a 2D image from it using only in focus regions of the images. A schematic overview of the processes behind the Pollen_Projector script is shown in Appendix A: Supplementary Fig. S2. Finally, to keep the original size information of the pollen grains they were inserted into a 276×276 frame.

2.2.4 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are widely used in the field of computer vision for image classification, object detection, facial recognition, autonomous driving, etc. For this study we used the VGG16 network [71], MobileNetV1 [72] and MobileNetV2 [73] in Keras [74]. Compared with traditional neural networks and shallow convolutional neural networks, VGG16 has deeper layers that extract more representative features from images. In contrast, MobileNets are small low-power models that offer a time-efficient alternative. A feature extractor and classifier are two key structural parts of the CNN that perform the classification task. The VGG16 network contains 13 convolutional layers that form five blocks, which generate features from images in the feature extraction phase. Subsequently, three fully connected (FC) layers were built and added to the convolutional layers to classify the different classes (Appendix A: Supplementary Fig. S3). The MobileNetV1 uses depth-wise separable convolutions to build light weight deep neural networks. It has 28 layers in total. A final average pooling reduces the spatial resolution to 1 and connected with FC and

Softmax layer for classification [72]. MobileNetV2, which has 53 layers, is an improved version of MobileNetV1 by introducing inverted residual structure and linear bottleneck layers50. MobileNetV2 is more accurate than MobileNetV1 and can be much faster. We trained classification models based on aforementioned CNNs using our pollen image dataset.

During the training process, the initial parameters of convolutional layers were derived from the pre-trained network on the ImageNet dataset. Subsequently, the convolutional layers and the following fully connected layers were further fine-turned based on our own image dataset so as to classify the different classes. The pre-trained models were compared to models trained from scratch. In order to avoid overfitting, we compared the results of five- and tenfold cross-validation in the training process. For fivefold cross-validation the pollen image dataset is split into a training and validation data set in the ratio 80/20 while this is 90/10 for tenfold cross-validation. For each fold, the number of epochs was set to 30. The accuracy of the model converged at this point and the model is therefore found not to be overfitting (Appendix A: Supplementary Fig. S4).

In order to quantify model accuracy, several commonly used performance measures were used:

$$precision = \frac{TP}{TP + FP} \tag{2.1}$$

$$recall = \frac{TP}{TP + FN} \tag{2.2}$$

$$F1\ score = \frac{2 \times precision \times recall}{precision + recall} \tag{2.3}$$

$$CCR = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.4}$$

where *TP* refers to True Positives, *TN* to True Negatives, *FP* to False Positives and *FN* to False Negatives. Recall is the number of True Positives divided by the total number of elements that belong to the correct class, which is the sum of the True Positives and False Negatives. The F1-score is the weighted average of the precision and recall. The correct classification rate (CCR) reflects the accuracy of the model. The values represent the average weighted by the number of images in each class.

2.2.5 Data Augmentation

A large number of images for each class is required to train a deep learning model, as the performance will increase when more variation is fed to the model. Due to the nature of

the images investigated in this study, the model was sensitive to small changes, since the differences between the pollen grains are very subtle. Therefore, data augmentation was used to increase the variety of pollen images used as input. We selected the augmentation options brightness and flip. These options were used since size and shape of pollen are key features for their identification, and using other augmentation options would artificially change the original morphology of the pollen grains. Brightness range was set from 0.1 to 2, with < 1 corresponding to a darker image and > 1 to a brighter image. Horizontal- and vertical flip were also applied randomly (Appendix A: Supplementary Fig. S5). In addition, we applied L2 regularization and dropout in our neural network structures to prevent overfitting.

2.2.6 Test Cases

For each aerobiological sample an area representing 10% of the total deposition area was scanned manually for Urticaceae pollen grains (i.e. eight full transects at 100× magnification) resulting in 112 pollen grains from the sample from Leiden (LUMC, the Netherlands), 63 from Lleida and 26 from Vielha (both ICTA-UAB, Catalonia, Spain). One aspect of the Catalonian aerobiological samples was the presence of pollen from families that produce pollen similar to Urticaceae, that are rarely encountered in the Netherlands. These included *Humulus lupulus* L. (Cannabaceae) and *Morus sp.* (Moraceae) which were not included in our training dataset. These can be distinguished from Urticaceae, however, in the case of *H. lupulus* by their much larger size (up to 35 µm) and the very large onci and, in the case of *Morus* by the more ellipsoidal shape. These pollen grains were removed from the dataset before they were fed to the CNN for classification.

2.3 Results

2.3.1 Model Performance

In this study three different CNNs were tested on unacetolyzed pollen of Urticaceae which cannot currently be separated by specialists. The highest accuracy of the models using the three classes *Urtica*, *Parietaria* and the species *Urtica membranacea* was obtained using fivefold cross-validation (i.e. 80% training, 20% validation) with either VGG16 (98.61%) or MobileNetV2 (98.76%) (Table 2.1). Since VGG16 and MobileNetV2 had very similar performance, we trained these two models two more times to see which model performed more consistently. The mean accuracy after three repetitions was 98.50% for VGG16 with 0.145% standard deviation and 98.45% for MobileNetV2 with relatively higher standard

deviation (0.289%). The models trained from scratch showed significant lower accuracy for MobileNetV1 and V2 (both < 89%) while this value was 96.29% for VGG16.

Table 2.1 Performance comparisons of VGG16, MobileNetV1 and MobileNetV2, comparing models trained from scratch with pre-trained models as well as fivefold versus tenfold cross-validation. Values in bold represent the highest accuracy scores obtained for each of the three models.

CNN	Methods	Cross-validation	Accuracy (%)	Precision	Recall	F1 score
VGG16	From scratch	Fivefold	96.29	0.9632	0.9629	0.9629
		Tenfold	96.14	0.9616	0.9614	0.9614
	Pre-trained	Fivefold	**98.61**	0.9861	0.9861	0.9861
		Tenfold	**98.30**	0.9831	0.9830	0.9830
MobileNetV1	From scratch	Fivefold	84.54	0.8454	0.8454	0.8454
		Tenfold	86.40	0.8640	0.8640	0.8641
	Pre-trained	Fivefold	**98.15**	0.9815	0.9815	0.9816
		Tenfold	**98.15**	0.9815	0.9815	0.9815
MobileNetV2	From scratch	Fivefold	87.64	0.8769	0.8764	0.8763
		Tenfold	88.56	0.8857	0.8856	0.8856
	Pre-trained	Fivefold	**98.76**	0.9877	0.9876	0.9876
		Tenfold	**98.45**	0.9849	0.9845	0.9846

As the CNNs showed equally high accuracies with the pre-trained method (> 98%), we applied the more consistent VGG16 model using fivefold cross-validation and show the results here. The model accurately identified pollen to the genus level for 97.8% of the test images for *Urtica* and 99.0% for *Parietaria* (Fig. 2.2). For *Parietaria* three images were misclassified, while five were misclassified for *Urtica* (all to *Parietaria*). The species *Urtica membranacea* was confidently distinguished from all other Urticaceae species (99.2%), but distinction at the species-level was not possible for any of the other *Urtica* and *Parietaria* species. This is because the distinguishing features of pollen from these species (e.g. exine ornamentation) could not be resolved in the used image projections.

For all species, pollen grains were collected from a minimum of four different plants. Looking at the raw pollen images from the different plants, we identified intra-specific differences that result from natural variability within each species. To test whether the CNNs learned the pollen-specific distinguishing features rather than sample-specific details, we produced feature maps for the VGG16 model (Fig. 2.3). Despite the highly variable input images of unacetolyzed pollen from different plants, the model consistently learned features such as edges in the first convolutional layers, while finer features such as pores and annuli were learned in deeper layers.

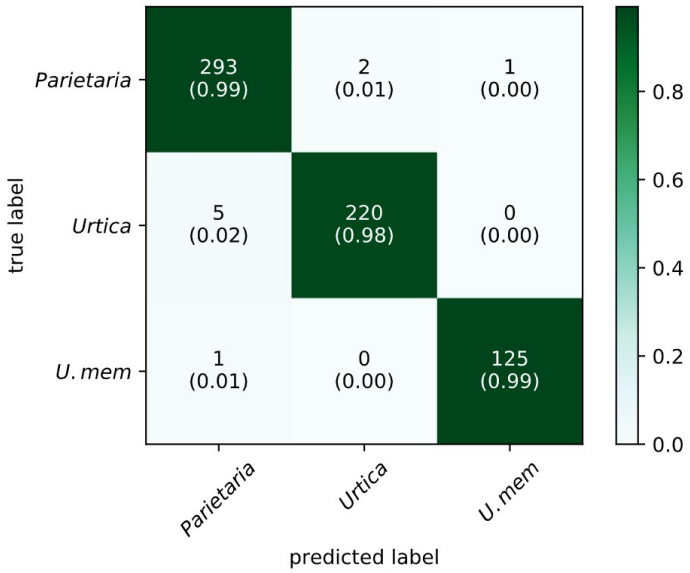


Fig. 2.2 Confusion matrix of results of pre-trained VGG16 using 80% of the images for training and 20% for testing. Numbers represent the actual number of correctly recognized images while those between brackets represent the ratio of correctly classified images. *U.mem* = *Urtica membranacea*.

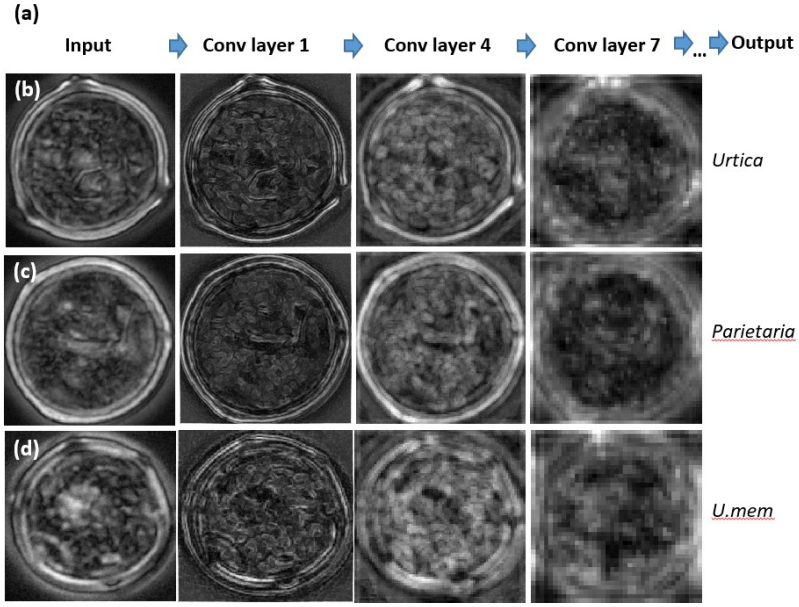


Fig. 2.3 Feature maps. (a) simplified view of the VGG16 model showing three convolutional layers. (b–d) Feature maps of Urticaceae pollen grains from the standard deviation projection created using ImageJ, that were confidently distinguished by the CNNs. (b) *Urtica urens*, (c) *Parietaria judaica* and (d) *Urtica membranacea*. Activation levels are indicated with white indicating high activation and black very low/no activation.

2.3.2 Application to Test Cases

Table 2.2 shows the results of the CNN on unknown and before unseen Urticaceae pollen from an aerobiological sample from Leiden, the Netherlands, as well as from Lleida and Vielha, Catalonia, Spain. We set the identification threshold at a value of 60% as derived from the model test images, and therefore the CNN also returned unknown images (see Appendix A: Supplementary Table S2 for the full results). For the sample from Leiden, 85.7% of the Urticaceae pollen was identified as *Urtica*, with only a minor presence of *Parietaria* (4.5%). The sample from Lleida shows dominance of Parietaria pollen grains (81.0%) while 14.3% of the Urticaceae pollen grains were classified as *Urtica*. Finally, for Vielha we find a mixture of $\sim$ 70% *Urtica* and $\sim$ 20% *Parietaria*. No *Urtica membranacea* pollen grains were identified in any of the samples. On average, unknown images account for 8.7% of the total images when using 60% identity threshold. When using a stricter identity threshold (e.g. 70%, see Table 2.2), the unknown image category increases to an average value of 13.5%.

Table 2.2 Results of the deep learning model VGG16 on Urticaceae pollen from an area representing 10% of the total deposition area of Hirst-type aerobiological samples from Leiden (the Netherlands), Lleida and Vielha (both Catalonia, Spain). Values in bold represent the highest accuracy scores obtained for each of the three classes. The threshold for identification was tested at 60% and 70%. Images that were classified below this level were classified as unknown. *U.mem = Urtica membranacea*.

Sample location	Date collected	No. pollen	*% Urtica*	*% Parietaria*	*% U. mem*	*% Unknown*	Identity threshold
Leiden, NL	23/08/2019	112	**85.7**	4.5	0	9.8	60%
Lleida, SP	16/06/2019	63	14.3	**81.0**	0	4.8	60%
Vielha, SP	09/08/2019	26	**69.2**	19.2	0	11.5	60%
Leiden, NL	23/08/2019	112	**83.0**	3.6	0	13.4	70%
Lleida, SP	16/06/2019	63	12.7	**79.4**	0	7.9	70%
Vielha, SP	09/08/2019	26	**69.2**	11.5	0	19.2	70%

2.4 Discussion

This study demonstrates incorporating neural networks to increase the taxonomic resolution of pollen grain identifications in aerobiological samples. The feature maps in Fig. 2.3 show that the trained deep learning model VGG16 looks at the traditionally used morphological features to distinguish *Urtica* from *Parietaria* pollen grains.

The characteristic thickening of the exine around the pores of *Urtica* shows the highest activation in the deeper convolutional layers. The distinct thickening is missing in *Parietaria* pollen, and the model instead focuses on the pollen outline. As expected, the only species to be distinguished by our model is *Urtica membranacea* which shows a slightly angular outline due to the larger numbers of pores (Fig. 2.3d). For the other species used in this

study, no distinction was possible even though it has been shown that pollen from species of *Urtica* (*U. dioica* and *U. urens*) (Fig. 2.3b) and *Parietaria* (*P. judaica* and *P. officinalis*) (Fig. 2.3c) can be separated based on differences in their exine ornamentation [57]. These differences can, however, only be imaged using specialized microscopy methods such as SEM or phase-contrast imaging, and are very hard to visualize using brightfield microscopy. Furthermore, these features are obscured when pollen grains are not acetolyzed. For our purposes, this species level distinction is not relevant as no known differences in allergenicity are known between either the species of *Urtica* [46] or *Parietaria* [53].

Similar to a recent study comparing pollen image classification methods, we found that using a pre-trained CNN consistently outperforms the models trained from scratch [75]. This transfer learning approach is also used by many other recent studies on deep learning of pollen images, mainly because of the limited amount of training images [61][62][63][64][76][77]. Still, we find that the VGG16 model trained from scratch achieves a high accuracy of 96.29%. This is because compared to the MobileNets, VGG16 architecture has more and deeper parameters. The MobileNets have less training parameters making them much lighter and faster, and the high accuracies found here indicate that they can be used as a light-weight alternative. In our models the amount of False Positives (*FP*) is nearly equal to the amount of False Negatives (*FN*) which is why recall, precision and F1-score were very similar.

This is the first time deep learning has been used to increase the taxonomic accuracy of unacetolyzed pollen identifications. The models represent a significant improvement of earlier attempts in distinguishing Urticaceae pollen using automatic image classification. In a previous study using hand-designed shape and texture features, pollen from three Urticaceae species could be distinguished from another with an 89% accuracy [78], though only a small image dataset was used to train the model (i.e. 100 images per species). Similar results were obtained by [59] where shape features were used with a minimum distance classifier to obtain a 86% accuracy between three species of Urticaceae. Because not all species of Urticaceae were included and a low amount of training images was used, these studies have limited applicability to the highly diverse pollen encountered in aerobiological slides. Furthermore, for both studies the trained model was tested on real case examples and only *Urtica membranacea* was successfully identified ($>$ 98%). The other two classes (*Urtica*) and (*Parietaria*) showed very high error rates (up to 44.4%) [59]. This could be because the model was not trained with sufficient variability. Because we trained the models with pollen from various sources and used data augmentation, they had a better generalizing capability.

Deep learning models have shown similar accuracy rates to ours on larger and more varied pollen datasets as well, but these either focussed on the family level [79][80][81] or on insect-collected pollen for honey analysis [61][62][63]. Increasing the taxonomic resolution

of pollen grains has been achieved by incorporating an extensively trained deep learning model with super-resolution microscopy on a case study of fossil pollen [77]. Similarly, incorporating SEM images has been found to allow for highly accurate distinction of pollen types [64]. These microscopy methods, however, are often much more expensive than using light microscopy and require extensive sample preparation. Moreover, nearly all of these studies work with acetolyzed pollen that allow easier recognition of distinguishing features, and used pollen collected from a single location.

To validate our model, we tested it on Urticaceae pollen from aerobiological samples collected from different locations in Spain and the Netherlands. Most of the pollen grains from the sample from Leiden, the Netherlands were identified by the deep learning model as *Urtica*, with only a low number of images identified as *Parietaria*. While *Parietaria* plants are relatively abundant around the sampling location in Leiden and were flowering on the chosen date, its pollen is most likely simply outnumbered by the much larger number of nettles in the area. For Lleida (Catalonia), where pellitory plants are abundantly present, *Parietaria* pollen grains dominated the assemblage, while the sample from Vielha showed a mixed assemblage. The number of unknown images was the highest for the sample from Vielha (11.5%), which is most likely the result of the presence of more debris on the pollen grains making a certain identification impossible. In all aerobiological slides, debris on top of or below the pollen grains was observed in different focal plains. Nevertheless, the model still successfully classified most of the pollen grains, and in most cases with high confidence (Appendix A: Supplementary Table S2). This shows the potential broad application of this method and opens up opportunities to study both seasonal as well as long-term yearly dynamics of *Parietaria* versus *Urtica* abundance of airborne pollen, as well as using this method to distinguish other morphologically similar species of allergenic importance from different families (e.g. Betulaceae, Amaranthaceae, Oleaceae). To further improve the generalization of this classification system, future work will focus on increasing the amount of training images from variable sources. Furthermore, more elaborate techniques like regularization will be considered to improve the variability in the image dataset [82]. Since for allergenic pollen monitoring reducing the amount of false negatives (i.e. increasing recall) is particularly important, more models will be tested to identify the best recall values.

A limitation of our method is that currently pollen from aerobiological slides have to be located manually. It has already been shown that automating this process is feasible, e.g. using a deep learning approach [83]. In other systems like the commercially available Classifynder system, pollen are automatically located and imaged using darkfield imaging after which a simple neural network classifies the pollen [84]. This is also the case for the BAA500 system used by, e.g. Oteros et al. [85], that was particularly developed for recognizing and

classifying unacetolyzed airborne pollen for hay fever predictions. Lastly, using a CNN and digital holography on pollen grains directly from the air (i.e. unacetolyzed) showed great promise in quantifying pollen automatically to the family level [86]. While these systems achieve automated and accelerated pollen counting, our method instead particularly increases the accuracy of information useful for allergy prevention by making it more specific.

2.5 Conclusions

In conclusion, using a combination of an image-processing workflow and a sufficiently trained deep learning model, we were able to differentiate unacetolyzed pollen grains from two genera and one species in the nettle family. These are genera that are indistinguishable with current microscopic methods but possess different allergenic profiles, and thus the ability to differentiate them is of medical significance. Our method can be more broadly applied to distinguish pollen from similarly challenging allergenic plant families and can help in producing more accurate pollen spectra to improve the forecasts for allergy sufferers.

Chapter 3

Analysis of Automatic Image Classification Methods for Urticaceae Pollen Classification

This chapter is based on the following publication:
C, Li., M, Polling., L, Cao., B, Gravendeel., F, J. Verbeek., Analysis of Automatic Image Classification Methods for Urticaceae Pollen Classification. *Neurocomputing*. Vol. 522, 2023. pp. 181-193.

Abstract:

Pollen classification is considered an important task in palynology. In the Netherlands, two genera of the Urticaceae family, named *Parietaria* and *Urtica*, have high morphological similarities but induce allergy at a very different level. Therefore, distinction between these two genera is very important. Within this group, the pollen of *Urtica membranacea* is the only species that can be recognized easily under the microscope. For the research presented in this study, we built a dataset from 6472 pollen images and our aim was to find the best possible classifier on this dataset by analysing different classification methods, both machine learning and deep learning-based methods. For machine learning-based methods, we measured both texture and moment features based on images from the pollen grains. Varied feature selection techniques, classifiers as well as a hierarchical strategy were implemented for pollen classification. For deep learning-based methods, we compared the performance of six popular Convolutional Neural Networks: AlexNet, VGG16, VGG19, MobileNet V1, MobileNet V2 and ResNet50. Results show that compared with flat classification models, a hierarchical strategy yielded the highest accuracy with 94.5% among machine learning-based methods. Among deep learning-based methods, ResNet50 achieved an accuracy of 99.4%, slightly outperforming the other neural networks investigated. In addition, we investigated the influence on performance by changing the size of image datasets to 1000 and 500 images, respectively. Results demonstrated that on smaller datasets, ResNet50 still achieved the best classification performance. An ablation study was implemented to help understanding why the deep learning-based methods outperformed the other models investigated. Using Urticaceae pollen as an example, our research provides a strategy of selecting a classification model for pollen datasets with highly similar pollen grains to support palynologists and could potentially be applied to other image classification tasks.

3.1 Introduction

The analysis of pollen grains is widely used in detection and monitoring of airborne allergenic particles. In recent years, pollen seasons are prolonged due to global warming and climate change [87]. This subsequently causes an increase of hay fever patients who are affected by rising allergenic pollen levels in the air [88]. In palynological research, identification of pollen grains plays a key role to suggest safety treatments to patients with allergic rhinitis. It helps patients and medical professionals to monitor the levels of airborne allergenic pollen and thus plan outdoor activities and medication treatments accordingly. Pollen recognition analysis is often implemented by human visual inspection under the microscope, and includes the identification of differences in shape, texture, size and other specific features of pollen categories [89]. However, merely relying on human inspection for pollen classification tasks is unrealistic as the size of image datasets is rapidly increasing due to high-throughput screening, while the expertise needed to perform this detailed analysis is rapidly disappearing. Another limitation of manual classification is that it may induce classification biases with varied inspectors when the differences among pollen categories are very subtle. Thus, automatic classification techniques are now being developed that have proven to perform well in pollen classification tasks [63][89][90][91][92].

Researchers have adopted different approaches to automate the process of pollen classification. In general, the two main technical approaches of pollen image classification tasks are machine learning-based methods [89][93] and deep learning-based methods [62][63][94][95][96].

Machine learning methods need to be fed with manually selected features before they can extract these from images. The, so called, handcrafted features used in machine learning techniques are mostly based on shape, texture and other related properties of pollen grain images. The extracted features play an important role in the performance of classification. In addition, suitable feature selection methods and classifiers are also crucial for machine learning-based classification methods.

In the work of del Pozo-Baños et al. [97], a combination of geometrical and texture characteristics was proposed as the discriminative features for a 17 class pollen dataset. Incorporation of Linear Discriminant Analysis (LDA) and Least Square Support Vector Machines (LS-SVM) accomplished the best performance of 94.92% accuracy. Marcos et al. [98] extracted four texture features including Gray-Level Cooccurrence Matrices (GLCM), log-Gabor filters (LGF), Local Binary Patterns (LBP) and Discrete Tchebychev Moments (DTM) from a pollen image dataset with 15 classes. Fisher's Discriminant Analysis (FDA) and K-Nearest Neighbour (KNN) were subsequently applied to perform dimensionality reduction and multivariate classification. It yielded an accuracy of 95%. Manikis et al. [93] used texture features obtained by GLCM and seven geometrical features computed from the

binary mask of a pollen image dataset. A Random Forest (RF) classifier was used in the classification stage; with this classifier 88.24% accuracy was achieved on 6 pollen classes. Machine learning thus show highly varying results, and is seemingly dependent on the dataset used.

Instead of manual design of the features, deep leaning methods automatically extract image features through convolutional layers of the network. In recent years, many state-of-the-art Convolutional Neural Networks (CNNs) were applied in pollen classification tasks. In the work of Sevillano et al. [63], pretrained AlexNet was used to classify a dataset with 46 different classes of pollen grains. By incorporating data augmentation and cross-validation techniques, an accuracy of 98% was achieved. In the work presented by Battiato et al. [90], both AlexNet and SmallerVGGNet were implemented to classify five classes of pollen grains, with 13,000 images. The two networks obtained a performance of 89.63% and 89.73% accuracy, respectively. A seven layer deep Convolutional Neural Network designed by Daood et al. [94], was trained on a dataset of 30 pollen classes and accomplished a 94% correct classification rate. Astolfi et al. [99] analysed a pollen dataset composed of 73 pollen categories. They compared the performance of eight state-of-the-art CNNs which included Inception-V3, VGG16, VGG19, ResNet-50, NASNet, Xception, DenseNet-201 and Inception-ResNet-V2. They showed that DenseNet-201 and ResNet-50 achieved superior performance against other CNNs with an accuracy of 95.7% and 94.0%, respectively.

Based on the analysis of related work mentioned above, both machine learning and deep learning-based methods have achieved comparable performance on pollen datasets. However, the pollen datasets used in these studies is derived from species or genera from different plant families [100]. The morphology of each class of pollen is already clearly distinctive under the microscopy by human analysts. For example, the public POLEN23E dataset [89] consists of 23 pollen classes from the Brazilian Savannah, derived from 23 genera in 15 families. Each class of pollen has a, different shape, size and texture. The other public pollen dataset from the Brazilian Savannah, called POLLEN73S, which was analysed by Astolfi et al. [99], has 73 pollen classes with clearly variable colour, shape and other morphological differences. These distinct features ensured the high performance of the classification model applied. However, in this research, we are more interested in distinguishing genera of the same family Urticaceae, namely, *Parietaria* and *Urtica* which are morphological very similar, but cause completely different allergy levels. Pollen of the two genera cannot currently be distinguished easily by a palynologist; the species *Urtica membranacea* represents the only species that can be specifically distinguished.

Parietaria and *Urtica* are two genera commonly encountered in the Netherlands. The occurrence of *Parietaria* plants is very much increasing and could induce severe allergy in hay

fever patients while *Urtica* does not [96]. Species from the genus *Parietaria* as well as *Urtica membranacea* originate from the Mediterranean area and now increase in North Europe. Due to climate change these species can maintain themselves in northern countries such as the Netherlands. The pollen grains from these taxa exhibit a similar roundness, and are all very small, but differ in the following features: 1) different number of pores: *Parietaria* and *Urtica* have 3 to 4 pores, while this is variable for *Urtica membranacea* (usually 5 to 10; i.e. pantoporate). 2) The average size of *Parietaria* pollen is slightly smaller ($\sim$11-18μm) and it with a coarser and more irregular surface than *Urtica*. *Urtica* pollen are bigger in size on average ($\sim$15.2-21.1μm), and often have a more pronounced thickened exine around the pore (annulus). The shape of *Urtica membranacea* is slightly angular and is easily distinguished because of its small size ($\sim$10-12μm) and high number of pores. Although these pollen grains have the aforementioned differences, it is not possible for experts to distinguish the three different classes by the naked-eye using a light microscope. This is mainly because of their small size. Therefore, in order to improve the accuracy and efficiency of Urticaceae pollen classification, automatic algorithms are required.

Currently, very few studies focused on pollen classification of the Urticaceae family. Rodrìguez-Damián et al. [100] extracted both geometrical and texture features and probed three classifiers: Support Vector Machines (SVM), Multi-Layer Perceptron (MLP) and Minimum Distance Classifier (MDC). The best performance of 88% success rate was reached on a total of 291 pollen images of the three species *Parietaria judaica*, *Urtica urens* and *Urtica membranacea*. Compared with their relatively small Urticaceae dataset, we aimed to analyse a much larger dataset that includes all species (*Parietaria judaica*, *Parietaria officinalis*, *Urtica dioica*, *Urtica urens* and *Urtica membranacea*) present in the Netherlands. We grouped these five species into 3 classes: *Parietaria* (*Parietaria judaica*, *Parietaria officinalis*), *Urtica* (*Urtica urens*, *Urtica dioica*) and *Urtica membranacea*. Both *Parietaria* and *Urtica* dominate in the Netherlands but cause a totally different allergy level. *Urtica membranacea* is an exotic Mediterranean species and it is the only species can be easily distinguished. Hence our starting point for three labels and thus, our study is based on a three-class classification task. The best performance achieved in our study is 99.4% by a ResNet50. Actually, it is also possible to do a classification task over all five species (see Appendix B: Supplementary Table S1). Another challenge is that the pollen grains that we used were unacetolyzed. Acetolyzed pollen grains are those that all pollen materials are destroyed by acetolysis with the exception of sporopollenin that forms the outer pollen wall, the exine. In contrast to acetolyzed pollen grains, unacetolyzed pollen keep their original organic features which are less apparent. To the best of our knowledge, our previous work [96] was the first and the only time that CNNs were applied and compared for the

analysis of the unacetolyzed Urticaceae pollen grains. In this study, we extended this work further and aimed to find an automatic classification model with the best performance in both machine learning-based and deep learning-based methods for our unacetolyzed Urticaceae dataset. In general, for a deep learning model, a large dataset is required as input. However, there are many limitations for researchers to collect a sufficiently large dataset in practice. Subsequently, we were curious about how machine learning-based and deep learning-based methods work on a smaller sized image dataset. Therefore, two additional experiments on smaller datasets were designed to compare the performance of different classification models. For a 1000-image dataset, a ResNet50 yielded the best performance of 96.3% while for a 500-image dataset, it achieved the best accuracy of 93.3%.

3.2 Methods

3.2.1 Sample and Image Preparation

3.2.1.1 Sample Preparation of the Pollen Grains

Our pollen data included both fresh pollen specimens and dry pollen specimens [96]. Fresh pollen specimens were collected by an experienced biologist in the surroundings of Leiden and The Hague (the Netherlands) during the flowering season of 2018 and 2019. Dry pollen specimens were collected from the herbarium of Naturalis Biodiversity Center, Leiden, the Netherlands, using identification keys and descriptions. For each species in our dataset, pollen samples from 4 to 8 plants were taken, from different geographical locations in order to cover as much variation as possible (see Appendix B: Supplementary Table S2). Microscope slides were freshly prepared by aerobiological experts from Naturalis Biodiversity Center. The thecae of open flowers were carefully opened on a microscopic slide using tweezers. Non-pollen materials were manually removed. The pollen grains were mounted using a glycerin:water:gelatin (7:6:1) solution with 2% phenol and stained with Safranin (0.002% w/v). Cover slips were sealed with paraffin. Each slide contained only one plant of each species of Urticaceae.

3.2.1.2 Image Capturing and Pre-processing

The slide area rich in pollen was scanned automatically using a Zeiss Observer Z1 microscope with a Plan Apochromat 100× objective (NA 1.4), equipped with a Hamamatsu c9100 EM-CCD camera. As pollen grains are three dimensional, it is difficult to set a focal plane for pollen samples. Therefore, we captured 20 slices of images along the Z axis for pollen grains.

The step size was 1.8 µm. After obtaining a stack of images including pollen, the grains were detected and cropped; this is referred to as the 3D pollen stack. Fig. 3.1 (a) shows an example of a slice from the raw image. Fig. 3.1 (b) shows all 20 slices of different focal depths of an individual pollen grain. In total, 6472 individual pollen stack images were captured. Three categories were included for the image classification study. These were (1) *Parietaria* (including *Parietaria judaica*, *Parietaria officinalis*), (2) *Urtica* (*Urtica dioica*, *Urtica urens*), (3) *Urtica membranacea* (see Fig. 3.2).

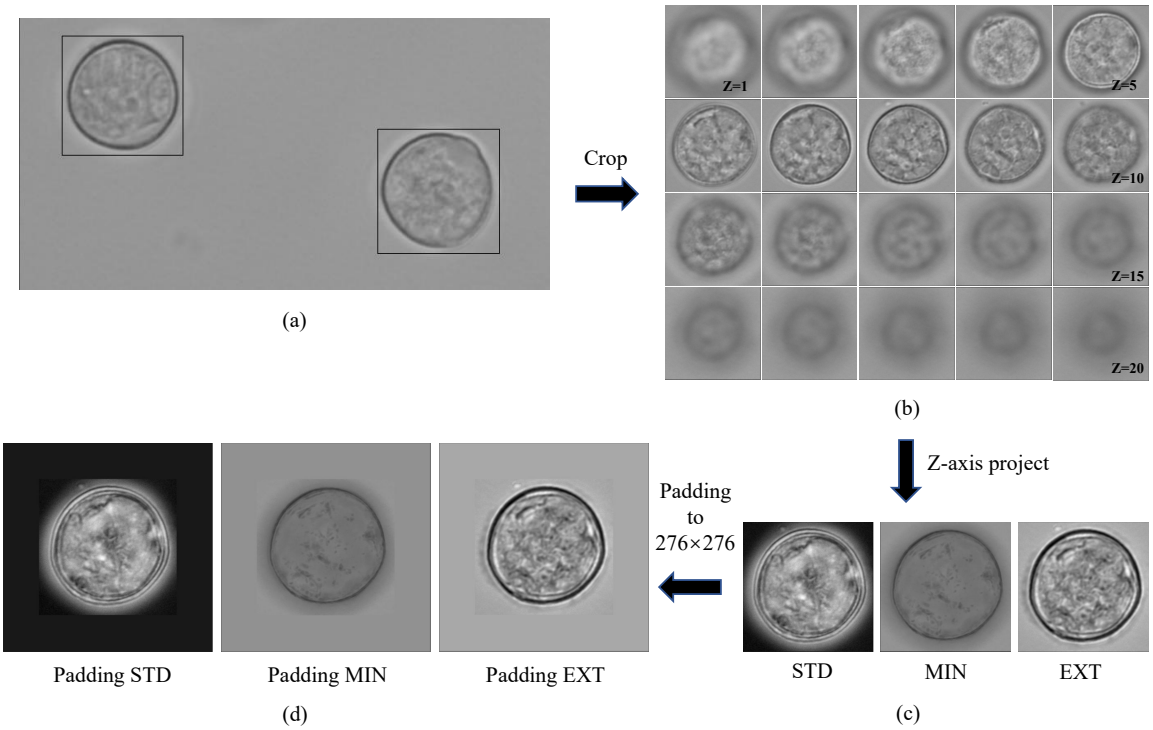


Fig. 3.1 The workflow of pollen image acquisition. (a) One plane of raw pollen image. (b) 20 slices at different focal depths of gray scale images of one individual pollen grain. (c) 3 different projections in Z axis, STD = Standard Deviation Projection, MIN = Minimum Intensity Projection, EXT = Extend Focus Projection. (d) The padding images of each projection.

As shown in Fig. 3.1 (b), not all of the 20 slices in the Z-stack were in-focus. In order to obtain as much informative features as possible, all Z-stack images were further processed using a Z-stack projection method [69]. Z-stack projection is a method of analysing and highlighting specific features from all slices in a stacked image without incorporating out-of-focus blurriness. The selected projections were Standard Deviation (STD), Minimum Intensity (MIN), and Extend Focus (EXT) [101], which are shown in Fig. 3.1 (c) and Fig. 3.2. The three projections per pollen grain were treated as three separate channel images

for the input of supervised classification models. Same-sized images are required to feed into classification models which is achieved by resizing images to the same size. However, for pollen images captured by a microscope, the morphology and details of pollen grains are expected to be changed by resizing. We did not opt for resizing as the resized images might ignore the original size differences of pollen grains, which is a potentially important diagnostic feature. There are several ways to preserve this nature of the features. One can crop images of the pollen grains to the same size from a slide-scan image. However, some pollen grains are very closely located to each other so that cropping to the same size might cause incomplete pollen separation. Therefore we chose for as another approach which is to use padding of the cropped images so that the resulting images all have the same size. For the padding size, the biggest size of all individual pollen images was selected, i.e.276 $\times$ 276 pixels. The padding value was set to the median value at the edge of each pollen image in order to make the content of the padding images more natural.

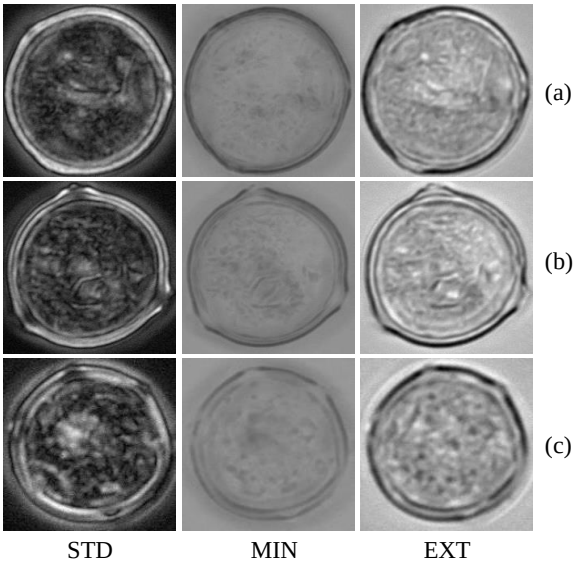


Fig. 3.2 A sample pollen grain of each category in our dataset. (a) *Parietaria judaica*. (b) *Urtica urens*. (c) *Urtica membranacea*. Each column represents the STD, MIN, EXT projections of each pollen grain, respectively.

After the pre-processing of the images, we aimed to find the best classification model for our Urticaceae pollen dataset. Machine learning and deep learning-based classification models were constructed and the performance of each model was evaluated and compared.

3.2.2 Machine Learning Methods

3.2.2.1 Feature Extraction and Selection

Machine learning methods require manual selection of relevant features before extracting these from images. One challenge is how to select an appropriate set of features for classification. By observing the characteristics of Urticaceae pollen grains, we noticed that *Parietaria* has a coarser ornamentation on the surface of its pollen grains, *Urtica* has thickened pores and *Urtica membranacea* has an angular outline. Texture attributes of surface and shape features was considered as the appropriate pollen descriptors for Urticaceae pollen grains. We aimed to include as much representative features as possible for Urticaceae pollen classification due to their high morphological similarities. The following selected features have been proven to be successful in classification tasks for pollen recognition: GLCM, LBP, Gabor filter texture features and Histogram of Oriented Gradients (HOG). These features have provided satisfactory results as reported in [90][98]. Both First Order Statistics (FOS), which are derived from statistical properties of the intensity histogram of an image, and Wavelet measurements, which is a texture analysis based on a Discrete Wavelet Transform (DWT) have been included as they have been successfully used in pattern recognition of cells [102][103]. In addition, the seven Hu invariant moments and three shape measures derived from the invariants, referred to as Extension, Dispersion and Elongation (EDE) were included as invariant descriptors for shape [104]. So, based on aforementioned image-based studies [105][106] we have selected six texture features and two moment-based features to represent the characteristics of pollen grains in our study. Table 3.1 shows the selected features with the dimensions of each feature vector.

Table 3.1 The dimension of feature vector of each feature.

Feature	Dimension
HOG	3600×3
LBP	416×3
Gabor filter	60×3
GLCM	24×3
FOS	5×3
Wavelet	9×3
Hu moments	7×3
EDE	3×3
Total	4124×3

HOG features in combination with a SVM classifier have proven to be a representative texture descriptor in the image recognition field [107]. In the procedure of HOG feature extraction, we divided an image into several small connected regions, aka cells. Each cell returns a 9×1 feature vector. In order to be more invariant in representing the changes of shadowing and illumination, a larger region, referred to as the block, is formed. The block consists of four cells and returns a 36×1 feature vector. In the experiment, a pollen image with size (276×276) can be divided into 100 blocks, consequently, a 3600×1 feature vector is returned at the end.

LBP is an invariant descriptor that can be used for texture classification. A n-digit binary number is obtained by comparing each pixel with its n neighbour pixels on a circle with radius r and used to compute the histogram. In our study, we fine-tuned the parameters and set it as $n = 24$, $r = 3$. Similar to the HOG feature extraction procedure, the image was also divided into 16 smaller blocks. In this manner a 416×1 LBP feature vector is returned.

GLCM characterizes the texture of images by considering the spatial relationship of pairs of pixels in an image. GLCM is created based on a statistical rule $P(i, j, d, theta)$, which refers to the number of times that gray-level j occurs at a distance d and at a direction $theta$ from gray-level i. Our experiment set $d = 1$ and the direction $theta = 0°, 45°, 90°, 135°$. We further calculated the properties based on the matrix which are defined by Haralick et al. [108], as the extracted feature vector. Finally, a 24×1 feature vector was returned.

A 5×1 texture feature vector of FOS named standard deviation of intensity, smoothness, skewness, uniformity and entropy was calculated [103]. These measures reflect the statistical properties of the intensity histogram of each pollen image. Wavelet-based texture measurements show the image details in different directions after DWT. We calculated the mean, standard deviation and entropy of intensity in three directions (horizontal, vertical, diagonal) of each pollen image. The dimension of wavelet measurement is 9×1.

Another commonly used texture descriptor is the Gabor filter: it reflects frequency content in a specific direction of a localized region of the image. In this study, 12 Gabor filters are designed at four directions 0°, 45°, 90°, 135° with three different frequencies $\pi/4, \pi/2, 3\pi/4$. Therefore, the dimension of the Gabor filter feature vector is 60×1.

Hu moments are normally extracted from pollen images with the property of scale and rotation invariance. A total of seven moment invariants as proposed by Hu [109] were extracted. EDE features are derived from the 1^{st} and the 2^{nd} order invariants. Even though the morphological differences of pollen between genera is subtle, it was expected that these image moment features could play a role in the pollen classification task.

Each pollen image in our dataset consisted of 3 projections (STD, MIN, and EXT) obtained by projecting 20-slice Z-stack images. Fig. 3.2 shows that the features of each pro-

jection are different, especially the texture features. In order to include as much information of the pollen grain dataset as possible, we calculated 8 features for 3 projections of pollen images and concatenated these together as the final feature vector (cf. Table 3.1). Therefore, after feature extraction, the dimension of feature vector reaches 4124×3. Compared with public pollen image datasets like POLEN23E, POLLEN73S [89][99] and the 2D Urticaceae pollen images used in [100], our dataset based on a method of projection of 3D images might intrinsically extract more representative features. This partially underlies the reason of the high performance results that we have achieved.

In order to remove redundant and irrelevant features, feature selection and dimensionality reduction techniques were applied. Feature selection returns prominent subsets of features while dimensionality reduction creates new features with lower dimension from the original features. Feature selection includes a filter method, wrapped method and embedded method [110]. These methods have been shown to improve the accuracy of classification studies [111][112].

In this study, feature selection methods including Mutual information, SelectFromModel and Principal Component Analysis (PCA) were assessed. Mutual information is a filter feature selection method. In this method a subset of the best K features which are most relevant to the target labels is chosen; the selection of the number K is mostly based on experience. The embedded method named SelectFromModel works more flexible. It selects the most relevant features according to the performance of machine learning models during the training process. This integrated approach ensures that the selected features are the best for the model. Alternatively, PCA is widely used in feature dimensionality reduction. It is a process of computing the principal components and preserving the first few of them that maximize the variances between different classes. PCA is known for obtaining lower-dimensional features and improving the accuracy of machine learning model in many fields such as image recognition [97][113].

In short, these selected feature selection and dimensionality reduction methods were applied after feature extraction. Subsequently, classification models were used as the last step to classify pollen grains.

3.2.2.2 Classifiers

Once features are extracted from images, an efficient classification model is required so that it can perform well on the pollen classification task. A large number of classification approaches exist. In this study, SVM, RF, MLP and Adaboost classifiers were used as they have shown to perform well in previous pollen classification studies [90][93][100].

Combined with extracted features and feature selection methods, the classifiers have been trained and the hyperparameters were tuned based on the performance of the experiment.

3.2.2.3 Hierarchical Strategy

A flat classification model is a straight-forward approach for taxonomic classification tasks. Only one classifier is used to classify all classes. However, the process ignores potential hierarchical structure among different classes which could reduce performance [114]. Hierarchical classification can be seen as a particular tree-structured approach. It merges the classes which are more similar into subgroups and classifies these subgroups separately. Varied classifiers are used to classify classes at different hierarchical levels until reaching the leaf nodes. Hierarchical strategy has been used in many classification tasks [103][115] and has proven to increase the performance compared with flat classification models.

For our work, we structured the three classes of pollen grains as a hierarchical tree as shown in Fig. 3.3. We used a local classifier per parent node approach to train a two-stage classifier for each parent node in the hierarchical tree. In the first stage we merged *Parietaria* and *Urtica* into one subgroup based on high morphological similarity. *Urtica membranacea* is a distinct species that can already be clearly distinguished under the light microscope and it was therefore treated as the other subgroup. In the second stage, *Parietaria* and *Urtica* were subsequently classified. At both stages we selected the best classifier and feature selection method for each parent node in order to get a better performance of the hierarchical classification model.

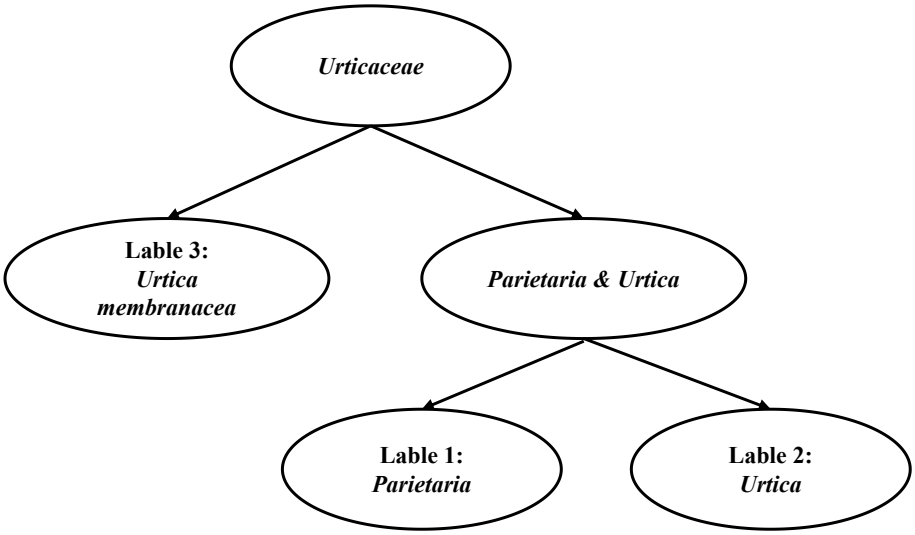


Fig. 3.3 Hierarchical tree of pollen classification.

3.2.3 Deep Learning Methods

3.2.3.1 Convolutional Neural Networks

We selected several well-established deep learning models for pollen classification. AlexNet achieved the best performance on an ImageNet classification task in 2012 [116]. In order to prove that convolutional network depth affects image classification accuracy, Simonyan et al. [117] proposed VGGNet for large-scale image recognition. A 16-layer VGG16 network and 19-layer VGG19 network have proven to be the two best-performing convolutional neural networks in other studies. ResNet introduces a residual learning framework to ease the training process of very deep networks—up to 152 layers [118]. Even though ResNet has lower complexity than VGGNets, it still has millions of parameters making the network computational heavy. A more light-weighted set of neural networks, named MobileNets, was designed in order to embed it into mobile devices or other applications [72]. In this study, we selected the aforementioned models: i.e., AlexNet, VGG16, VGG19, ResNet50, and MobileNet V1 and V2 [73] to classify our pollen dataset.

In addition, we used a transfer learning technique to alleviate the computational burden of training from scratch. With our three classes pollen dataset, we fine-tuned the pretrained AlexNet based on the ImageNet dataset in the PyTorch framework. The other pretrained networks implemented in the Keras Library were fine-tuned on the TensorFlow platform. All experiments were executed on a dedicated server equipped with two NVidia GeForce GTX 2070 with 8 GB GPUs using Linux Ubuntu operating system.

3.2.3.2 Data Augmentation

Deep learning models need a large number of image datasets covering diverse scenarios. Data augmentation techniques play an important role in increasing the variety of images. Furthermore, if appropriate transforms are applied to a dataset, data augmentation can greatly improve the performance and reduce overfitting. In our case, differences between pollen of Urticaceae genera are very subtle and slight configurational changes during image capturing may affect the classification performance. Therefore, a large amount of training data was needed for our study.

In order to simulate the possible transforms of pollen data, brightness and flip transforms were most obvious and straightforward to select and therefore applied as augmentation options. Other transforms like rotation, zoom range, etc., were not selected.

3.2.3.3 Cross-validation and Hard Voting

Cross-validation is applied in an image training process to improve the effectiveness, robustness and generalization ability of deep learning models, as well as to prevent overfitting. In this study, K values of 5 or 10 were used [62][119]. A 5-fold cross-validation means the ratio of training data and validation data is 8:2 while 10-fold cross-validation means the ratio is 9:1. We compared the performance of deep learning models with 5-fold cross-validation and 10-fold cross-validation, respectively. After 5/10-fold cross-validation, 5/10 models were obtained and tested on test datasets. In this study, hard voting was adopted to calculate the final accuracy rather than the average accuracy of 5/10 models. Hard voting sums the votes for class labels from each model for predicting the class with the majority votes. The experimental results show that the hard voting technique further improves the classification performance on the test dataset.

3.2.4 Performance Evaluation

Before addressing the results, we first introduce the performance measures that we used. These were:

$$precision = \frac{TP}{TP + FP} \tag{3.1}$$

$$recall = \frac{TP}{TP + FN} \tag{3.2}$$

$$F1\ score = \frac{2 \times precision \times recall}{precision + recall} \tag{3.3}$$

Where *TP* refers to true positives, *TN* represents true negatives, *FP* is false positives and *FN* is false negatives. High precision and recall values are able to verify good performance against false positives and false negatives of a model [63]. The F1 score is an overall measurement which combines precision and recall together. A high F1 score means that a model retrieved both low false positives and low false negatives, which proves the consistency of those measures and the reliability of the model. Precision, recall and F1 score were calculated as the average weighted by the number of true instances for each class in our experiments. The accuracy of the classification model was also calculated by the number of true predictions divided by the total number of samples.

The performance measurements mentioned above are commonly applied in flat classification models. However, they are not suitable for hierarchical classification models since they do not differentiate the misclassification errors among different hierarchical stages.

Instead, we adopted the measures suggested in [115], which include hierarchical precision (*hP*), hierarchical recall (*hR*) and hierarchical f-measure (*hF*). These are defined as follows:

$$hP = \frac{\sum_i |\widehat{C}_i \cap \widehat{C}'_i|}{\sum_i |\widehat{C}'_i|} \tag{3.4}$$

$$hR = \frac{\sum_i |\widehat{C}_i \cap \widehat{C}'_i|}{\sum_i |\widehat{C}_i|} \tag{3.5}$$

$$hF = \frac{2 \times hP \times hR}{hP + hR} \tag{3.6}$$

Where $\widehat{C}_i$ is a set of real classes with all of its ancestors and $\widehat{C}'_i$ is a set of predict classes with all of its ancestors. Ancestors here refer to all the nodes which are connected to the specific real/predict class node in the hierarchical tree structure.

3.3 Experiment Results and Discussion

We derived results from two parts: a comparison of pollen classification performance with different machine learning algorithms and an analysis of performance of deep learning neural networks. Two additional experiments show how machine learning-based and deep learning-based methods work on smaller-size image datasets. Our results are based on performance measures.

3.3.1 Results with Machine Learning Methods

For machine learning methods, the 6472 pollen images were divided into training and test datasets in a ratio of 9:1. In this experiment, we compared the performance of each classification model using 5-fold and 10-fold cross-validation, respectively. In addition, a grid search technique [120] was applied in the training process to help search for optimal hyperparameters automatically. With this technique, a list of hyperparameter values is defined beforehand and the optimal set of parameters, that can maximize the accuracy of the model, is returned.

Table 3.2 shows the performance of each classifier with corresponding hyperparameter settings and feature selection methods. In Table 3.2 we present the results of a SVM with a Radial Basis Function (RBF). This SVM, with a penalty parameter C=4, was shown to be optimal for this type of data. These parameters were determined by a grid search technique. Two dictionaries which included kernel functions (linear, rbf, poly) and penalty values (from

1 to 10) were set and the best parameter combination was returned. Other parameters in SVM were selected by default values. The highest score with an accuracy of 91.5% (σ= ±0.008) and F1 score of 0.915 was achieved in combination with a PCA threshold of 0.8 with 5-fold cross-validation. The threshold 0.8 of the PCA means that the first few principal components with the ratio of accumulated data variation and total data variation greater than 0.8 are preserved while all others are discarded. In this case, the final size of the selected feature vector is 179 $\times$ 1 (see Appendix B: Supplementary Table S3). For the RF classifier, the number of trees was set to 500. The SelectFromModel function of a threshold 'mean' embedded in a classifier can achieve the best performance of 88.6% (±0.015) with 10-fold cross-validation. The threshold 'mean' was set according to the importance of each feature. It means that a feature whose importance is greater or equal to the 'mean' is kept while others are discarded. The final size of the selected feature vector is 2064 $\times$ 1. Similarly, all of these parameters are fine-tuned by the grid search technique.

Table 3.2 Performance comparison of different flat classification models. Standard deviation, of each subset via cross-validation, is given in brackets.

Classifiers	Hyperparameters	Feature selection with threshold	Cross-validation	Precision	Recall	F1 score	Accuracy
SVM	kernel='rbf', C=4	PCA (0.8)	10-fold	0.913 (±0.012)	0.913 (±0.012)	0.913 (±0.012)	0.913 (±0.012)
			5-fold	0.915 (±0.008)	0.915 (±0.008)	0.915 (±0.008)	0.915 (±0.008)
RF	Estimators=500	SelectFromModel (mean)	10-fold	0.886 (±0.015)	0.886 (±0.015)	0.886 (±0.015)	0.886 (±0.015)
			5-fold	0.884 (±0.010)	0.884 (±0.010)	0.884 (±0.010)	0.884 (±0.010)
MLP	Solver='sgd', Maxiter=300	PCA (0.85)	10-fold	0.898 (±0.011)	0.898 (±0.011)	0.898 (±0.011)	0.898 (±0.011)
			5-fold	0.890 (±0.008)	0.890 (±0.008)	0.890 (±0.008)	0.890 (±0.008)
Adaboost	Estimators=500, LR=0.5	Mutual Information (2000)	10-fold	0.789 (±0.014)	0.754 (±0.021)	0.749 (±0.025)	0.754 (±0.021)
			5-fold	0.784 (±0.015)	0.745 (±0.024)	0.743 (±0.028)	0.748 (±0.024)

Furthermore, we carried out experiments with MLP and Adaboost classifiers. The MLP led to the best accuracy of 89.8% (±0.011) with the following settings: the optimizer was Stochastic Gradient Descent (SGD); the number of maximal iterations was 300; PCA feature reduction was at a threshold of 0.85. The final size of feature vector after PCA feature reduction becomes 337 $\times$ 1. Adaboost reached a performance of 75.4% (±0.021) with the number of Estimators set to 500 and a Learning Rate (LR) of 0.5. Mutual Information plays an important role in the accuracy of the Adaboost classifier because it selected 2000

features most relevant to target classes of pollen datasets. In addition, 5-fold and 10-fold cross-validation obtained a comparable performance with different machine learning-based classification models.

In order to improve the performance of the flat classification model further, we applied a hierarchical strategy classification. We have implemented different combinations of flat models to form a two-level hierarchical structure which includes SVM + SVM, SVM + MLP, MLP + SVM, SVM + RF, RF + SVM. Table 3.3 shows all permutations with SVM for the hierarchical classification model except for Adaboost. This is because SVM achieved the best performance of the flat classification models while Adaboost achieved the lowest performance. In Table 3.3 the best combination is SVM + SVM which obtained 94.5% of accuracy and 0.941 of all of the *hP*, *hR* and *hF*. The reason why *hP*, *hR* and *hF* are equal is that our simple hierarchical tree structure only has 3 layers and for each parent node, it only has 2 children. According to the definition of *hP* and *hR* (cf. eq. (5), (6)), in this case, the calculation of *hP*, *hR* and *hF* is equal. Based on our experiments, a hierarchical model which combined SVM + PCA and SVM + PCA at both 2 levels was considered as the best model among machine learning-based methods.

Table 3.3 Performance comparison of hierarchical classification models.

Hierarchy Level 1		**Hierarchy Level 2**					
Classifier with Hyperparameters	Feature selection with threshold	Classifier with Hyperparameters	Feature selection with threshold	**hP**	**hR**	**hF**	**Accuracy**
SVM (kernel='rbf',C=6)	PCA (0.8)	SVM (kernel='rbf',C=4)	PCA (0.8)	0.941	0.941	0.941	0.945
SVM (kernel='rbf',C=4)	PCA (0.8)	MLP (Solver='sgd', Maxiter=300)	PCA (0.85)	0.920	0.920	0.920	0.916
MLP (Solver='sgd', Maxiter=300)	PCA (0.85)	SVM (kernel='rbf',C=4)	PCA (0.8)	0.929	0.929	0.929	0.933
SVM (kernel='rbf',C=4)	PCA (0.8)	RF (Estimators=500)	SelectFromModel (mean)	0.907	0.907	0.907	0.897
RF (Estimators=500)	SelectFromModel (mean)	SVM (kernel='rbf',C=4)	PCA (0.8)	0.913	0.913	0.913	0.911

3.3.2 Results with Deep Learning Methods

Our starting point has been to work with commonly available deep learning methods, the AlexNet, VGG16, VGG19, ResNet50, MobileNet V1 and MobileNet V2. First of all, a total of 6472 pollen grain images were divided into a training set and test set in a ratio of 9:1 randomly. The test set was composed of images that were not seen by the model during the training process and that were used to test the trained classification model. Secondly, considering that deep learning models require a huge amount of data, a data augmentation

technique was applied. Thirdly, similar with machine learning models, 5-fold and 10-fold cross-validation were used to prevent overfitting and increase the robustness as well as the generalization ability of the deep learning models. Data augmentation process was performed for each cross-validation set independently.

Based on these aforementioned procedures, we fine-tuned six representative deep learning classification models. The pretrained AlexNet was implemented in the PyTorch framework. The whole five convolutional layers and three fully connected layers were fine-tuned with a learning rate 0.001. We set the batch size to 128 and the number of batches to 4000. Fig. 3.4 (a) shows the performance plot of AlexNet. The plot shows the accuracy and loss for both training and validation dataset. The accuracy drastically increases in the first 700 batches and then converges gradually. AlexNet achieved an accuracy of 94.1% with a standard deviation of (±0.002) using 10-fold cross-validation while 5-fold cross-validation retrieved a comparable accuracy of 92.4% (±0.002) (see Table 3.4). The average accuracy and standard deviation were calculated by training each model three times. The accompanied precision, recall and F1 score achieved with the same 0.941. The reason why these three measurements are so similar is that, for this case, False Positive (FP) samples are nearly equal to the number of False Negatives (FN). The consistency of these measurements shows the reliability of the model. Six positive samples and six negative samples among three classes of pollen grains performed by AlexNet are shown in Fig. 3.5 (a) and (b). The actual label, predicted label and confidence score of each sample are indicated. Label 1 to 3 represent the 3 classes of pollen: *Parietaria*, *Urtica*, *Urtica membranacea*. In Fig. 3.5 (a), positive samples clearly show the distinguished properties. *Urtica* pollen has obviously thickened pores compared with pollen of *Parietaria* and *Urtica membranacea*. Pollen of *Urtica membranacea* has more angular outlines than pollen of the other two genera. In Fig. 3.5 (b) illustrates that, when the properties of the 3 classes of pollen are not clearly displayed, the network will misclassify these samples because of high similarities among the 3 classes.

Similarly, the pretrained VGG16 and VGG19 models were fine-tuned with a batch size of 64 and the number of epochs set to 30. The whole network was fine-tuned using learning rate 2e-5 without freezing any layers. Fig. 3.4 (b) and (c) show the performance plots of VGG16 and VGG19, respectively. Both plots show that the models converge well in the training process. Table 3.4 lists detailed measurements of these two models. For 10-fold cross-validation, VGG16 obtained an average accuracy of 98.3% with a standard deviation of (±0.001) while VGG19 achieved a comparable average accuracy of 98.6% with (±0.002).

The pretrained ResNet50, MobileNet V1 and MobileNet V2 models were constructed based on Keras as well. And all of these models were fine-tuned on our pollen dataset. Table 3.4 shows that light-weights MobileNets achieved a comparable accuracy with VGGNets

but it had a slightly higher standard deviation. This means that light models are not as robust as heavy-weight models. ResNet50 obtained the highest performance of 99.4% with 10-fold cross-validation among the six models investigated due to its deeper network layers and a creative residual structure. Consequently, ResNet50 was selected as the best performing model among all the models that we implemented. In addition, both 5-fold and 10-fold cross-validation achieved comparable performance for all of the deep learning models studied.

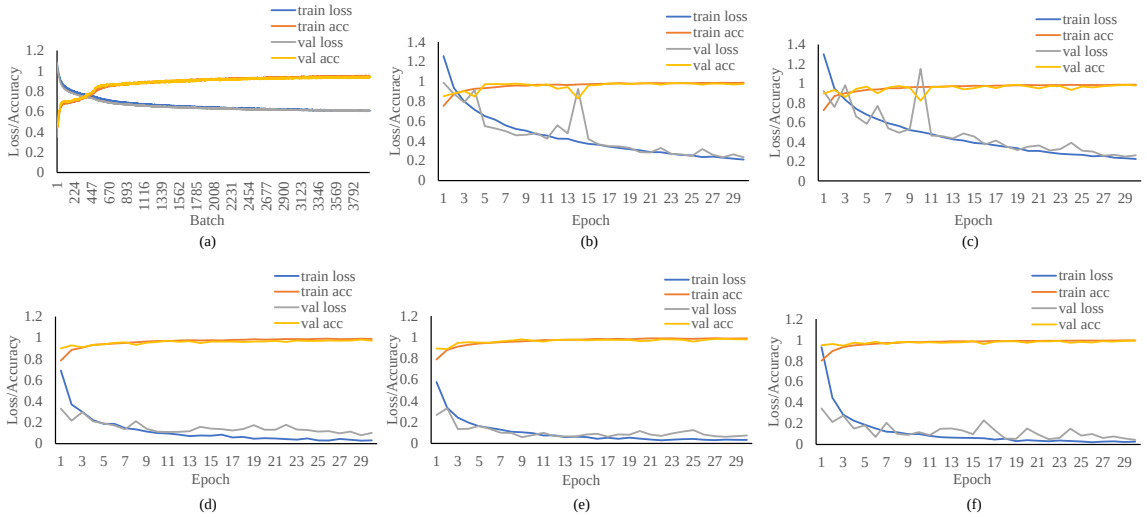


Fig. 3.4 The performance plots of (a) AlexNet, (b) VGG16, (c) VGG19, (d) MobileNet V1, (e) MobileNet V2, and (f) ResNet50, in terms of training loss, etc., with respect to the number of epochs.

Table 3.4 Classification performances of different deep learning classification models. Standard deviation, training each model three times, is given in brackets.

	Cross-validation	**Precision**	**Recall**	**F1 score**
AlexNet	10-fold	0.941(±0.002)	0.941(±0.002)	0.941(±0.002)
	5-fold	0.924(±0.002)	0.924(±0.002)	0.924(±0.002)
VGG16	10-fold	0.983(±0.001)	0.983(±0.001))	0.983(±0.001)
	5-fold	0.985(±0.002)	0.985(±0.002)	0.985(±0.002)
VGG19	10-fold	0.986(±0.002)	0.986(±0.002)	0.986(±0.002)
	5-fold	0.988(±0.003)	0.988(±0.003)	0.988(±0.003)
ResNet50	10-fold	0.994(±0.002)	0.994(±0.002)	0.994(±0.002)
	5-fold	0.993(±0.002)	0.993(±0.002)	0.993(±0.002)
MobileNet V1	10-fold	0.981(±0.003)	0.981(±0.003)	0.981(±0.003)
	5-fold	0.980(±0.002)	0.980(±0.002)	0.980(±0.002)
MobileNet V2	10-fold	0.985(±0.003)	0.985(±0.003)	0.985(±0.003)
	5-fold	0.984(±0.003)	0.984(±0.003)	0.984(±0.003)

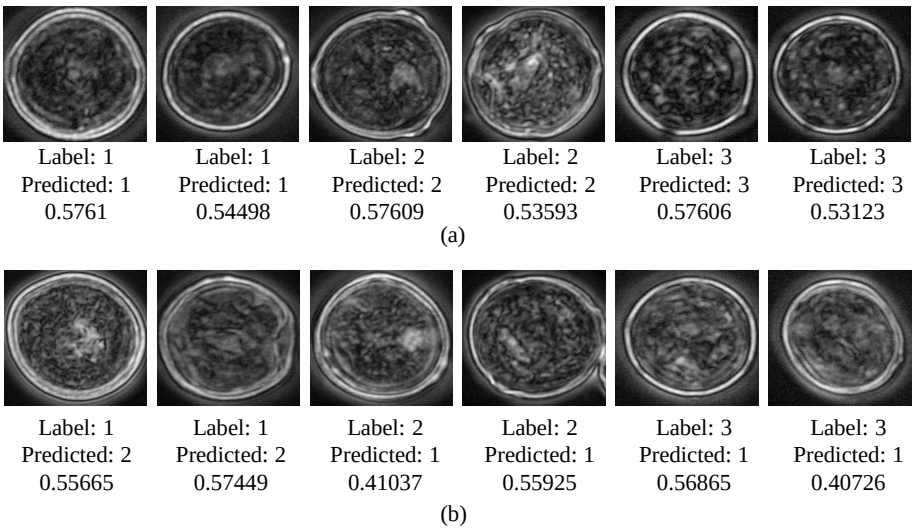


Fig. 3.5 Examples of classification performed by AlexNet. (a) Positive samples with their predicted label and confidence score. (b) Negative samples with their predicted label and confidence score.

Fig. 3.6 and 3.7 show the positive and negative samples classified by VGG16 and ResNet50, respectively. Compared with AlexNet (Fig. 3.5), the confidence score of the classified pollen of VGG16 was higher. The reason is that VGG16 has deeper layers which results in the extraction of more detailed and distinct features of pollen data. ResNet50 has a much deeper and complex network structure and the classification accuracy was higher than that of VGG16 (Table 3.4). For positive samples in Fig. 3.7 (a), the confidence score of ResNet50 was almost 1.00 which was higher than that of VGG16. And in the test dataset, only three negative samples, shown in Fig. 3.7 (b), were misclassified due to the high performance of the ResNet50 model.

After analysing automatic classification models based on both machine learning and deep learning methods on our pollen dataset, we observed that the ResNet50 neural network reached an accuracy of 99.4% (±0.002) which is 4.9% higher compared to the hierarchical machine learning model. Deep learning-based methods perform better to classify our Urticaceae pollen grains. In addition to our pollen images dataset, we have used the deep learning classifiers to other pollen image datasets available to us. These have not been used in the training/testing but are used as unseen samples to probe the classifiers from our study. The classification results with these additional datasets confirm the findings from study. Early results with these extra datasets, based on VGG16, have already been reported in [96]. With our ResNet50 model, the results with unseen data are even better. In Appendix B: supplementary Table S4 these results are summarized.

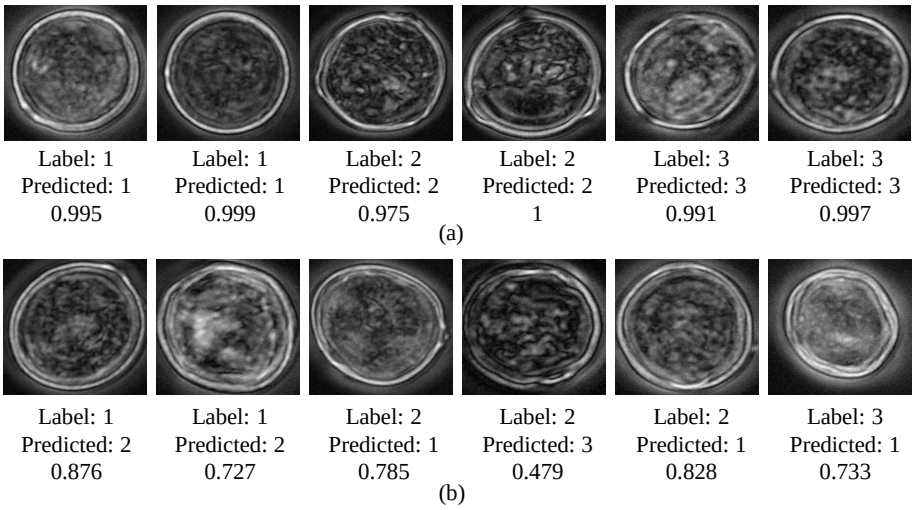


Fig. 3.6 Examples of classification performed by VGG16. (a) Positive samples with their predicted label and confidence score. (b) Negative samples with their predicted label and confidence score.

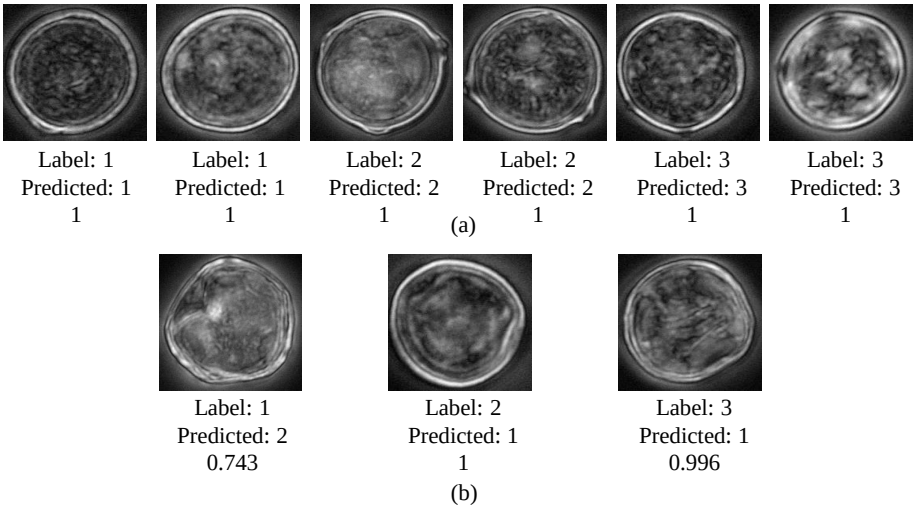


Fig. 3.7 Examples of classification performed by ResNet50. (a) Positive samples with their predicted label and confidence score. (b) Negative samples with their predicted label and confidence score.

3.3.3 Results on Smaller-size Image Datasets

It is common knowledge that the training process of deep learning model requires the use of a large data set. However, in daily practice, there are limitations in the collection of sufficient samples and images. Therefore, we examined the robustness of both machine learning-based and deep learning-based methods when facing a smaller dataset. Are machine learning-based method and deep learning-based method comparable in performance? To answer this question, starting from the original data, two smaller pollen image datasets consisting of 1000-sized and 500-sized image subsets, were constructed. These image subsets were randomly selected from 6472 images. And the ratio of the 3 classes was 1:1:1. The experimental results on smaller datasets shown in Table 3.5 was based on one round of selection.

On both smaller pollen datasets (1000 and 500 images), the same six deep learning-based models were applied. For machine learning models, we refine-tuned the hyperparameters of the best performed flat model (SVM) and hierarchical model (SVM+SVM). Table 3.5 shows the performance of both machine learning-based and deep learning-based methods on the two smaller image datasets. Compared with the 88% accuracy of the flat model on the 1000-image dataset, the 93.9% accuracy obtained by the hierarchical model demonstrated that hierarchical strategy improves the performance. The performance of the hierarchical model was, however, still lower than the deep learning models, except for AlexNet. We obtained similar results with the larger 6472-image dataset (Table 3.3 and Table 3.4). This is probably due to the fact that AlexNet has a shallow layer-structure which includes only five convolutional layers and three fully connected layers. The results indicate that extracting as many features as possible manually as well as using a hierarchical strategy outperforms a shallow deep learning neural network such as AlexNet. For the 500-image dataset we obtained similar results.

Table 3.5 Performance comparison of different methods on smaller-size image datasets. Standard deviation, training each model three times, is given in brackets.

	Deep learning-based						**Machine learning-based**	
	AlexNet	VGG16	VGG19	ResNet50	MobileNet V1	MobileNet V2	Flat model	Hierarchical model
Accuracy of 1000 images	0.916 (±0.006)	0.943 (±0.006)	0.943 (±0.012)	0.963 (±0.012)	0.947 (±0.015)	0.950 (±0.010)	0.880	0.939
Accuracy of 500 images	0.861 (±0.032)	0.920 (±0.000)	0.920 (±0.020)	0.933 (±0.012)	0.927 (±0.012)	0.907 (±0.023)	0.760	0.896

In order to obtain sufficient information for a statistical analysis of the performance of the models, we used a cross-validation approach over the entire image set with two

differently sized groups of subsets [1]. We implemented the 5-fold cross-validation to select five image subsets from the 6472-image dataset, as well as the 10-fold cross-validation to select ten smaller image subsets. With this selection method, the average performance of all subsets from different models was compared, the results of which are given in Appendix B: Supplementary Table S5. The experiments confirmed that ResNet50 achieved the best performance on both 1000- and 500-sized dataset.

Deep learning-based methods show a better performance on both the large and smaller pollen datasets. An ablation study was conducted to help understand why this difference was retrieved. Convolutional layers of deep neural networks can catch more representative features compared with extracting handcrafted features manually. We visualized intermediate feature maps of VGG16 and ResNet50 in Fig. 3.8 and Fig. 3.9 to provide extra insight in the procedure of feature extraction. For each different layer of the model, different features were extracted. In Fig. 3.8, the feature maps of convolutional layers 1, 4 and 7 of VGG16 are shown. In Fig. 3.9, we show the feature maps of the convolutional layer in stage 1 and three bottlenecks in stage 2 of ResNet50. The structure of ResNet50 can be divided into five stages [118]. Stage 1 consists of 1 convolutional layer and stage 2-5 consist of a different number of bottleneck structures. From both feature maps, we can conclude that, in the first several convolutional layers, basic pollen features such as edges and textures (surface ornamentation) are clearly displayed as was also found in [96]. With an increase of the number of network layers, more and more complex and abstract features influence the performance of pollen classification. For example, in convolutional layer 4 of VGG16 and the first bottleneck in stage 2 of ResNet50, other important parts of pollen such as the pores are highlighted. In the higher layers of the network, only the most representative features are retained but these features are difficult to grasp. With the help of a deep convolutional network that can extract different features from low level (detail) to high level (abstract), the best score in pollen classification tasks is achieved.

In addition, all of the techniques, i.e., transfer learning, data augmentation and hard voting, clearly contributed to improve the performance of the deep learning models under study. Table 3.6 shows to what extent the accuracy can be improved by different techniques applied on the around 1000-sized image subset. Five 1000-sized image subsets were selected via 5-fold cross-validation. The average performance of five subsets was calculated and the results are shown in Table 3.6. The first row shows that ResNet50 achieved 81.4% accuracy if the model was trained from scratch. Transfer learning improved the accuracy to 95.0% using pre-trained parameters which were trained on the ImageNet dataset. Based on the

[1]Two differently sized groups of subsets consist of 1000- and 500-sized image subsets, they are given as an indication; the real number is slightly higher.

95.0% accuracy of transfer learning, the ResNet50 model with data augmentation improved the accuracy to 96.2%. The accuracy is 1.2% higher than without data augmentation which is shown in the second row. Data augmentation helped to increase the variety and the size of image data. Hard voting predicted the class labels with the majority votes of different classification models. The combination of all these techniques significantly improved the accuracy of the ResNet50 model to 97.5%. The third row of Table 3.6 shows that without transfer learning, the accuracy of the ResNet50 model was only 86.1%. We can conclude that, in this study, transfer learning plays a more important role in the performance of deep learning models comparing with data augmentation and hard voting. Because of these advanced techniques, we achieved great success with our deep learning models in the classification of our Urticaceae pollen data. Appendix B: Supplementary Table S6 shows the ablation study of ResNet50 based on 10-fold cross-validation selection method.

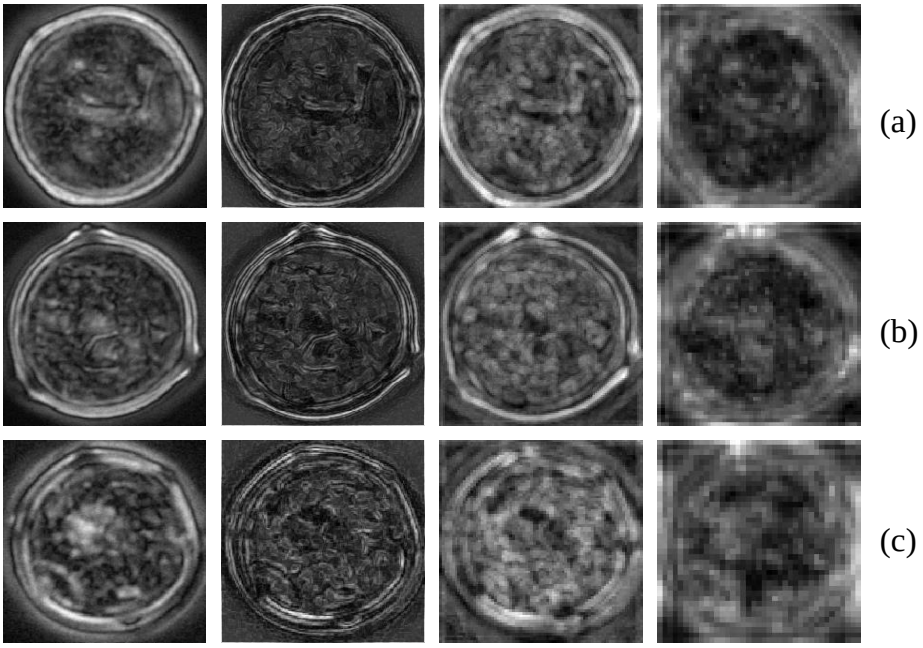


Fig. 3.8 Example of feature maps of VGG16. (a) *Parietaria*. (b) *Urtica*. (c) *Urtica membranacea*. Column 1 represents the input data, column 2-4 are the output of convolutional layer 1, 4, and 7, respectively.

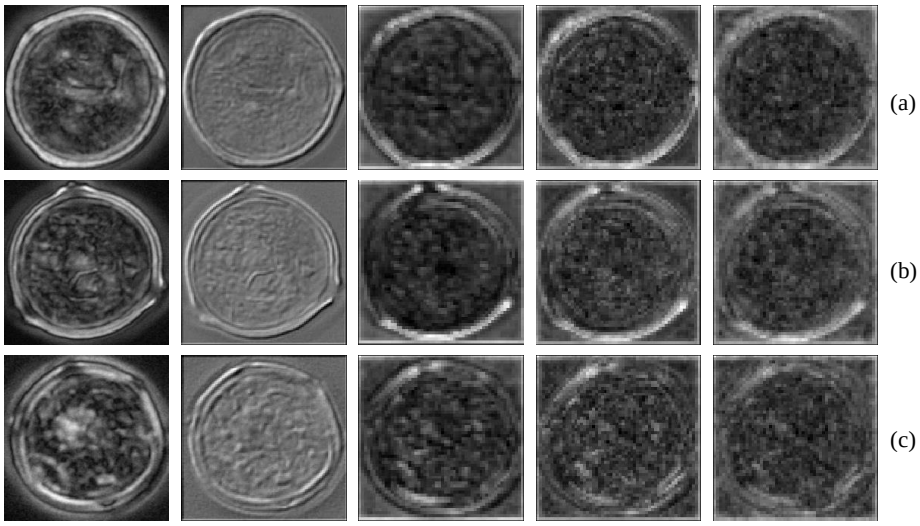


Fig. 3.9 Example of feature maps of ResNet50. (a) *Parietaria*. (b) *Urtica*. (c) *Urtica membranacea*. Column 1 represents the input data, column 2-5 are the output of convolutional layer in stage 1, and output of three bottlenecks in stage 2, of ResNet50, respectively.

Table 3.6 Ablation study with ResNet50. The average performance of ResNet50 based on five (about) 1000-sized image subsets via 5-fold cross-validation selection method is given. Standard deviation, of five subsets, is given in brackets. Numbers in italics refer to training without transfer learning and data augmentation, respectively.

	Training from scratch	**With/*without* transfer learning**	**With/*without* data augmentation**	**With hard voting**
Accuracy	0.814 (±0.025)	0.950 (±0.017)	0.962 (±0.004)	0.975 (±0.002)
	0.814 (±0.025)	0.950 (±0.017)	*0.950 (±0.017)*	0.971 (±0.022)
	0.814 (±0.025)	*0.814 (±0.025)*	0.837 (±0.026)	0.861 (±0.023)

3.4 Conclusions

This study aimed to find the automatic classification model with the best performance to classify Urticaceae pollen grains. Pollen grains of this family have high morphological similarity while they induce different allergenic levels. Few researchers focused on classification of pollen of the Urticaceae nettle family to genus and species level. For our research, a pollen

grain image dataset of the Urticaceae family was constructed, consisting of 6472 images. The pollen grains were unacetolyzed and to our knowledge, these had not yet been used before the analysis of pollen image classification tasks except for our own previous work [96]. Two approaches in image classification techniques including machine learning-based methods and deep learning-based methods were implemented and analysed. For machine learning-based methods, six texture features and two moment features were extracted. Subsequently, several popular feature selection techniques and classifiers were applied. Compared with flat classification models, a hierarchical strategy was confirmed to achieve great success with the classification task. Among the different machine learning methods, the highest performance of 94.5% accuracy was achieved by hierarchical classification models. For deep learning-based methods, six well-established deep Convolutional Neural Networks were used to perform a classification task. Together with data augmentation, cross validation and hard voting techniques, the pretrained ResNet50 model, which achieved an accuracy of 99.4% (±0.002) was considered the best classification model among the six models investigated.

From our comparison of machine learning-based with deep learning-based methods, we conclude that deep learning-based methods perform better for pollen image classification. Two additional experiments demonstrated that deep learning models are more successful for both large and smaller sized datasets. One reason is that deep learning models can extract more representative features of pollen images from low (detailed) to high (abstract) level. The performance of machine learning methods is, however, highly dependent on the quality of features that are extracted from the image dataset. In addition, transfer learning, data augmentation and hard voting techniques drastically improved the performance of deep learning models. An ablation study showed that the accuracy of deep learning models is improving step by step. Deep nets such as Inception-V3, DenseNets, and NASNets have shown to perform well on datasets in the public domain. Nevertheless, ResNet50 has already yield an accuracy of 99.4% on our dataset. We may apply these deeper networks on a larger dataset in the future. Our work clearly demonstrates what automatic classification methods can accomplish for highly similar images of pollen species in the Urticaceae family. This technique can be broader applied to similar pollen from other families. This method could also potentially be extended to cope with other image classification tasks.

CRediT Authorship Contribution Statement

Chen Li: Conceptualization, Investigation, Methodology, Validation, Writing-Original Draft **Marcel polling:** Conceptualization, Investigation, Resources, Writing-Review Editing **Lu Cao:** Conceptualization, Investigation, Writing-Review Editing, Supervision **Barbara**

Gravendeel: Resources, Writing-Review Editing **Fons J. Verbeek:** Conceptualization, Writing-Review Editing, Supervision

Declaration of Competing Interest

The authors declare that they have no competing interests.

Acknowledgments

This work was supported by the Chinese Scholarship Council through Leiden University and the European Union's Horizon 2020 research and innovation programme under H2020 MSCA-ITN-ETN Grant agreement No 765000 Plant.ID. We would like thank Xiaoqin Tang, Zhan Xiong, Shima Javanmardi and other colleagues who assist us through feedback during discussions and meetings.

Chapter 4

An Automated Cell Tracking Pipeline for the Analysis of Neutrophil Dynamics

This chapter is based on the following publications:
C, Li., W, W.C. Yiu., W, Hu., L, Cao., H, P. Spaink., F, J. Verbeek., An Automated Cell Tracking Pipeline for the Analysis of Neutrophil Dynamics. **Prepared for submission**.

W, Hu., L, van Steijn., **C, Li.**, F, J. Verbeek., L, Cao., R, M.H. Merks., H, P. Spaink., A Novel Function of TLR2 and MyD88 in the Regulation of Leukocyte Cell Migration Behavior During Wounding in Zebrafish Larvae. *Frontiers in Cell and Developmental Biology*. Vol. 9, 2021.

Abstract:

Neutrophils play a key role in the innate immune system. They act as the primary line of defense when bacteria, viruses or other harmful foreign particles invade the immune system. Accurate measurement of movements of neutrophils including velocity, direction and displacement, are crucial to study the regulation of cell migration behaviour. Cell tracking is a key technology to realize the quantification of these measurements. In this paper, we developed a pipeline, including cell segmentation, cell motion tracking between two frames and trajectory linkage, to realize tracking of the cell. Our starting point was to collect time-lapse sequences of neutrophils using a confocal microscope. We pre-processed each frame in the time-lapse sequence so as to improve the image quality by denoising, smoothing and contrast enhancement. Subsequently, a deep learning model, i.e. U-Net, was used to segment cells in each image frame. U-Net was used again to track the cells between two adjacent frames by calculating the score matrices representing the posterior probability of linkage. Moreover, an extended Viterbi algorithm was applied to find optimal trajectories based on score matrices obtained from U-Net. Results demonstrate that our pipeline outperforms the other state-of-the-art methods. It has a great potential to be applied in other cell migration studies.

4.1 Introduction

Acute inflammation could be identified by the monitoring of neutrophil migration, which can be caused by invading pathogens [121]. Neutrophils are one of the crucial immune cells. The migration pattern of neutrophils reflects the process *in vivo* of the innate defense mechanism against invading pathogens. However, the underlying mechanisms of neutrophil migration are not yet completely understood. In the field of biology and medicine [122][123], microscopy techniques enabling observing at high spatial and temporal resolution have contributed to the advancement of the analysis of the dynamics of the cellular processes. As the first line of defense against harmful pathogens, the exploration of movements of living neutrophils, *in vivo*, in the spatial-temporal domain can contribute to a better understanding of their behaviours and mechanisms. Cell tracking is the key technology to realize the quantitative analysis of neutrophil migration. It aims to identify the trajectory of each cell in a time-lapse sequence. However, due to the complexity of cell behaviour this field still lacks accurate and automated algorithms. The manual tracking of cells is a time-consuming and challenging task. Especially with the introduction of high throughput screening, it becomes impossible to conduct manual tracking for a large number of image datasets. Therefore, developing automatic and powerful cell-tracking approaches is essential.

In biomedical research, cell tracking necessarily is preceded by cell detection and based on an approach with either single-cell tracking or multi-cell tracking [124][125][126][127].

The main idea of tracking by detection is to segment cells from each frame first and then associate the cells frame by frame. Cell segmentation algorithms are designed to localize single cell object in each frame. Initially, the majority of existing approaches are based on different strategies such as thresholding, watershed, and deep learning neural networks [128][129][130][131], etc. The Otsu adaptive thresholding method [132] is one of the thresholding strategies, which can find an optimal value by maximizing the variance between foreground and background, so as to distinguish them.

The watershed algorithm described by Vincent [133] calculates the gradient magnitude of the image to identify potential region boundaries, and markers are selected and placed on the image. The watershed transform simulates a flooding process where water is poured into the valleys, gradually filling them up. When water from different regions meet at the same point, dams are formed, which represent segment boundaries. This guide to partitioning an image into distinct regions or objects.

The aforementioned rule-based approaches are somewhat reliant on human intervention for the fine-tuning of parameters. Recently, supervised deep learning methods become popular due to their learning-based capacity and do achieve good results in the field of cell segmentation. U-net [134] is a very successful deep learning model for segmenting

biomedical images due to its light-weighted and well-performed architecture. Subsequently, different improved versions of U-Net such as Res-UNet [135] and Dense-UNet [136] have been constructed and applied on biomedical segmentation tasks. U-Net is originally a semantic segmentation method that focuses on the separation between foreground and background. It could not identify each object individually if objects are colliding or overlapping. Cell segmentation, as a pre-step of tracking, is expected to separate each single cell from the image. This can be achieved by instance segmentation. A network with one encoder and two decoder paths, was designed based on U-Net structure, along with a watershed post-processing, to perform the instance cell segmentation [125]. It has proven to achieve top performance in the IEEE ISBI 2020 Cell Tracking Challenge. All of this related research confirms the power of the U-Net architecture for biomedical image segmentation tasks.

Once cell segmentation is performed, the next step is the cell association. It determines the linkages between cells on a frame-by-frame basis. Classic linkage methods include nearest neighbour and linear programming [131][137]. The nearest neighbour method links the cells to the nearest cell on adjacent frames based on the characteristics of cells such as intensity, shape [138], motion etc. For instance, Yan [126] proposed an algorithm named Kernel Density Estimation Mean Shift (KDE) as a linkage solution. It started by converting the initial object into density models and recursively update the mean shift factor based on a local density in the consecutive frames until a stationary location is reached. The cell closest to the stationary point was chosen as the candidate. However, a simple nearest neighbour linkage method could not solve the more complex cell behaviour well; i.e. events such as cell appearing, disappearing, merging, and splitting during the movements. Linear programming algorithms proved to successfully solve these problems [124][135][137]. In particular, the Viterbi algorithm, a classic linear programming algorithm. It is a global method to link cell trajectories based on the probabilistic functions or scoring functions obtained from feature similarities of cells. The excellent performance of the Viterbi linkage algorithm has been reported in the past years [124][139][140].

The main idea of the aforementioned methods is to extract the handcrafted cell similarities followed by a linkage method. In recent years, deep learning methods become popular for cell tracking tasks [127][141][142][143]. There are two ways to realize cell tracking using deep learning methods. One is a two-step approach that a deep learning model learns the cell migration patterns from the ground truth data, which replaces the process of handcrafted cell similarity extraction. Subsequently, a linkage method is constructed. Lugagne et al. [127] proposed a U-Net structure for learning the cell movement patterns from an *E.coli* dataset. From the predictions of the tracking model, a set of score matrices, including the probability of each pair of cells among all consecutive frames, were calculated. A nearest-neighbour

linkage method was used to associate the cells over the frames, by which a sound performance was achieved. The other way is a one-off training process. A deep learning model is trained to simultaneously learn the cell movement patterns and also the linkages. Christian et al. [141] presented a novel recurrent fully convolutional network for tracking on ISBI Cell Tracking Challenge. The network was trained on a dataset including both segmentation and tracking ground truth data for each frame. Cells in the public dataset have associated instance IDs which are used to annotate the trajectory of each instance cell. Marzieh et al. [142] proposed a framework based on a novel multi-channel feature learning model to track cells. This model can learn the spatial and temporal features simultaneously over time-lapse sequences from the ground truth data. They built a ground truth dataset by manually cropping individual cells of any consecutive frames to form image pairs. The pairs of images were annotated as true if they are the same cell in two adjacent frames, otherwise, it was annotated as false. In related work, simultaneous cell tracking in spatial and temporal feature space with a deep learning framework is gradually becoming a trend but still has its limitations. This is because the preparation of a large volume of ground truth data to train a deep learning model is very laborious. This is especially the case for the complex time-lapse sequences such as those focussing on neutrophil movement as subject in our research. Therefore, a two-step tracking strategy is most commonly used in the existing literature and is also preferred for our study.

Furthermore, recently many standalone tools were developed in support of cell tracking tasks, such as Baxter [140], TrackMate [144], PhagoSight [145], TrackPad [146], and CellProfiler [147]. In general, these open-source and interactive tools integrate many functionalities including cell segmentation, tracking, visualization, manual correction, and analysis. These tools have advanced the field of cell tracking for different cell types, however, do not fit all types. In our neutrophil dataset, complex cell events are encountered such as appearing, disappearing, merging, and splitting. Existing tools have difficulties to track these cell accurately.

Based on the motivations, for our research, we designed a cell tracking pipeline to track the complex migration dynamics of neutrophils. The proposed pipeline comprises three steps. For the first step of cell segmentation, different segmentation models were investigated. U-Net model was recognized as the most suitable model to segment neutrophils from images. Subsequently, a two-step association strategy was applied. The scoring function of cell similarities was obtained by another U-Net-based tracking model. It learned the migration patterns between adjacent frames from the ground truth data. Lastly, an extended Viterbi linkage algorithm was proposed to solve the complex linkage problem. Compared with the other state-of-the-art methods [126][127][148], our algorithm achieved the lowest mean value of 0.010 of Falsely Identified Tracker (*FIT*) and 0.173 of Falsely Identified Object

(*FIO*), together with the highest Track Purity (*TP*) value of 0.763 and a second-best Object Purity (*OP*) of 0.299 for our neutrophil dataset.

The contributions of our research are summarized as follows: (1) We built a ground truth dataset of neutrophil migration for the zebrafish model, which provides a potential chance for other researchers to investigate similar programs; (2) We developed a pipeline to solve the neutrophil tracking challenges, which integrates cell segmentation, cell motion tracking, and trajectory linkage; (3) We designed a creative Viterbi algorithm for trajectory linkage. Different heuristics were formulated for specific migration patterns of neutrophils, which aims to solve complex tracking problems.

4.2 Materials and Methods

The framework of our pipeline is shown in Fig. 4.1. At first, a pre-processing was done to improve the image quality of the time-lapse sequence. Two U-Net-based deep learning models were applied respectively for segmentation and tracking. After training the tracking model, the predictions of the possible cell location on the next frame were generated. Finally, the extended Viterbi algorithm was performed to link the trajectories based on the predictions. The trajectories were visualized through a time-lapse sequence. More methodology details were introduced in the following subsections.

4.2.1 Image Capturing and Pre-processing

The neutrophil time-lapse sequences are captured from the tail of a zebrafish after an induced tail-wounding. All the experiments with zebrafish followed the international guidelines specified by the EU Animal Protection Directive 2010/63/EU. The zebrafish in the experiment were 3 days post-fertilization (dpf) and the tail wounding was conducted [121]. The wounded tail area of specific samples was imaged using a Leica TCS SP8 confocal microscope (Leica Microsystems) with a 10× objective (N.A. 0.40). Neutrophils, localized within an area of 200μm from the wounding edge toward the body trunk, were counted as recruited cells. Under the microscope, the zebrafish tails were scanned from top to bottom. A total 8-layer 3D stack at each time point was captured. The image stack size is $512 \times 512 \times 8$. The layer interval is 5μm-6μm, while the thickness of the tail is 35μm-42μm. For each sample, a 2-hour time-lapse sequence was captured, with a time interval of one minute. Thus, there are 120 frames for each time-lapse sequence. Each time-lapse sequence contains around 10 to 40 cells. We captured 20 time-lapse sequences in total.

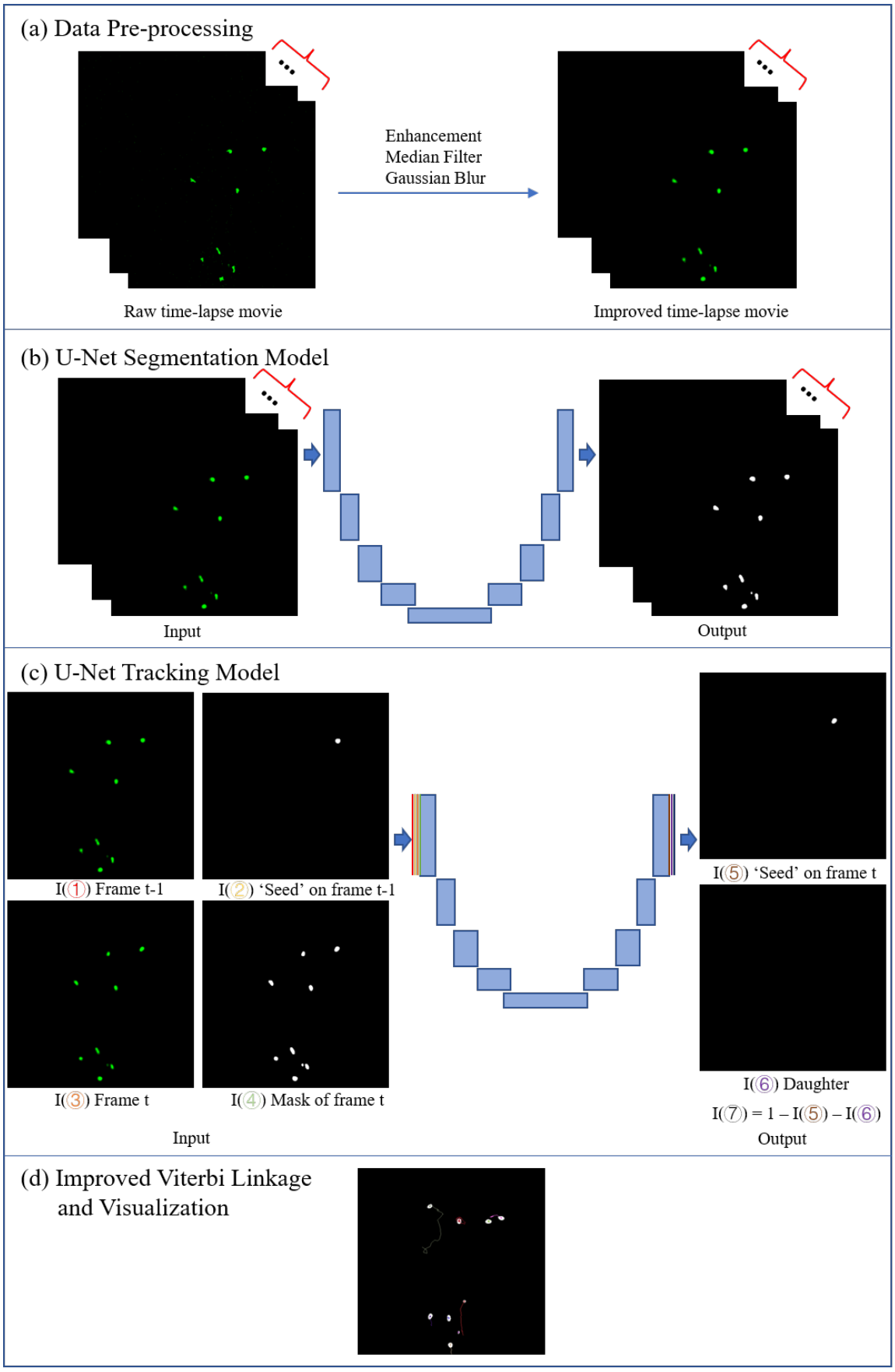


Fig. 4.1 The pipeline of cell tracking. (a) Data pre-processing. (b) U-Net segmentation model. (c) U-Net tracking model. I(①)I(②)I(③)I(④) represent the four inputs and I(⑤)I(⑥)I(⑦) represent the three outputs of the model. I(⑦) is calculated by the reversion of the other two outputs I(⑤) and I(⑥). (d) Viterbi linkage and visualization.

The raw data we obtained are thus 3D + t time-lapse sequences. Due to the relative undersampling in Z-axis direction, for this paper, we intend to focus on solving tracking problem for neutrophils in 2D space and to evaluate how well a 2D cell tracking method can solve the problem. To that end, a maximum intensity projection was applied to the 8-layer image stack so as to derive a 2D projection data at each time point. In Fig. 4.2 (a), the first projected frame of one sequence is depicted. The intensity is relatively low and the cells are not clearly visible. Therefore, image pre-processing steps were applied to improve the image quality. First, we enhanced the image contrast to highlight the cell; cf. Fig. 4.2 (b). However, this also enhanced the background noise. Therefore, to denoise a median filter was used; cf. Fig. 4.2 (c). Subsequently, we chose a Gaussian blur to smooth the cell surface area; cf. Fig. 4.2 (d). The corresponding magnified area in the red box of Fig. 4.2 (a)(b)(c)(d), are shown in Fig. 4.2 (e)(f)(g)(h) respectively, so as to clearly present the image details.

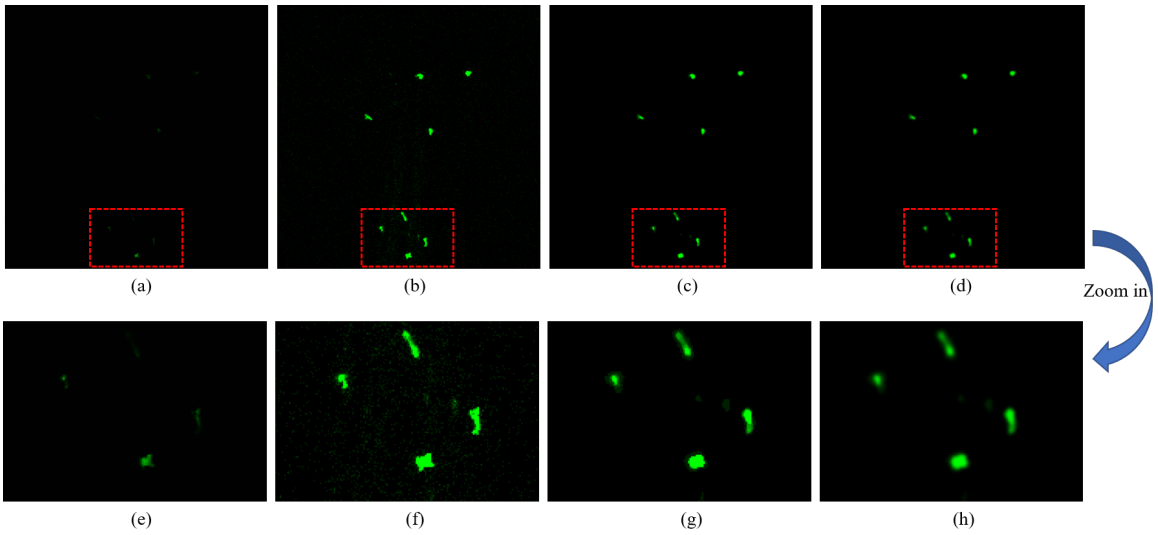


Fig. 4.2 Data pre-processing. (a) One frame of the raw data. (b) Image contrast enhancement. (c) Image denoising with median filter. (d) Image smooth with Gaussian blur. (e)(f)(g)(h) are the corresponding zoom-in areas of the red boxes in (a)(b)(c)(d), respectively.

4.2.2 Cell Segmentation

4.2.2.1 Ground Truth Labelling for Segmentation

Annotated images should be prepared. They are used for the stages of both training and evaluation of segmentation models. We used AnnotatorJ [149] to label each cell. This is a semiautomatic tool to find an approximate cell mask by applying U-Net-based pre-segmentation. It follows a manual correction to precisely create the masks of individual cells. Users can create their own annotated dataset based on different options: binary mask annotation and instance mask annotation. In our study, we created 240 images of instance

mask annotations. Binary mask annotations can be obtained by thresholding. Both binary and instance mask annotations were used to train the U-Net semantic segmentation model and instance segmentation model, respectively, as well as evaluate the segmentation performance.

4.2.2.2 Segmentation Models

A good segmentation model is expected to identify each cell, without missing cells, over-segmentation, and under-segmentation. In our study, we intend to find an accurate segmentation model for the neutrophil image dataset. Four models were built and compared.

First, we implemented a U-Net model (U-Net 1). It has a similar structure to the original U-Net model [134] but without an overlap-tile strategy. Both the input and output sizes of the network were set to 512 × 512, which is the same as the raw image size. U-Net 1 was trained over 10 epochs of 300 steps on the Tensorflow platform on a dedicated server equipped with two NVidia GeForce GTX 2070/8 GB GPUs. The batch size was set to 1 due to the large image size and the limitation of GPU memory. Data augmentation was applied in order to increase the number of training sets and avoid overfitting of the network. In our case, width and height shift, vertical and horizontal flip were used to augment our dataset because these transforms do not change the cell morphology. For our dataset, the pixel number of the foreground (positive sample) and the background (negative sample) are unbalanced. The negative samples occupy the most area of the image. The consequence is that no candidate cells would be predicted and generated by the network. Thus, selecting a suitable loss function plays a key role during training. Dice loss function and focal loss function [150][151] have demonstrated to be very effective in solving class unbalanced problems and thus were compared these two in our experiments. Subsequently, the original U-Net model (U-Net 2) [134] was implemented. We intend to compare both nets. The differences between U-Net 1 and U-Net 2 are: U-Net 2 works with an overlap-tile strategy. The input and output sizes are 512 × 512 and 388 × 388, respectively. The output images need to be resized to 512 × 512 so as to get the same size images as our raw data.

In addition, an instance U-Net segmentation model (U-Net 3) [125] was adopted. The structure of U-Net 3 has one encoder path and two decoder paths which aim to do instance-based segmentation task. This model was involved to test the possibility of using instance-based segmentation model to solve the under-segmentation problems encountered in the two U-Net-based semantic segmentation models. In addition to the deep learning models, a rule-based Watershed Masked Clustering (WMC) segmentation algorithm [152] was included since it has been used with a similar cell tracking task.

4.2.2.3 Evaluations for Segmentation

In order to evaluate the performance of the segmentation models, we collected five commonly-used evaluation metrics [153]. The first one is the Jaccard index, which is defined as:

$$IoU(T,E) = \frac{T \cap E}{T \cup E} \tag{4.1}$$

T represents the ground truth image, E represents the predicted segmentation image. The metric relies on the computation of Intersection-over-Union (IoU) between a pair of objects on T and E, respectively. IoU measures pixel-wise foreground overlap between them.

The second commonly-used evaluation metric is the $F1$ score. We measured the cell number of True Positive (TP), False Negative (FN), and False Positive (FP). $F1$ score is computed by:

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{4.2}$$

Where TP represents the corresponding pair of objects on both T and E. FP represents the objects which are predicted on E while not annotated on T. FN is the objects which are missed by the segmentation model compared with T.

FN is considered as a third evaluation criterion individually. Because cell missing is not expected in the tracking task. It causes a loss of target and a wrong connection of trajectory. Besides IoU, $F1$ score, and FN, we also counted the number of under-segmented and over-segmented cells. These two measurements reflect the ability of the segmentation model to solve the under-segmentation and over-segmentation problems.

In principle, a better performance is achieved with a higher IoU and $F1$ score, together with a lower cell number of FN, under-segmentation, and over-segmentation. In practice, however, all of these metrics are interrelated. It depends on the value of threshold t that sets the predicted probability of each pixel above t as foreground and below t as background. If t is too high, the number of pixels that are recognized as the foreground decreases. It causes the predicted cells to have a much smaller surface area, resulting in a lower IoU value, a higher number of over-segmented cells, and a lower number of under-segmented cells. In addition, some small cells are missed, which caused FN to increase further. Similarly, if t is too low, we got results on the contrary. In our experiment, we set $t = 0.1$ in order to keep a balance among all of these evaluations.

4.2.3 Cell Tracking

4.2.3.1 Ground Truth Labelling for Tracking

For a supervised deep learning model for tracking, the ground truth dataset is required. In support of our study so as to obtain such dataset, a manual tracking was performed by experts [121] using the TrackMate plugin [144]. For each time-lapse sequence, the expert labeled 3 to 9 trajectories. In total, there are 110 trajectories from all 20 time-lapse sequences. The (x, y) coordinates of each trajectory were recorded, which were used to localize the cells' positions in the time-lapse sequence.

4.2.3.2 U-Net for Tracking and Evaluations

A U-Net-based deep learning model was adopted to learn the features of cell motion from adjacent frames of the ground truth trajectories.

The U-Net-based model was designed for tracking *E.coli* cells trapped in mother machine microfluidic chambers [127]. In this case, the cell movements were constrained in the vertical direction. It is a simplified cell tracking problem. We intend to apply and extend this idea to a more complicated case. The U-Net architecture has four inputs and three outputs (cf. Fig. 4.1 (c)). So, for every cell at every time point of the ground truth trajectory, there are four inputs and three outputs. The four inputs include the current frame, the segmentation mask of the current frame, the previous frame, and the 'seed' cell that we want to track from the previous frame. The three outputs include the 'seed' cell on the current frame, the potential daughter cell on the current frame if a division just happened and the mask image of the reversion of the other two output images. For our dataset, we generated the training set including input and output images from ground truth trajectories based on the recorded coordinates. Cell division events do not occur in our case, thus all of the daughter images are empty.

In this study, we divided all 110 ground truth trajectories into training, validation and test sets based on a ratio 8:1:1. We compared the performance of three tracking models on test set. The first one was the pretrained model which was already trained on yeast cells [127]. This model was trained over 400 epochs of 250 steps per epoch with a batch size of 5 and a learning rate of $1*10^{-4}$. We used this model to evaluate the performance of test set directly. Moreover, we trained the other two models using TensorFlow on GPUs. One concerned a model from scratch. All the parameters of the network are initialized randomly and we trained this model based on our neutrophil training set. This model was trained for 20 epochs with mini-batches of size 1 and 250 steps per epoch and a learning rate of $1*10^{-5}$. The other model concerned a fine-tuned model. We fine-tuned the pretrained model on our training set, so as to update the parameters of the network. This network was trained for 20 epochs with a

batch size of 1 and a learning rate of $1*10^{-5}$. All of these hyperparameters were set based on experimental fine-tuning and empirical knowledge.

After we trained these three U-Net models, we evaluated the performance of each model by counting the number of incorrectly predicted cells on the test set.

4.2.3.3 Extended Viterbi Linkage Method

After the training, the images of predictions were generated from the tracking models. The score matrices were computed from these predictions. The pixels attributed to the cell in each predicted image were matched to the pixels in the segmentation mask of the corresponding frame. The overlap area of each pair of cells on the two images was calculated and a score matrix for this specific time point was generated. Fig. 4.3 shows three simple examples of score matrices among four consecutive frames. The number of rows represents the number of cells in frame t, and the number of columns represents the number of cells in frame $t+1$. The values in the matrix represent the score between each pair of cells of the two adjacent frames. For each time-lapse sequence in our study, it has 120 frames in total. Therefore, we can obtain 119 adjacent score matrices for each time-lapse sequence.

Frame t + 1

Frame t	0	1	2	3	4
0	0.9	0	0	0	0
1	0	0.1	0.7	0	0
2	0	0.8	0.2	0	0
3	0	0	0	0.7	0.1

(a)

Frame t + 2

Frame t+1	0	1	2	3	4
0	0.9	0	0	0	0
1	0	0.1	0.7	0	0
2	0	0.8	0.2	0	0
3	0	0	0	0.7	0.1
4	0	0	0	0	0

(b)

Frame t + 3

Frame t+2	0	1	2	3	4
0	0.9	0	0	0	0
1	0	0.1	0.7	0	0
2	0	0.8	0.2	0	0
3	0	0	0	0.7	0.1
4	0	0	0	0	0

(c)

Fig. 4.3 Score matrices. (a) Matrix between frame t and $t+1$. There are four cells in frame t and five cells in frame $t+1$. (b) Matrix between frame $t+1$ and $t+2$. (c) Matrix between frame $t+2$ and $t+3$. For each value in the matrix represents the computed score of each pair of cells on adjacent frames.

Based on these score matrices, our aim was to find the optimal path for each tracked cell that achieved the highest accumulated score over the whole time-lapse sequence. A linkage method was required subsequently.

Behavioural Patterns of neutrophil migration in the time-lapse sequence Before introducing the linkage method, identifying the possible behavioural patterns of neutrophils can help us design a customized strategy for each pattern. We have observed and identified the following patterns of neutrophil movement: (pattern 1) migration over all frames in the

sequence; (pattern 2) newly appearing cells; (pattern 3) disappearing cells; and (pattern 4,5) merge and split of cells. These patterns are depicted in Fig. 4.4. For our 120-frame time-lapse sequence, very few neutrophils are always in the field of view of the images (pattern 1). In most cases, cells go in/out of view at some time point, which are defined as appearing and disappearing of cells. Merge and split patterns are the key problems that we aim to solve with some strategies. In this study, an extended Viterbi linkage method was proposed. The use of Viterbi algorithm was motivated by the power of the global track linkage and its superior performance in Muscle Stem Cells (MuSCs) and myoblasts cell tracking problems in [140]. The optimal trajectory for each cell is ensured by solving a dynamic programming problem over a state space diagram.

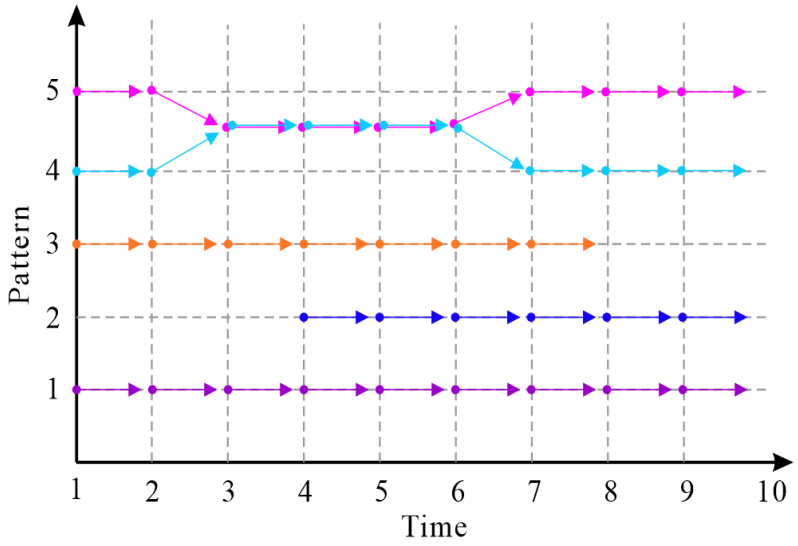


Fig. 4.4 The cell behavioural patterns in neutrophil time-lapse sequences. Pattern 1: migration over all frames; Pattern 2: newly appearing; Pattern 3: disappearing; Pattern 4 and 5: merge and split.

Viterbi Basic Rules The basic logic of the Viterbi algorithm comprises two phases, including a routing phase (forward) and a best track retrieval phase (backward). A routing phase starts from each cell on the first frame, iteratively, to ensure the optimal path based on the accumulated score on each node of the state space diagram. Fig. 4.5 is an example of the state space diagram. It is a fully connected network. It shows all cell nodes of four consecutive frames in Fig. 4.3. Suppose these four frames are the first four frames in a time-lapse sequence, thus t is equal to 1. We first looped each cell node (from $A0$ to $A3$) in frame 1 to find the optimal path through all frames. The values in score matrices correspond to the weights on the edges between each pair of nodes. Therefore, we can calculate the

accumulated score for each node. For example, we obtained the accumulated score of all the possible paths from node $A0$ to $C0$ via column B. The path $A0$-$B0$-$C0$ got the highest score of 0.72 (0.9×0.8). The other paths $A0$-Bi-$C0$ (i is from 1 to 4) reached a score of 0. We only maintained the best path to the next step and other paths were discarded in order to reduce the complexity of the algorithm. Similarly, from node $A0$ to $C1$, the optimal path is $A0$-$B0$-$C1$ with the highest score of 0.09 (0.9×0.1). From $A0$ to $C2$, $C3$, $C4$, the best paths are $A0$-Bi-$C2$ (0), $A0$-Bi-$C3$ (0), $A0$-Bi-$C4$ (0), respectively. Bi means no matter which B node the paths pass, the score of all paths was 0. Then we calculated the score from node $A0$ to all the nodes on column D based on the formerly remained best paths via columns B and C. These best paths are: $A0$-$B0$-$C0$-$D0$ (0.9×0.8×0.85), $A0$-$B0$-$C1$-$D1$ (0.9×0.1×0.2), $A0$-$B0$-$C1$-$D2$ (0.9×0.1×0.7), $A0$-$B0$-Ci-$D3$ (0). Ci represents no matter which C node the paths pass, the score of all paths is 0. Suppose column D is the last frame of the sequence, then the process enters the best track retrieval phase when the routing process reaches the last frame. Among all nodes D, $D0$ was the node with the highest score (0.9×0.8×0.85) and the path $D0$-$C0$-$B0$-$A0$ was retrieved as the optimal path. In the end, we obtained the optimal path from $A0$ to $D0$, which was shown in green nodes and edges in Fig. 4.5. In this way, other trajectories started from $A1$, $A2$ and $A3$ were also obtained and highlighted in different colours.

After all the cell trajectories in frame 1 were found, the cells that not passed by any of those trajectories in the next frame were treated as the newly appearing cells. In Fig. 4.5, $B4$ is a new cell appearing in frame 2, the same procedure was applied to find the best trajectory. For the cell which is disappearing, the corresponding value in the score matrix between itself and all the candidate cells in the next frame is very low. Here, we set a truncated threshold $T_{truncated}$. The score of a cell that is lower than $T_{truncated}$ was treated as a disappeared cell, referred to as the cell out of the field of view.

Dealing with Cells Merge/Split One drawback of the basic Viterbi algorithm is that it cannot deal with merge/split problems. Once the cell trajectories merged due to the collision or occlusion of cells at the same time point, the trajectories never split because they shared the same score matrices after the merge happened and ended up with the same optimal trajectory. This circumstance occurs as Viterbi treats each track to be independent from one another. However, in fact, the calculation of an individual cell trajectory is largely affected by the surrounding tracks. Therefore, to solve this problem, an extended Viterbi algorithm was developed to incorporate the surrounding tracks. To that end, three rules were set on top of the basic Viterbi algorithm to manage the merge and split scenarios among cell trajectories.

Rule 1: an alternative path routing is developed. It enables the track routing to merge and split with two specific conditions 1 and 2.

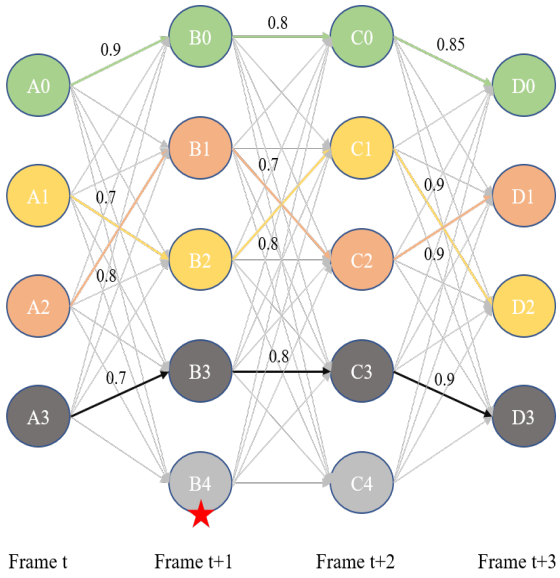


Fig. 4.5 State space diagram of four consecutive frames. The different four colours represent four trajectories which derived from basic Viterbi algorithm. This state space diagram also corresponds to the score matrices in Fig. 4.3. The red star represents a new start of the other trajectory in the loop.

Rule 2: a threshold T is created to support the alternative path routing. It provides a reference for merge/split decisions.

In the routing phase, the best node to frame $t-1$ could possibly be occupied by one or more cell tracks. In such circumstances, connection to that node would only be allowed if:

Condition 1: The total score of the routing track is above the threshold T up to that layer, or;

Condition 2: All occupied cell tracks' scores on the specific node are below the threshold T up to that layer.

These rules are also applied to the last layer in the best track retrieval phase.

The initial value of the threshold is an adjustable hyperparameter $T\_initial$. The threshold value has to be discounted for every additional frame because the connection score has a decreasing trend from probability multiplication when routing to each new frame. The discount rate is calculated with the following equation.

$$DiscountThreshold = T_{initial}{}^{frame\_number-1} \tag{4.3}$$

In Fig. 4.6 these concepts are illustrated.

Suppose the track of Cell 1 is settled in the red line in Fig. 4.6 and a Cell 2 was considered to be linked. When considering node connections from frame 3 to 2, the upper node in frame

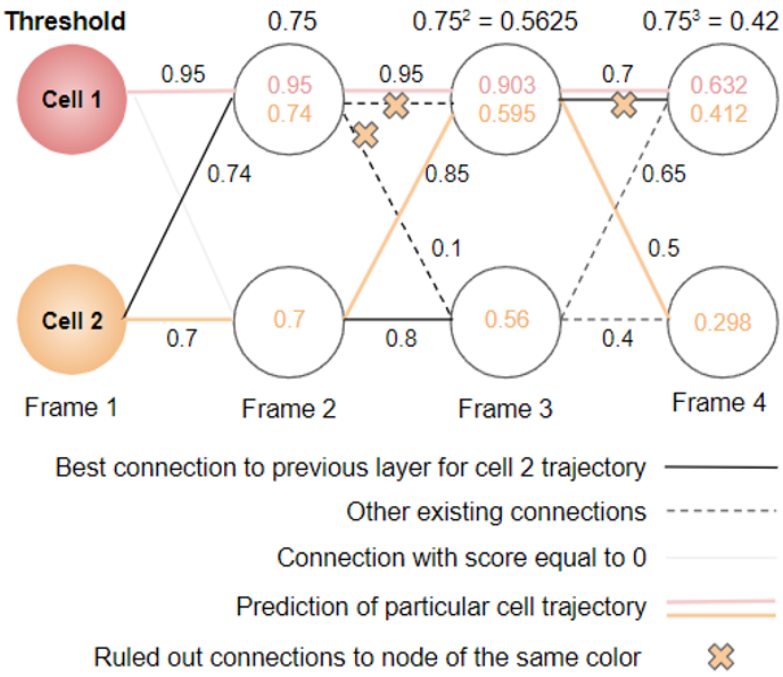


Fig. 4.6 Alternative path selection example.

2 is originally the best node to be connected for orange cell track, but it is occupied by the red cell track with a score higher than threshold. On the other hand, the routing track score of orange is 0.74 which is lower than threshold 0.75. Thus, connection to that node would be ruled out and a connection to the lower node in frame 2 (the second best connection) is established. This is illustrated by the solid orange line from frame 3 to 2 in Fig. 4.6. In the best track retrieval phase, which starts from frame 4, although the upper node has a higher overall score of 0.412 (0.412>0.298), it is lower than the discount threshold 0.42. It does not meet **Condition 1**. The score of 0.632 in the upper node of frame 4 in the red cell track is higher than 0.42, which is not meet **Condition 2**. Therefore, the lower node is chosen as the end node of the track (score 0.298) and a split event in frame 3 is detected for the two cell tracks.

Rule 3: a reroute strategy is constructed. During the process of routing, a circumstance occurs where the later routed track is a much better match compared to an occupied node in an earlier track. The reroute strategy allows an earlier routed cell track to be reset. To exemplify, let us consider the yellow Cell 3 is being routed after the orange cell, and the best track of the yellow cell is determined as shown in Fig. 4.7. It shares the second node with the orange cell track in frame 2 at the beginning. Nevertheless, only the yellow track score is above the threshold (0.9>0.75), which meets **Condition 1**. Therefore, the orange cell track is sent to reroute as it is occupying a node that is more suitable for the yellow track. While

rerouting the orange track, it is connected to node 3 in frame 2 since the first two nodes are more matched to the red and yellow tracks respectively. The orange and yellow tracks are sharing the same node in the next frame on the grounds that both of them are above the threshold of frame 3. In the last frame, they share the same node again because both of their scores are below the threshold of frame 4.

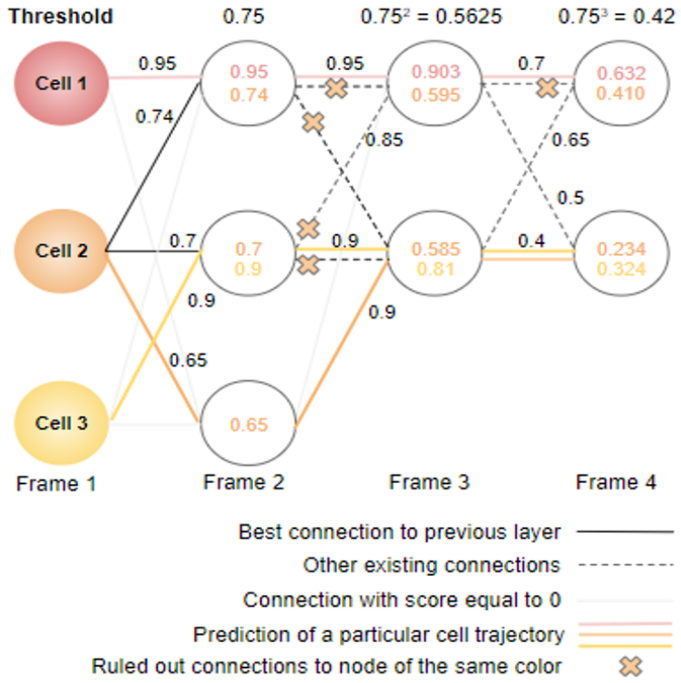


Fig. 4.7 The orange route is being rerouted due to a collision with the yellow track. The new route connects to a different node in frame 2.

Dealing with Cells Get into/out of View In Fig. 4.8, a situation is depicted where a new Cell 4 is being detected at frame 3, which will be routed largely similar to the rules mentioned except with a start frame discount. A start frame discount is a hyperparameter. Here it was set to 0.8 as an example. It is applied during score calculation because cells starting in later frames have fewer multiplications than track begins at an earlier frame. For instance, Cell 1 has two multiplications up to frame 4 (0.95×0.95×0.7). In order for Cell 4 to have a fair comparison, we set the hyperparameter to two multiplications (0.9×0.8×0.8) as well.

The cell trajectory terminates in three conditions. The first one is when it reaches the last frame of the time-lapse sequence. The second condition is when the score in the next frame is all zero. The third condition is when the next frame does not have an available cell due to nodes owned by other trajectories and the merge criteria is not fulfilled.

Additional Occupation Rules In the process of rerouting, a track that starts in a later frame (Cell 6) may occupy the rerouting path of a track beginning from an earlier frame (Cell

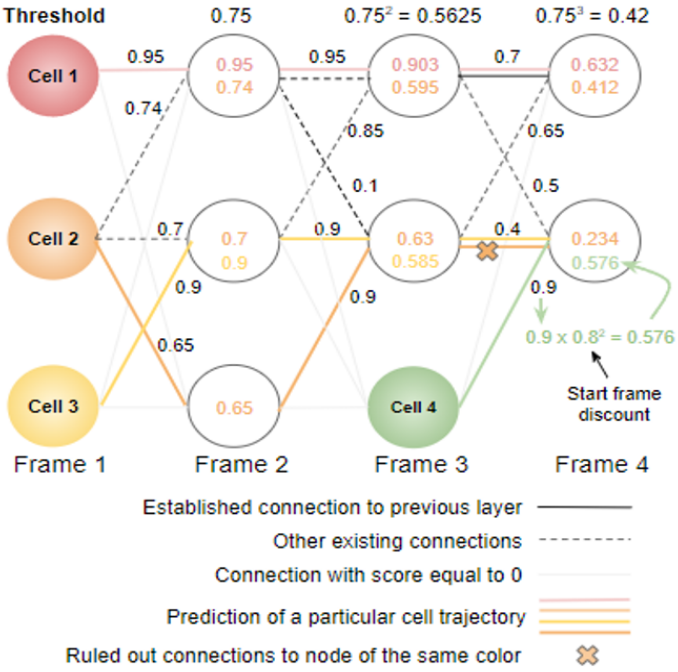


Fig. 4.8 A new cell (green) is detected in frame 3. It goes through a start frame discount for its connection, subsequently, takes over the lower node from yellow in frame 4.

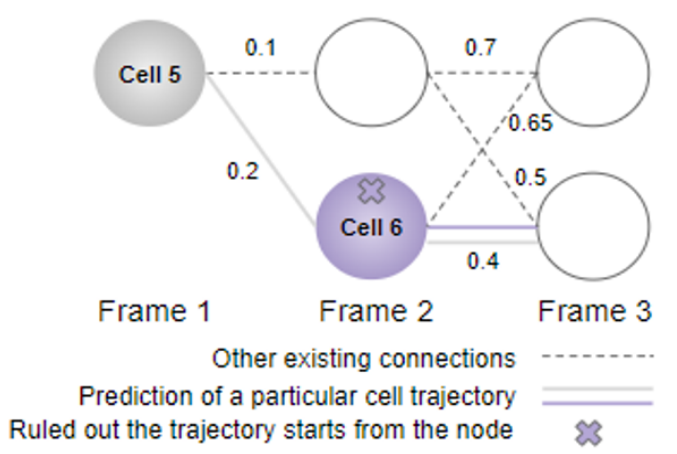


Fig. 4.9 Cell 5 established a connection to the beginning of Cell 6 during reroute. Cell 6 is then removed.

5) as shown in Fig. 4.9. In such case, the track that starts earlier would take over the later one even if it has a low connection score, the later track will be removed from track results. This is because the cell in the later trajectory no longer satisfies the definition of a new cell as it is owned by an existing trajectory. This situation only occurs in the reroute phase.

4.2.3.4 Evaluations for Linkage Trajectories

After implementing the extended Viterbi algorithm for each time-lapse sequence, all of the predicted cell trajectories of each sequence were saved, the same as the ground truth trajectories. Evaluation metrics were used to measure the performance of the linkage methods. Four measures [154] have been selected to evaluate how well a tracker identifies objects. These are:

Track purity (*TP*): a measure of the degree to which the predicted tracks ($\in$) follow ground truth objects (*GT*). It is the ratio of frames that $\in$ correctly identifies *GT* to the total number of frames $\in$ exists.

Object purity (*OP*): is a measure of the degree to which the ground truth objects (*GT*) are followed by predicted tracks ($\in$). It is the ratio of frames that *GT* is correctly identified by $\in$ to the total number of frames *GT* exists.

Falsely Identified Tracker (*FIT*): a measure of the degree to which the *GT* are tracked by the incorrect $\in$.

Falsely Identified Object (*FIO*): a measure of how often the $\in$ is tracking a different cell than the *GT* it was matched to.

In addition, the average length of tracks was computed.

Average length of tracks: the ratio of the total length of all predicted tracks to the number of predicted tracks.

4.3 Results and Discussion

We extracted results from each part of the pipeline: cell segmentation, cell motion tracking, and trajectory linkage, respectively.

4.3.1 Segmentation Results

240 ground truth segmentation images were divided into train-validation set (192 images) and test set (48 images) based on the ratio of 8:2. 48 images were used to test the segmentation model as the "unseen" data. The train-validation set was split to training set (153 images) and validation set (39 images) in a ratio of 8:2. Table 4.1 shows the performance of four segmentation models on 48 test images. The U-Net 1 was the structure that finally used in our study. It achieved 0.912 of *IoU* and 0.985 of *F1* score, which outperformed the other methods compared. U-Net 2 was the original U-Net model [134]. The input and output sizes of this model were 572×572 and 388×388, respectively. Thus, the output masks of this model should be resized to 512×512 as the original image size. During this up-sampling

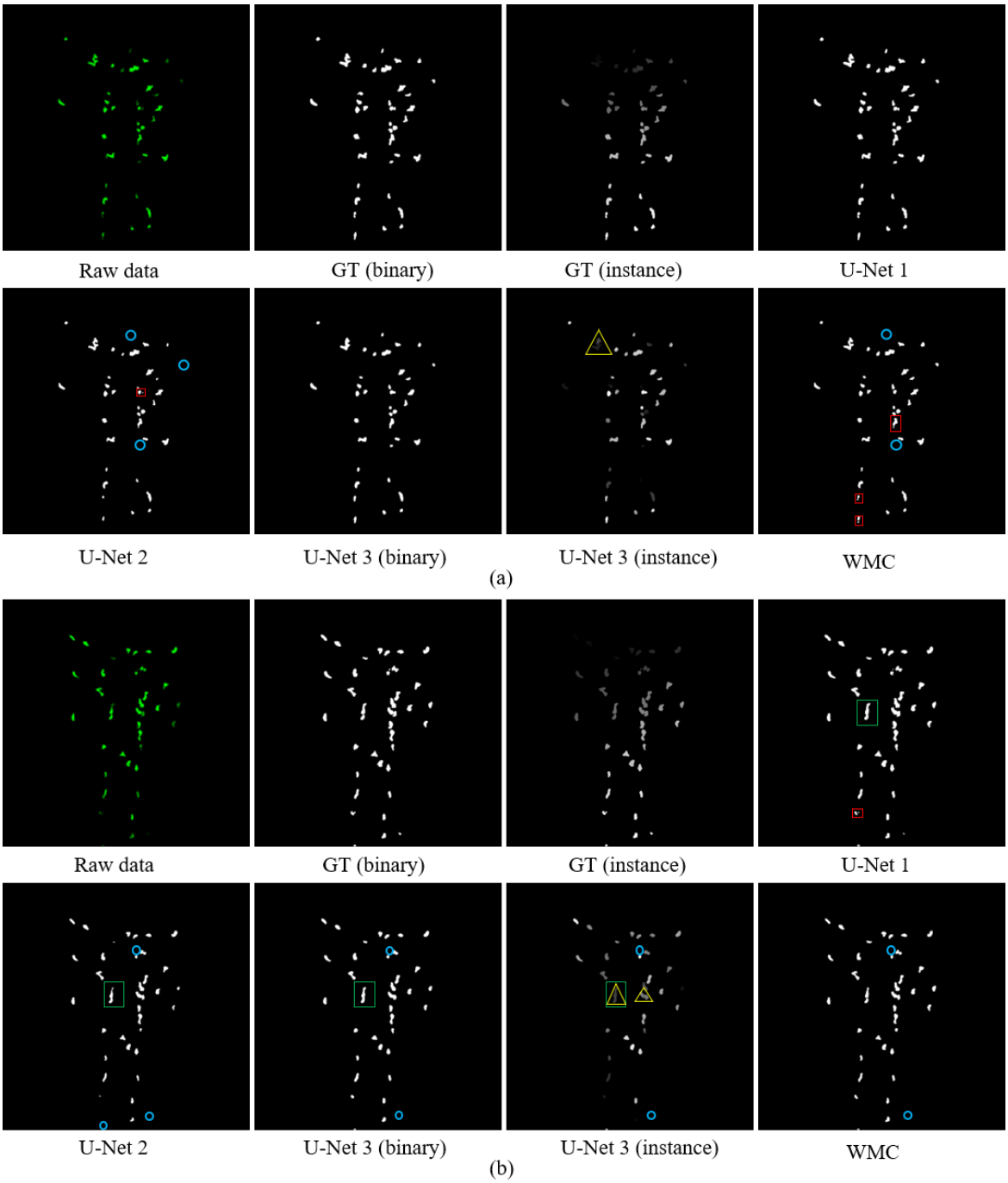


Fig. 4.10 (a) and (b) show two examples of segmentation results. It shows raw data, ground truth of binary mask and instance mask, respectively. The five segmentation results (U-Net 1, U-Net 2, U-Net 3 (binary), U-Net 3 (instance), Watershed) are shown. The red rectangular boxes represent the over-segmentation samples. The green rectangles represent the under-segmentation samples. The blue circles represent the *FN*. The yellow triangles represent the samples that are expected to separate by the instance segmentation model, however, not work so well yet.

resizing process, some mask information was not well preserved. This caused the predicted cells to have a smaller surface area compared with U-Net 1. It resulted in a lower *IoU* score of 0.697. Some small cells were missed as well which resulted in a high number of *FN*. U-Net 2 had less under-segmented cells and more over-segmented cells than U-Net 1 due to its smaller surface area. For U-Net 3 model, the binary masks were obtained first and a post-processing based on watershed transform was applied to obtain the instance masks. In the experiments, both the binary masks and the instance masks on test set were obtained and used to compute the evaluation metrics. The results in Table 4.1 show that, compared to U-Net 1, U-Net 3 achieved a lower *IoU* but a comparable *F1* score. U-Net 3 (binary mask) achieved a small number of under-segmented and over-segmented cells but with a little bit higher *FN* while U-Net 3 (instance mask) achieved higher values of *FN*, under-segmentation and over-segmentation. The experiments show that the complex instance-based segmentation model does not outperform much compared with a simple U-Net model. Moreover, it took 6-7 times more computation time than U-Net 1. We fine-tuned the parameters of the WMC segmentation algorithm for our neutrophil dataset, it yielded an acceptable accuracy of 0.725 of *IoU*, 0.976 of *F1* score, and 34 *FN* numbers. Besides, the WMC algorithm has only one under-segmented cell, demonstrating that it performs well in separating the touched cells. However, it has a relatively high number of 26 over-segmented cells compared with other models and has the lowest efficiency.

Table 4.1 The performance of five segmentation models on the test set.

Models	***IoU***	***F1* score**	***FN***	**Merge**	**Split**	**Time(s)**
U-Net 1	0.912	0.985	8	18	5	64.00
U-Net 2 [134]	0.697	0.956	53	6	7	313.47
U-Net 3 (binary) [125]	0.695	0.984	15	12	4	402.50
U-Net 3 (instance) [125]	0.682	0.985	16	28	12	402.50
Watershed [152]	0.725	0.976	34	1	26	702.58

Fig. 4.10 shows the two examples of the different segmentation results. U-Net 1 obtained the most closed segmentation results compared to the ground truth. The surface area of the predicted cells in U-Net 2 is much smaller which leads to more *FN* and over-segmented samples. In Fig. 4.10, this is indicated by blue circles and red rectangles. The WMC algorithm also resulted in more *FN* and over-segmented objects. This corresponds to the results as presented in Table 4.1. U-Net 3 (binary) was the most comparable model to U-Net 1, however, with a lower computational efficiency. U-Net 3 (instance) was expected to

separate the touched cells. This worked well in some cases. But sometimes it failed as shown with yellow triangles in Fig. 4.10.

All of these segmentation results have pros and cons. We selected the one which performed better using less computation time for the cell tracking task in the next step. Based on the analysis above, U-Net 1 was selected as the segmentation model in our pipeline.

4.3.2 Tracking Results

In order to predict the most possible location of each candidate cell on test set, three models including a pretrained model, a model from scratch and a fine-tuned model were trained. The predicted candidate cell was matched to both original image and segmentation mask to find if they match the corresponding region of interest (ROI) of the cell. If so, it was counted as a correct prediction; otherwise, a wrong prediction. In this process, the original image helps to provide the intensity value and segmentation mask provides the location of the cell. Both of them are necessary because sometimes there is more than one cell ROI predicted from the output of the model. In this case, we selected the ROI with the highest average intensity as the final localized cell. Subsequently, the total number of cells (predictions) on each of the ground truth trajectories in the test set was also counted. The error rate is calculated by erroneous predictions divided by the total number of predictions.

Table 4.2 The predicted errors of each tracking model on the test set.

Models	**The number of erroneous predictions**	**The number of predictions**	**Error rate**
Model from scratch	572	11641	0.049
Pretrained model	781	11641	0.067
Fine-tuned model	578	11641	0.050

Table 4.2 shows the error rate of the three models. In the total number of 11614 predictions, there were 572, 781 and 578 cells mis-predicted for model from scratch, pretrained model and fine-tuned model, respectively. From Table 4.2, the pretrained model produced the highest error rate of 0.067. This seems to be reasonable as this model was trained on an *E.coli* cells dataset, which is a cell type totally different from neutrophils. The model from scratch and the fine-tuned model obtained a comparable error rate whereby the model from scratch has the lowest one. In this study, we selected the outputs of fine-tuned model to do the next step of cell linkage rather than the model from scratch. Because, in the follow-up

experiments, we found that the tracking trajectories that were obtained on the basis of the model from scratch have many more linkage errors. The ideal situation is that there should be only one predicted cell in each output image due to the fact that our ground truth trajectories have no splitting annotations. However, the model trained from scratch predicts more than one cell in most output images. The extra predicted cells do not increase the error rate but it does affect the result of the Viterbi linkage process. Clearly, given our data, the fine-tuned model is much more robust than the model from scratch. Therefore, it is chosen for our study.

4.3.3 Linkage Results

For a 120-frame time-lapse sequence, a total number of 119 score matrices were obtained. The score was calculated based on the predictions of the tracking model and was defined as the overlapping area between the predicted cell on the current frame with each cell on the previous frame. Based on the score matrices, we performed our extended Viterbi algorithm, as well as several different linkage methods to compare the performance. The first one is the DeLTA linkage method [127]. DeLTA linkage is a straightforward method to link the trajectories. It starts from every cell in the first frame, to find the candidate cell in the next frame with the highest score value. If the values of two candidate cells are the same, it chooses to link the first candidate cell directly while the other cell can start a new trajectory. Each cell node can only be passed once by one trajectory. The second involved linkage method is the Hungarian algorithm [148]. The Hungarian algorithm aims to find the optimal path in the score matrices by solving an assignment problem. For the linkage process, it starts from each cell in the first frame to link the candidate cell frame by frame, until no candidate cell can be linked. The end condition depends on the configured threshold. In the experiment, the threshold was set to 0.001. Each frame was looped and the unpassed cell was treated as the start of a new trajectory. Comparing the aforementioned two linkage methods, the Viterbi algorithm is more powerful due to its concept of optimization of a global linkage. Viterbi optimizes the path by taking the relation of all frames into account, not only the adjacent frames. A basic Viterbi algorithm was also taken into our comparison in order to show the improvement of our extended Viterbi algorithm. In addition, a rule-based tracking algorithm named KDE with mean shift [126] was also involved, so as to demonstrate the outperformance of our pipeline incorporating a deep-learning-based tracking model and an extended Viterbi linkage method. Fig. 4.11 shows the performance comparison of the methods in our evaluation.

In Fig. 4.11, the box plots of the different linkage methods with five selected evaluation metrics on the test set are shown. The evaluation metrics *FIT*, *FIO*, *TP*, *OP* are normalized, in addition, average track length is computed. The normalized values represent the average

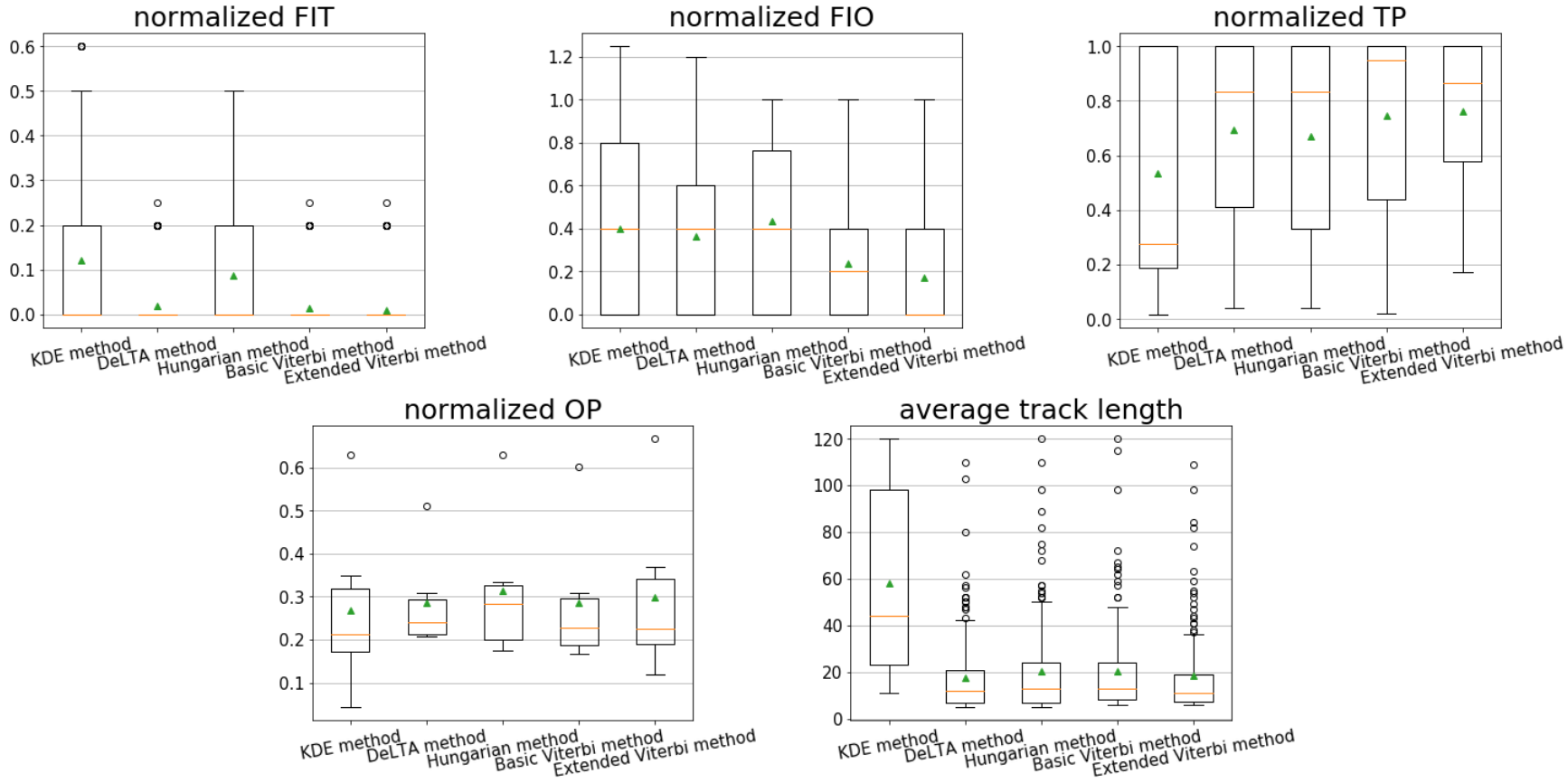


Fig. 4.11 The performance of different linkage methods with the selected evaluation metrics on the test set. The metrics are: falsely identified track (*FIT*), falsely identified object (*FIO*), track purity (*TP*), object purity (*OP*) and average predicted track length. In the boxplots, green triangle shows the mean values, the orange line is the median value, the box represents the first to third quartile of the data, the whiskers indicate the lower and upper extreme (1.5 times the interquartile range) and the black circles are the outliers.

number of errors (*FIT*, *FIO*) and purities (*TP*, *OP*) per frame per cell on the test set from the ground truth. In the box plot (cf. Fig. 4.11), the green triangle represents the mean value of each evaluation. The resulting values are presented in Table 4.3; these mean values correspond to the Fig. 4.11. From Table 4.3, our extended Viterbi achieved the lowest mean value of 0.010 of *FIT* and 0.173 of *FIO* of the methods in our comparison. For purity evaluations, the extended Viterbi also yielded the highest *TP* value of 0.763 as well as a second-best *OP* of 0.299. From the experiments, our extended Viterbi algorithm shows superior performance and produces a better cell tracking result.

Table 4.3 The performance of each linkage method. The footnote is the rank for all compared methods of each evaluation metric.

Algorithms	**Normalized *FIT***	**Normalized *FIO***	**Normalized *TP***	**Normalized *OP***	**Average track length**
KDE [126]	0.122	0.399	0.537	0.268	57.844
DeLTA [127]	0.020	0.365	0.694	0.285	17.520
Hungarian [148]	0.087	0.437	0.668	0.314	20.356
Basic Viterbi	0.015	0.237	0.747	0.286	20.158
Extended Viterbi	0.010[1]	0.173[1]	0.763[1]	0.299[2]	18.427[4]

The results in Fig. 4.11 show that compared with a rule-based KDE method, a deep learning-based tracking model together with any of the aforementioned linkage methods performs much better for our cell tracking task. KDE method obtained a highest *FIT* and a lowest *TP* and *OP*. It only outperformed 0.01 of *FIO* compared to the Hungarian algorithm. This is due to the features used in the KDE method; i.e. the kernel density of each cell. Only using kernel density is limited to distinguish the cell tracks on neutrophil dataset because of the complex morphology and migration behaviour of neutrophils. However, a deep learning model can learn more features of cell morphology and movement from the ground truth trajectories.

In Fig. 4.11, DeLTA, Hungarian, basic Viterbi and extended Viterbi methods show to achieve a comparable average track length. However, KDE algorithm creates a much longer average length than the other four methods. This is due to the fact that it does not solve the merge/split problems. Once the cells are merged, the trajectories of these cells are overlapping. It causes a longer track length of each cell trajectory which leads to a longer average track length. The original Viterbi algorithm exhibits the same problem. We applied

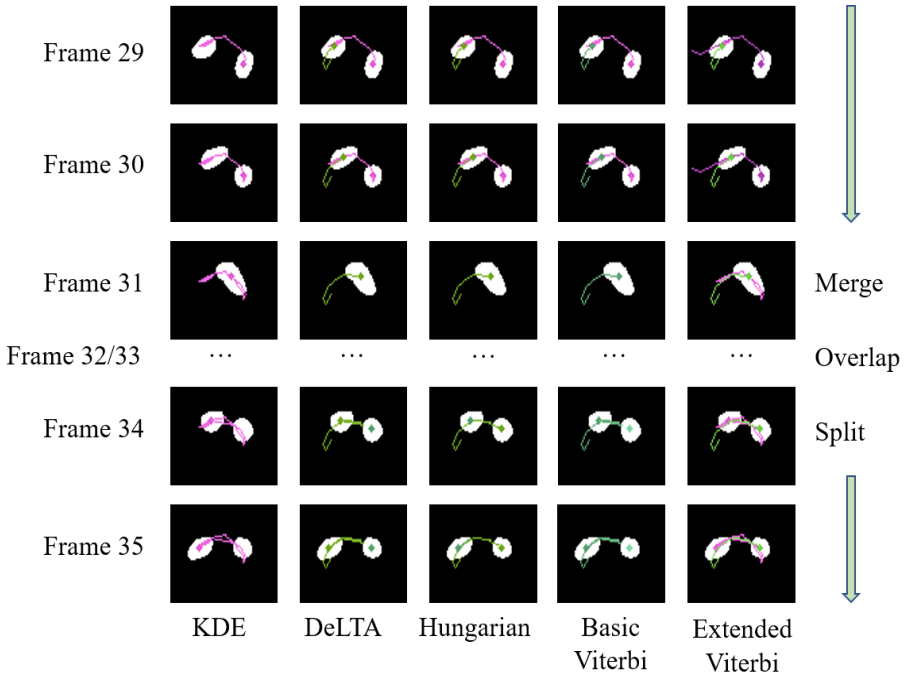


Fig. 4.12 A case of the comparison of how these methods solve the cell merge/split problem.

QR code 4.1 QR code 4.2 QR code 4.3

Fig. 4.13 Scan each QR code, different example time-lapse sequences of neutrophil tracking are given.

a post-processing for the basic Viterbi algorithm. The trajectory which has a lower score compared to the previous cell on the previous frame was truncated. The post-processing is helpful and makes the performance much higher. For our extended Viterbi algorithm, we designed different strategies to deal with the merge/split problems. The experiments show that the extended Viterbi algorithm further increases the performance.

A case of how the merge/split problem is solved by the five algorithms is depicted in Fig. 4.12. We present a series of consecutive frames between frames 29 to 35 of one particular time-lapse sequence. For the KDE algorithm, only the right cell, represented with a purple line, was tracked, the left cell was missed. For the DeLTA algorithm, the left cell, represented with a green line, was linked to the left one when the cells split in frame 34. This is not an expected direction. The right cell was treated as the start of a new cell trajectory. Same situation with the Basic Viterbi algorithm. For the Hungarian algorithm, the left cell, represented with a green line, was linked to the right cell after the split, which is an expected cell moving direction. For our extended Viterbi algorithm, the merge/split problem was solved. The left cell, represented with the green line is expected to move from left to right even after a collision, whereas the right cell, represented with the purple line, moves from right to left.

The trajectories of neutrophils are visualized subsequently and three examples are given in the following QR codes in Fig. 4.13. Scanning the QR codes and watching the time-lapse sequence.

4.3.4 Results on Other Datasets

In order to prove the robustness of our pipeline, two more experiments were conducted on datasets of two cell types. One type of data is downloaded from the cell tracking challenge [155]. It contains the stained nuclei of HL60 cells (Fluo_N2DH_SIM) and the HeLa cells stably expressing H2b-GFP (Fluo_N2DL_HeLa). The cells in this dataset move slowly and only express mitosis behaviour. The other type of data are MTLn3 plGFP cells [126][152]. This dataset consists of two groups. One is the experiment group, in which the cells move slowly without mitosis or any other behaviour. The other is the control group, in which the cells move faster but still without mitosis or any other behaviour.

Both of these cell data have simpler cell behaviour and slower movement compared to neutrophils. The length of the time-lapse sequence is much shorter than the 120-frame neutrophil sequences which facilitates the cell tracking task. The HL60 dataset has five time-lapse sequences in total with 65 frames per time-lapse sequence. The cell density is 10 to 40 per frame which is comparable to the neutrophil dataset. The MTLn3 dataset has two time-lapse sequences with 30 frames per sequence but with a very high cell density; about 80 to 100 cells per frame.

We retrained the U-Net 1 segmentation model and tracking model on the two new datasets, respectively. Subsequently, the same linkage methods were implemented. The performance results are shown in Fig. 4.14 and Fig. 4.15. Table 4.4 and Table 4.5 present the mean values (green triangles) corresponding to Fig. 4.14 and 4.15.

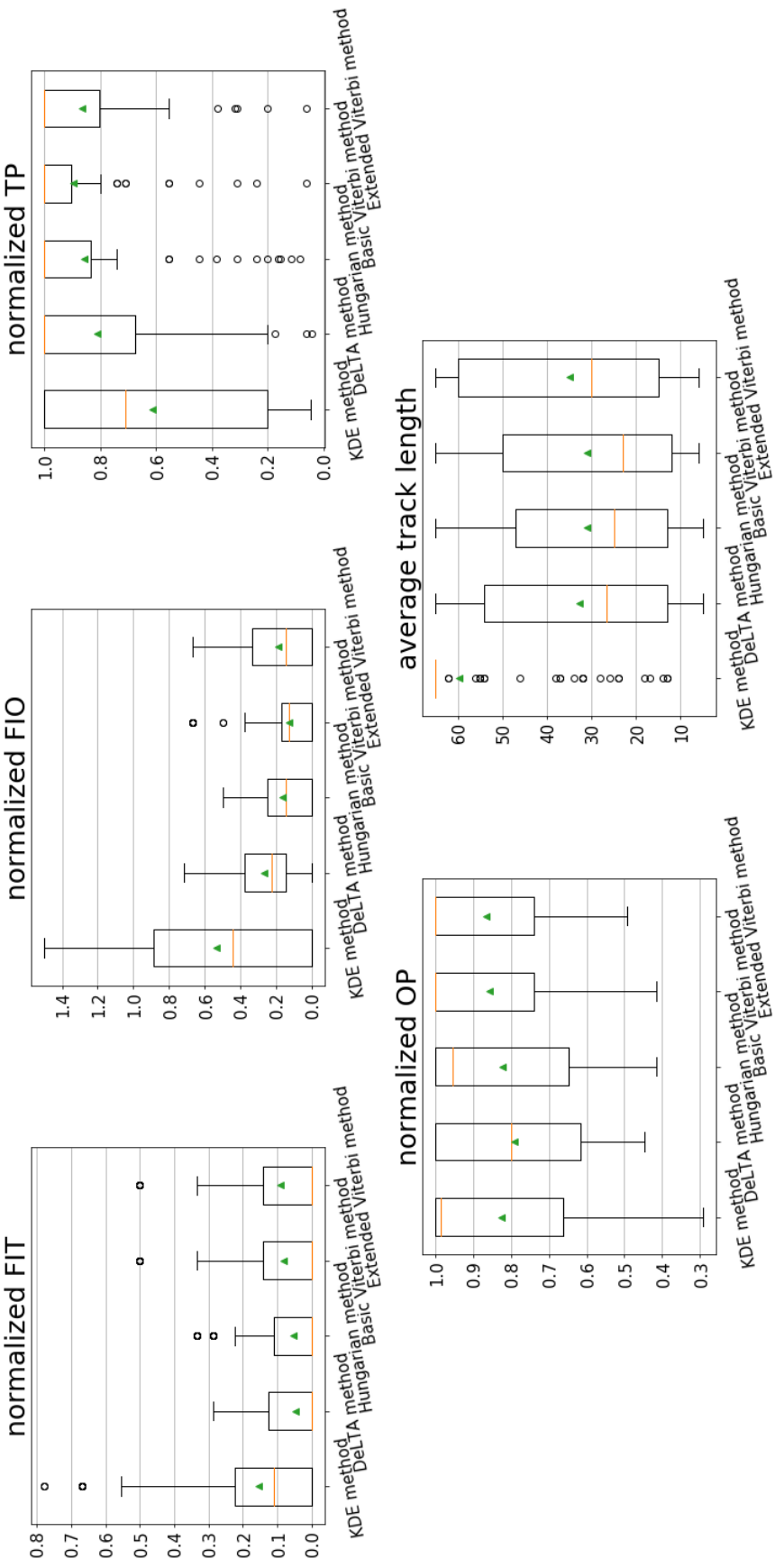


Fig. 4.14 The performance of different linkage methods with the selected evaluation metrics on the HL60/HeLa dataset. The metrics are: falsely identified track (*FIT*), falsely identified object (*FIO*), track purity (*TP*), object purity (*OP*) and average predicted track length. In the box plots, green triangle shows the mean values, the orange line is the median value, the box represents the first to third quartile of the data, the whiskers indicate the lower and upper extreme (1.5 times the interquartile range) and the black circles are the outliers.

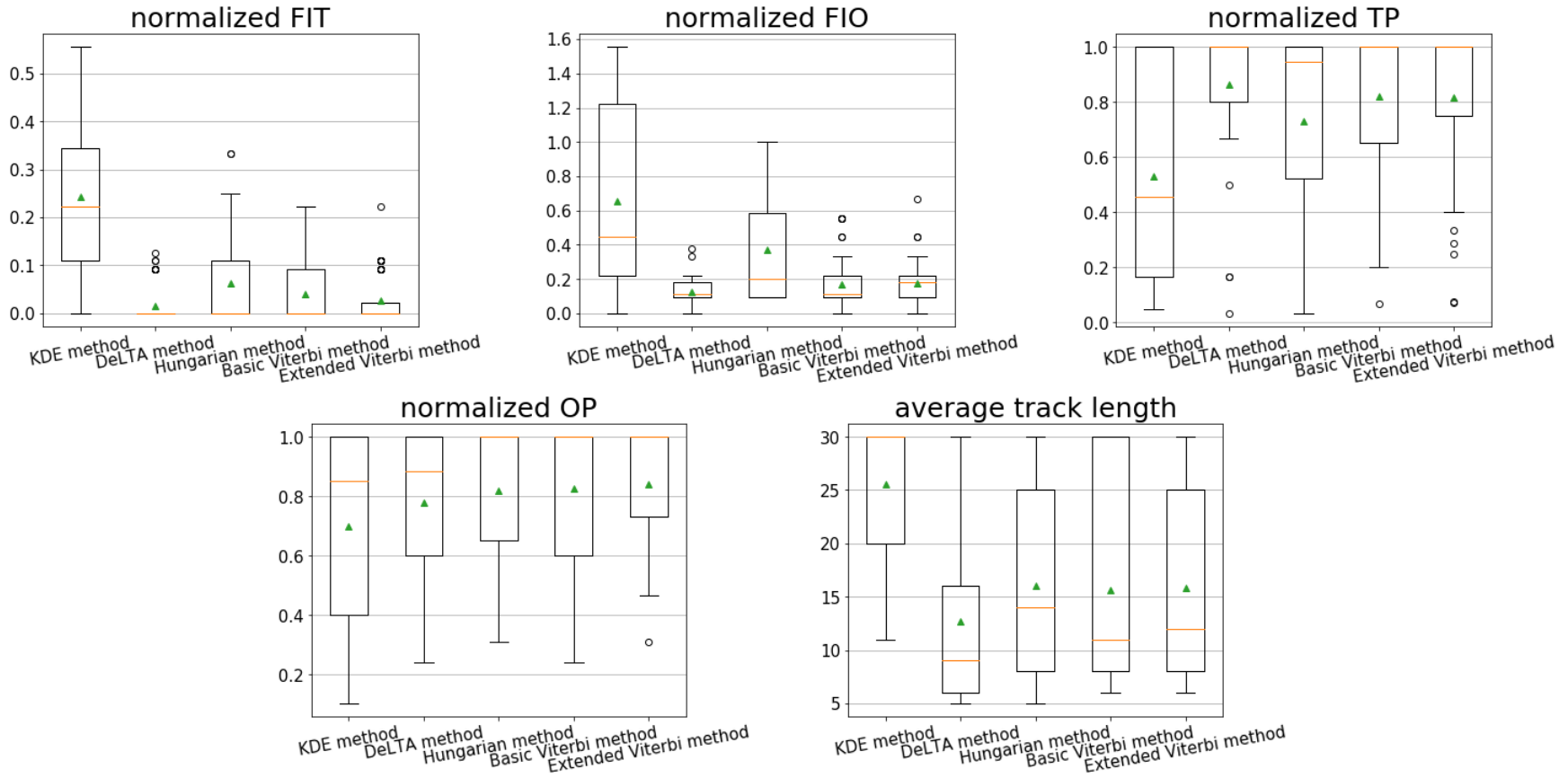


Fig. 4.15 The performance of different linkage methods with the selected evaluation metrics on the MTLn3 dataset. The metrics are: falsely identified track (*FIT*), falsely identified object (*FIO*), track purity (*TP*), object purity (*OP*) and average predicted track length. In the box plots, green triangle shows the mean values, the orange line is the median value, the box represents the first to third quartile of the data, the whiskers indicate the lower and upper extreme (1.5 times the interquartile range) and the black circles are the outliers.

Table 4.4 The performance of each tracking model on the HL60/HeLa dataset. The footnote is the rank for all compared methods of each evaluation metric.

Algorithms	**Normalized *FIT***	**Normalized *FIO***	**Normalized *TP***	**Normalized *OP***	**Average track length**
KDE [126]	0.154^{5}	0.537^{5}	0.612^{5}	0.825^{3}	59.801^{1}
DeLTA [127]	0.048^{1}	0.265^{4}	0.808^{4}	0.790^{5}	32.577^{3}
Hungarian [148]	0.053^{2}	0.165^{2}	0.857^{3}	0.823^{4}	30.867^{5}
Basic Viterbi	0.082^{3}	0.125^{1}	0.897^{1}	0.856^{2}	30.915^{4}
Extended Viterbi	0.091^{4}	0.186^{3}	0.864^{2}	0.866^{1}	34.866^{2}

Table 4.5 The performance of each tracking model on the MTLn3 dataset. The footnote is the rank for all compared methods of each evaluation metric.

Algorithms	**Normalized *FIT***	**Normalized *FIO***	**Normalized *TP***	**Normalized *OP***	**Average track length**
KDE [126]	0.243^{5}	0.658^{5}	0.532^{5}	0.700^{5}	25.509^{1}
DeLTA [127]	0.015^{1}	0.126^{1}	0.862^{1}	0.779^{4}	12.670^{5}
Hungarian [148]	0.062^{4}	0.373^{4}	0.732^{4}	0.819^{3}	16.010^{2}
Basic Viterbi	0.040^{3}	0.172^{2}	0.820^{2}	0.827^{2}	15.650^{4}
Extended Viterbi	0.028^{2}	0.175^{3}	0.818^{3}	0.841^{1}	15.796^{3}

For the HL60/HeLa dataset, the basic Viterbi algorithm realized the best score in *FIO*, *TP* and a second-best score of *OP* among these different methods. The extended Viterbi only achieved the best performance of *OP*. This experiment obviously demonstrated that our strategies work better on a cell dataset with more complex patterns such as neutrophils. For the cell dataset with simple patterns, the basic Viterbi is robust and outperforms the others. Compared with KDE method, deep learning methods still perform better than machine learning-based method which is consistent with the results on neutrophils.

For the MTLn3 dataset the performance of the two Viterbi-based algorithms is comparable with DeLTA and Hungarian methods. All of these four methods have their pros and cons, but still perform better than the KDE method. The extended Viterbi achieved the best score of *OP* whereas DeLTA obtained the best *FIT*, *FIO* and *TP*. The basic Viterbi yielded the second best *FIO*, *TP* and *OP*. The possible reasons that the Viterbi-based algorithms are not outperformed compared to the other two methods can be: (1) The experiment contains just two time-lapse sequences with 30 frames per sequence. This number of datasets is rather

limited for good training of a deep learning model. (2) Furthermore, the high cell density of the sequences easily causes linkage errors due to the global linkage way of the Viterbi algorithm. This is because the cells are very close to each other. The prediction would result in an overlap of two or more comparable adjacent cells causing the about same score in the matrix. It easily leads to an error trajectory in the global linkage process. Therefore, Viterbi-based algorithms are more robust on time-lapse sequences with a low or medium cell density instead of a very high density.

4.4 Conclusions

In this paper, we developed a pipeline incorporating cell segmentation, cell motion tracking between two frames, and trajectory linkage, to track the complex migration patterns of neutrophils in the time-lapse sequences. The pipeline integrates two U-Net models, to segment cells and learn the cell movement behaviour, respectively. The extended Viterbi algorithm can handle different patterns in cell migration such as appearing, disappearing, go in/out of the field of view, so as to more accurately link the cell trajectories. Compared with the existing state-of-art, our algorithm achieved superior performance and provides an effective approach to help understanding cell migration behaviour. In the next chapter, we will extend our work to 3D cell tracking using our neutrophil dataset. 3D data provide more spatial information in Z-axis and are expected to track the cell trajectories in a better way.

Chapter 5

A Feature Weighted Tracking Method for 3D Neutrophils in Time-lapse Microscopy

This chapter is based on the following publication:
C, Li., W, W.C. Yiu., W, Hu., L, Cao., F, J. Verbeek., A Feature Weighted Tracking Method for 3D Neutrophils in Time-lapse Microscopy. *2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. 2022, pp. 2196-2202.

Abstract:

Neutrophils are one of the crucial immune cells. It plays a key role in the immune system defending the invasion of harmful particles, such as viruses and bacteria. The analysis of the dynamic process of neutrophil migration helps biologists to understand underlying mechanisms of neutrophils in response to wounding. However, accurate neutrophil tracking is still a challenging task in 3D space due to the complexity of cell morphology and behavior. In this study, we improved the quality of raw data by denoising and linear interpolation. A 3D U-Net was trained and used to detect the location of each cell in the time-lapse sequence. Subsequently, a feature weighted 3D tracking method was proposed. The experimental results show that our method performs well compared to the existing tracking algorithm. Our pipeline is also much more reproducible than other state-of-arts.

5.1 Introduction

The migration of leukocytes is a process of identifying the acute inflammation which could harm organisms. The function of leukocytes is to fight against the invading pathogens so as to protect organisms. Neutrophils are the first cells to rapidly respond to the site of inflammation and act as the first line of defense [121]. Research on the migration of neutrophils in living organism (*in vivo*) helps biologists to understand biological mechanisms better. For this kind of research, zebrafish larva is a popular model for studying cell functions due to their small size and transparency characteristics for screening and imaging [156]. In recent years, the microscopy techniques have greatly advanced in the field of biology and medicine. Multi-functional microscopy allows scientists to visualize the dynamic process of living cells in spatial and temporal resolution, both in brightfield and fluorescence modes [157][158]. Despite these advanced conditions, the accurate measurements of neutrophils' migration are still challenging, not only in 2D + T (x,y,t) space but also in 3D + T (x,y,z,t) space.

In general, cell tracking tasks are divided into two steps: cell segmentation to locate the position of each cell, and cell tracking to associate the trajectory based on cell similarities [124][127][159][160][161].

Segmentation algorithms have gone through a long developed process from threshold segmentation (e.g. Otsu [132]), watershed-based segmentation [152][141], to the deep learning segmentation methods (e.g. U-Net [134][162]). 3D U-Net is the popular segmentation model in the biomedical field. 3D U-Net segmentation model was first proposed in [163]. It was proved to achieve a good performance on kidney embryo segmentation. Although it is a deep learning model, it only requires small amount of annotated data for training. 3D U-Net reduces laborsome annotation work. However, unlike an individual kidney embryo, cells are often collided with each other. The watershed method was successfully used to help separate the touched cardiac cells in zebrafish as a post-processing [164].

Once the segmentation is performed, the next step is to associate the cell tracks frame by frame based on the similarities. Rule-based methods are studied the most in recent years[165][166]. A graph-based tracking algorithm was proposed in [165] which utilizes relative cell location information to associate the cell tracks. It achieved comparable performance to state-of-art methods in Cell Tracking Challenge 2019 and 2020. Rule-based methods are easier to tailor the features for specific datasets. Deep learning tracking methods extract features automatically. [164] proposed a deep learning-based software pipeline named 3DeeCellTracker for tracking. The datasets they used are cardiac cells in zebrafish and neurons in worm' brain. The characteristics of these datasets are that the cells move in the same pattern which can be simulated using random affine transformations. A simulated dataset was created by random affine transformations and used to train the deep learning

model. For neutrophils, the cell movement is much more complex and irregular. It is difficult to find a pattern in it. In [157], a 3D cell association learning network was designed and used for neutrophils tracking problem. It achieved a significant performance and solved the cell collision problem well. However, the deep learning models require a huge number of annotated data for training. The ground truth data were annotated by labeling each cell position and associating the cells over time manually. It is time-consuming and for other researchers, it is difficult to reproduce the process.

In our study, a 3D U-Net segmentation model and a rule-based tracking method were proposed that tailored to our captured neutrophil dataset. At first, the image pre-processing, containing enhancement, denoising, and smoothing, were used to improve the image quality. In this dataset, the size of 3D raw data is (512, 512, 8). Eight layers were scanned on Z-axis which makes the 3D data too thin to observe the movement on Z-axis. We did the linear interpolation to enlarge the dimension to (512, 512, 29). Secondly, a 3D U-Net segmentation model was trained and a watershed algorithm was applied to locate the positions of cells. Thirdly, we designed the features, incorporating cell distance, cell direction, and average cell movement. The final similarity scores calculated by weighting those features were used to improve the cell tracking. Subsequently, the Hungarian algorithm was used to associate the cell tracks at the end[1]. The results show that our feature-based method outperformed the graph-based method in [165]. In addition, Hungarian linkage performed better than a straightforward linkage method in [127].

5.2 Data Collection and Pre-processing

5.2.1 Data Capturing

The time-lapse dataset of neutrophil was captured by the experts. All the experiments were done with zebrafish and followed the international guidelines specified by the EU Animal Protection Directive 2010/63/EU. All the zebrafish were 3 days post fertilization (dpf) and the tail wounding was conducted with the protocol provided in [121]. The wounded tail area of specific samples was imaged using a Leica TCS SP8 confocal microscope (Leica Microsystems) with a 10× objective (N.A. 0.40). Neutrophils, localized within an area of 200 μm from the wounding edge toward the body trunk, shown in Fig. 5.1, were counted as recruited cells. Under the microscopy, the zebrafish tails were scanned from top to bottom. An 8-layer 3D stack at each time point was captured. The size of each stacked image is 512 × 512 × 8. The layer interval is 5∼6μm, along with the thickness of tail is 35∼42μm. For

[1] https://surfdrive.surf.nl/files/index.php/s/IQ1wEMn4lA7bB7x/download

each sample, a 2-hour time-lapse sequence was captured, the time interval is one minute, along with the 120 frames for each sequence. Then we obtained the time-lapse 3D data with the size of (512, 512, 8, 120), which corresponded to the axis (x, y, z, t). Each time-lapse sequence contains almost 10 to 40 cells. We captured 9 time-lapse sequences in total.

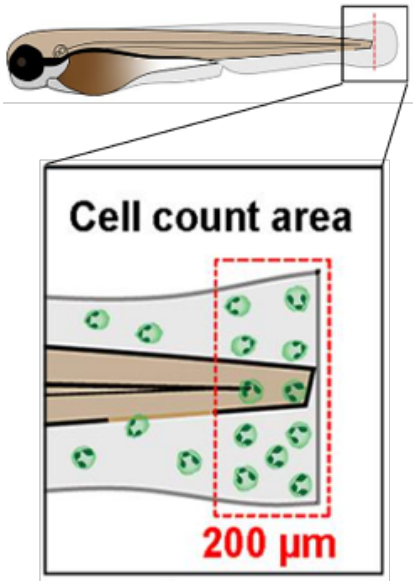


Fig. 5.1 The area of neutrophils recruited. The red line is the wounding edge in zebrafish tail. The red dashed box shows the area where neutrophils were counted as recruited neutrophils.

5.2.2 Data Pre-processing and Linear Interpolation

During the procedure of data capturing, there could be noises on the background that reduce the quality of images. Insufficient image brightness makes the cells in a lower intensity and not clearly visible as well. Therefore, several methods were applied to improve image quality. At first, we enhanced the image contrast to highlight the cell. However, noises at the background were enhanced as well. Thus, a median filter was used to denoise. Subsequently, we chose a Gaussian blur to smooth the cell surface. Fig. 5.2 shows the procedure of image pre-processing.

The 8 layers on Z-axis result in a very thin 3D bounding box. It is very difficult to observe the cell movement along the Z-axis. Due to the experimental design, the expert had to scan 3 channels (one brightfield channel, and two fluorescence channels) within one minute of the time interval. There was not enough time to scan more layers on the Z-axis. In order to observe the Z-axis movement of cells more clearly, we used linear interpolation twice [167] to increase the layers on the Z-axis from 8 to 15 for the first time, then to 29 for the second

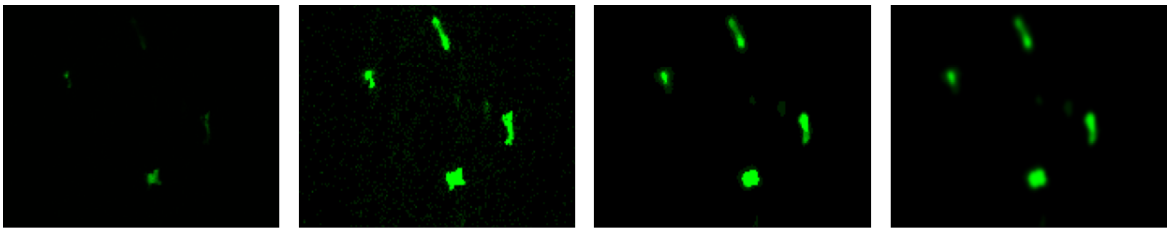


Fig. 5.2 Image preprocessing. (a) A region of raw data. (b) Image contrast enhancement. (c) Image denoising with median filter. (d) Image smooth with Gaussian blur.

time. Each interpolated layer was calculated as the average intensity of each two adjacent layers. We did not interpolate more times because the more layers the data have, the bigger the memory size, which is a burden for computing. We have kept a balance between data quality and data size.

5.3 3D U-Net Segmentation

The segmentation is a step to locate the position of each cell before tracking procedure. The 3D U-Net structure [164] was used in our study. Like the standard U-Net, it has a contracting path and an expansive path, which is a symmetric U-shaped structure. 3D U-Net is powerful for training a segmentation model with very little annotated images [163]. To prepare the annotated data, an open-source tool named Segmentor [168] was used for its efficiency and user-friendly interface. We annotated two 3D images with size of $512\times512\times8$, one for training and the other one for validation of the model. The same two interpolated 3D images with size of $512\times512\times29$ was annotated as well. We trained two segmentation models for different size images on a dedicated server equipped with two NVidia GeForce GTX 2070 with 8 GB GPUs using Linux Ubuntu operating system. In order to reduce the memory usage during training, we divided the large images ($512\times512\times8$, $512\times512\times29$) into small ones ($160\times160\times8$, $160\times160\times16$) as the input to train the models and then combine the sub-images together to form a whole images [164]. The dividing process help to enlarge the number of training images. Data augmentation was used to increase the training data as well. The output of the segmentation model is the probability of whether each voxel belongs to the cell region or not. The watershed segmentation in [164] was applied to separate each individual cell. It was used twice in the x-y plane and z-dimension, respectively, due to the data have differed resolution in X-Y axis and Z-axis.

5.4 Feature Weighted Tracking

Feature weighted algorithm contains several representative features generated from the properties of cell movements. They are: cell distance, cell direction, average cell movement distance. The similarity scores of each pair of cells on every two adjacent frames were calculated by weighted of those features. Based on the similarity scores, the Hungarian algorithm [148] was used to associate the related cells frame by frame.

5.4.1 Cell Distance

Cell distance D is defined as the distance from the center point $(x_{i,t}, y_{i,t}, z_{i,t})$ of a cell i on the frame t to the center point $(x_{j,t+1}, y_{j,t+1}, z_{j,t+1})$ of a candidate cell j on frame $t+1$. The distance is calculated based on Euclidean distance in (5.1).

$$D = \sqrt{(x_{i,t} - x_{j,t+1})^2 + (y_{i,t} - y_{j,t+1})^2 + (z_{i,t} - z_{j,t+1})^2} \tag{5.1}$$

The distance score was normalized by (5.2). A threshold T was given. It represents the maximum distance that the cell could move between two frames. If the distance is larger than T, the distance score is 0.

$$Distance\ Score = \max(0, (T - D)/T) \tag{5.2}$$

5.4.2 Cell Movement Direction

The direction of cell movement is an important factor that affects its next destination. Fig. 5.3 is a ground truth sample which shows a clue that the cell movement tends to stick to similar direction in a collision-split event.

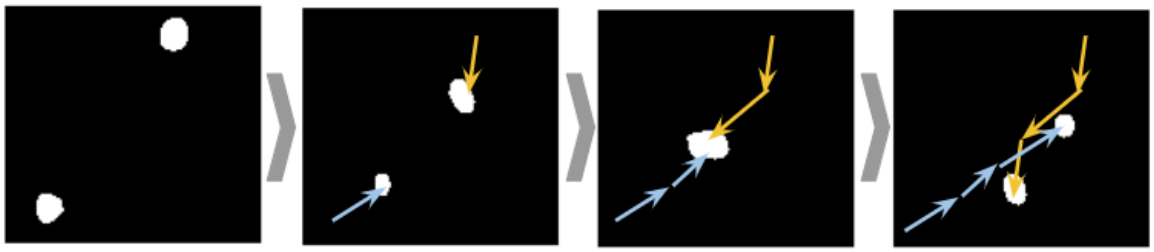


Fig. 5.3 Two cells that are moving from different directions, collide and then split according to their trend. Only one layer in Z-axis is shown.

The directional variation is more reliable when several prior frames are referenced. It can produce a sufficient direction variation of the cell movement. The number of the former

frames to be included is a hyperparameter which can be tuned. To calculate the direction variation, we first have to calculate the directional vector. Suppose there is a cell A on frame t, Fig. 5.4 displays how the directional vector is calculated by five frames before frame t in a 3D space with the coordinate of the Z-axis is zero.

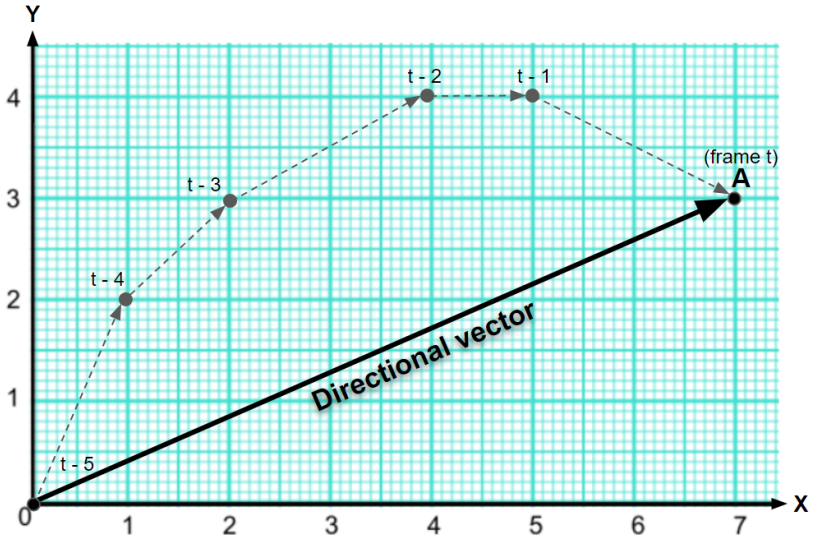


Fig. 5.4 Directional vector calculation from the last five frames of a cell track.

The start location of the cell on frame $t-5$ will be set to (0,0,0) in 3D space. The relative movement for each frame after is calculated and added. In this case, the relative coordinates of the previous five frames are (1,2,0)→(1,1,0)→(2,1,0)→(1,0,0)→(2,-1,0). The result of the directional vector is (7,3,0) computed by adding all the five relative vectors.

Next, the vectors from cell A to the surrounding candidate cells on frame $t+1$ were calculated. The angle difference between the directional vector a and each candidate vector b is calculated using the dot product equation in (5.3). An inverse cosine is applied on both sides to derive the degree α in (5.4).

$$a \cdot b = |a| \times |b| \times \cos\alpha \tag{5.3}$$

$$\alpha = \arccos\left(a \cdot b \;/\; (|a| \times |b|)\right) \tag{5.4}$$

The candidate cell that has a smaller degree to the directional vector will receive a higher score. The score will range between 0 (180°) to 1 (0°) in (5.5).

$$\textit{Degree Score} = (\cos\alpha + 1) \times 0.5 \tag{5.5}$$

5.4.3 Average Cell Movement Distance

Average cell movement distance is also an important factor to distinguish inactive and active cells. Cells that are inactive tend to linger around the proximate region, while cells that are more active tend to travel at a similar distance over each frame. Fig. 5.5 is a cell movement track extracted from a ground truth example which illustrates this scenario.

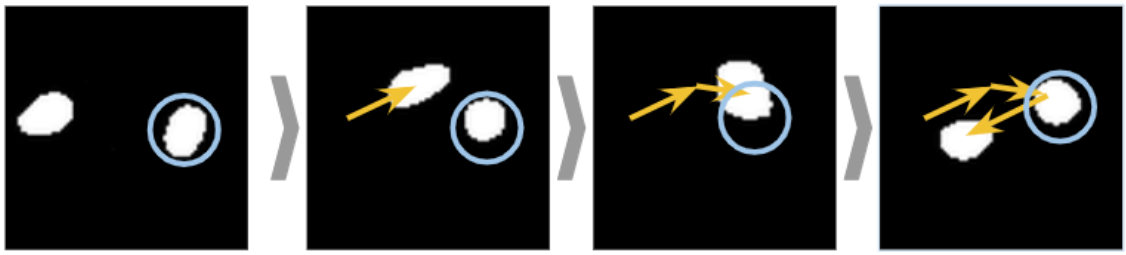


Fig. 5.5 Two cell tracks with different movement rates collide in frame 3 and splitted right after. The cell on the right remained inactive throughout the frames.

The cell track labeled with yellow arrows has a distant moving distance, whereas the cell track circled with light blue lingered in the nearby region. It can be seen that both cells maintained a similar movement distance over the 4 frames. The average movement is derived from (5.6). D_i is the distance of each pair of cells. n is the frame in which the cell moves.

$$Average\ Movement = \frac{1}{n}\sum_{i=1}^{n} D_i \tag{5.6}$$

The score is normalized to unity with (5.7). T is the same setting in (5.2).

$$Average\ Movement\ Score = Average\ Movement/T \tag{5.7}$$

5.4.4 Similarity Scores

The similarity scores are defined as the sum of all weighted feature scores in (5.8). The *feature_scores* is [*Distance Score*, *Degree Score*, *Average Movement Score*]. The influence of each feature can be adjusted through a 3 by 1 weight matrix. The weight matrix consists of 3 positive rational numbers, e.g., weight = [0.4, 0.3, 0.3].

$$Similarity\ Score = \sum_{i=1}^{n} weight(i) \times feature\_scores(i) \tag{5.8}$$

5.4.5 Hungarian Association

We derived all 119 of the similarity score matrices between each adjacent frame over the 120-frame time-lapse sequence. The scores in matrices were used as input for the Hungarian algorithm for constructing the cell associations. Hungarian is one of the linear programming algorithms that can be used to solve a profit maximization assignment problem [148] [169]. Fig. 5.6 (a) shows one example of similarity matrix between frame t and frame $t+1$. The number of rows and columns represents the cell numbers on frame t and frame $t+1$, respectively. The Hungarian algorithm requires a square matrix and returns an optimal path over the matrix. Hence, a dummy row with zero values was added as shown in Fig. 5.6 (b). If the number of rows is more than the number of columns, a dummy column is added.

The values in the matrix represent the similarity between all combinations of cell candidates on two adjacent frames. Therefore, we should maximize the sum of profits of all candidates using the Hungarian algorithm. In Fig. 5.6 (c), the nodes on the best path were underlined. The maximum sum of profit in the matrix is 0.9+0.7+0.8+0.7+0=3.1. The index pairs of each optimal position were returned as a list, such as [(0,0), (1,2), (2,1), (3,3), (4,4)]. Similarly, a new index list was obtained between frame $t+1$ and frame $t+2$. The cell tracks were formed by linking matched cell index frame by frame.

In addition, the candidate cells on frame $t+1$ that are associated by a dummy cell node on frame t are regarded as newly appeared cells. The candidates on frame t that are associated with dummy candidates on frame $t+1$ are regarded as disappeared cells.

Frame t + 1

Frame t	0	1	2	3	4
0	0.9	0	0	0	0
1	0	0.1	0.7	0	0
2	0	0.8	0.2	0	0
3	0	0	0	0.7	0.1

(a)

Frame t + 1

Frame t	0	1	2	3	4
0	0.9	0	0	0	0
1	0	0.1	0.7	0	0
2	0	0.8	0.2	0	0
3	0	0	0	0.7	0.1
4	0	0	0	0	0

(b)

Frame t + 1

Frame t	0	1	2	3	4
0	<u>0.9</u>	0	0	0	0
1	0	0.1	<u>0.7</u>	0	0
2	0	<u>0.8</u>	0.2	0	0
3	0	0	0	<u>0.7</u>	0.1
4	0	0	0	0	<u>0</u>

(c)

Fig. 5.6 The process of the Hungarian algorithm. (a) Raw matrix. (b) A dummy row was added. (c) The optimal path between two frames was found.

In this way, 119 index lists were obtained from these matrices. The final cell trajectories were associated based on the index lists.

5.5 Experimental Results

In order to evaluate the performance of our proposed method, the annotated ground truth and evaluation criteria are required.

5.5.1 Ground Truth Annotation

It is challenging and time-consuming to annotate cell trajectories in 3D + T space. In this study, the annotation was done in a combination way. At first, the 3D data were projected in 2D space. In a 2D time-lapse sequence, we recorded the cell center coordinates of the tracks over the cell movement. This procedure was done using 'Manual Tracking' plugin in ImageJ [69][170]. The next step is to map the 2D cell center coordinates on each track to the 3D + T sequence. Then the center point on the Z-axis of each cell was located and added manually. We annotated 20 ground truth trajectories over the raw dataset with different sizes of (512, 512, 8, 120) and (512, 512, 29, 120), respectively.

5.5.2 Evaluation Criteria

Four measures to evaluate how well a tracker identified objects, proposed by [154], were selected. They are:

Falsely Identified Tracker (*FIT*): a measure of the degree to which the ground truth objects (*GT*) are tracked by the incorrectly predicted tracks (ε).

Falsely Identified Object (*FIO*): a measure of how often the predicted tracks (ε) is tracking a different cell than the *GT* it was matched to.

Track purity (*TP*): a measure of the degree to which the predicted tracks (ε) follow *GT*. It is the ratio of frames that ε correctly identifies *GT* to the total number of frames ε exists.

Object purity (*OP*): is a measure of the degree to which the *GT* are followed by predicted tracks (ε). It is the ratio of frames that *GT* is correctly identified by ε to the total number of frames *GT* exists.

The average length of tracks was computed as well.

Average Track Length: the ratio of the total length of all predicted tracks to the number of predicted tracks.

5.5.3 Results

We did the experiments on both datasets with different sizes: (512, 512, 8, 120) and (512, 512, 29, 120). A graph-based tracking method proposed in [165] was compared. In [165], a

feature vector, consisting of cell volume, the total number of cell neighbors, and the average distance from all other cells, were obtained. Subsequently, the similarity between each pair of cells was calculated based on the feature vector.

In our algorithm, the parameters were configured as follows. The threshold T in (5.2)(5.7) was set to 35. It means if the distance between two cells is more than 35 pixels, they would not be considered the same cell. Thus, the *Distance Score* equals to 0. It was decided by observing the possible maximum movement distance. Once *Distance Score* equals 0, the other two scores were set to 0 as well. To compute cell movement direction, the number of prior frames was set to 6. To compute the average cell movement distance, the number of consecutive frames was set to 6. To calculate the similarity scores, the weight was set to [*Distance Score*, *Degree Score*, *Average Movement Score*]=[0.4, 0.3, 0.3]. All parameters were selected by tuning manually.

Based on the features we extracted, a Hungarian algorithm was applied to associate the cells frame by frame. In addition, a straightforward associated method in [127] was compared. The cell pairs with a high score in the matrix were linked, and the relation over frames was not taken into account. If there is a score in conflict between cells (e.g. two same scores in a row or a column), this method only chooses the first cell as the candidate cell rather than an optimal choice.

The performance comparison on the raw dataset with size of (512, 512, 8, 120) is shown in Fig. 5.7. In order to quantify the values clearly, Table 5.1 shows the mean values which correspond to Fig. 5.7. From Table 5.1, our algorithm reaches 0.145 of *FIO*, 0.845 of *TP*, 0.386 of *OP* and 20.238 of Average Track Length, respectively. It performs better than [165]. Only *FIT* is lower. Compared to [127], our algorithm also increased the performance by 0.050, 0.056, 0.039, and 3.654 of *FIO*, *TP*, *OP*, and Average Track Length, respectively.

Table 5.1 The Mean Value of Each Evaluation for Two Methods Comparison on the raw dataset with size of (512, 512, 8, 120).

Algorithms	***FIT***	***FIO***	***TP***	***OP***	**Average Track Length**	**Time** (s)
Algorithm in ref [165]	0.015	0.273	0.798	0.344	19.620	614.49
Algorithm in ref [127]	0.025	0.195	0.789	0.347	16.584	389.39
Our algorithm	0.045	0.145	0.845	0.386	20.238	358.96

Fig. 5.8 shows the performance of the three methods on the interpolated dataset. From the corresponded Table 5.2, we derived that both the false rates of our method are lower

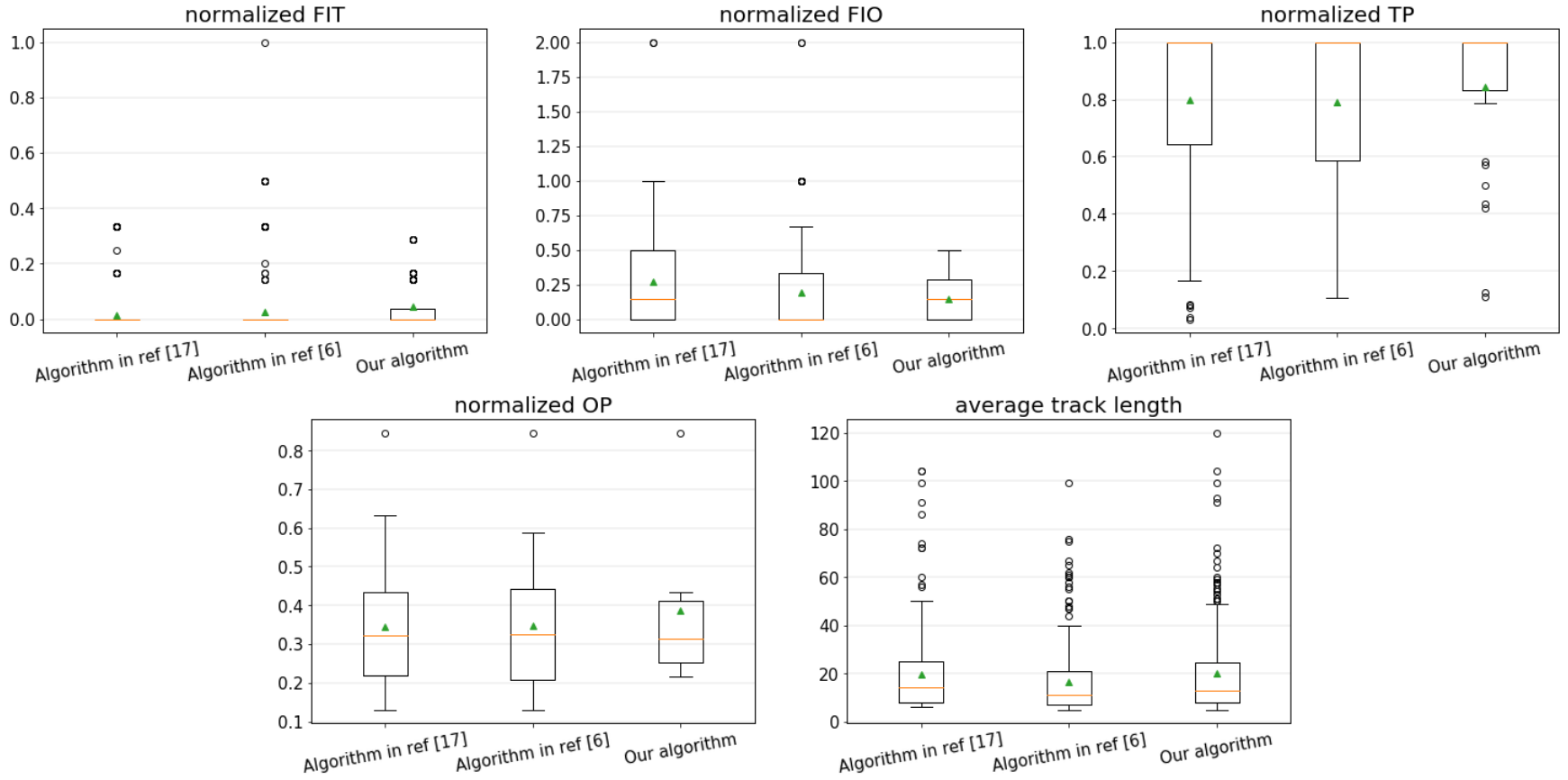


Fig. 5.7 The performance comparison on raw dataset with size of (512, 512, 8, 120). In the boxplots, green triangle shows the mean values, the orange line is the median value, the box represents the first to third quartile of the data, the whiskers show the lower and upper extreme (1.5 times the interquartile range) and the black circles are the outliers.

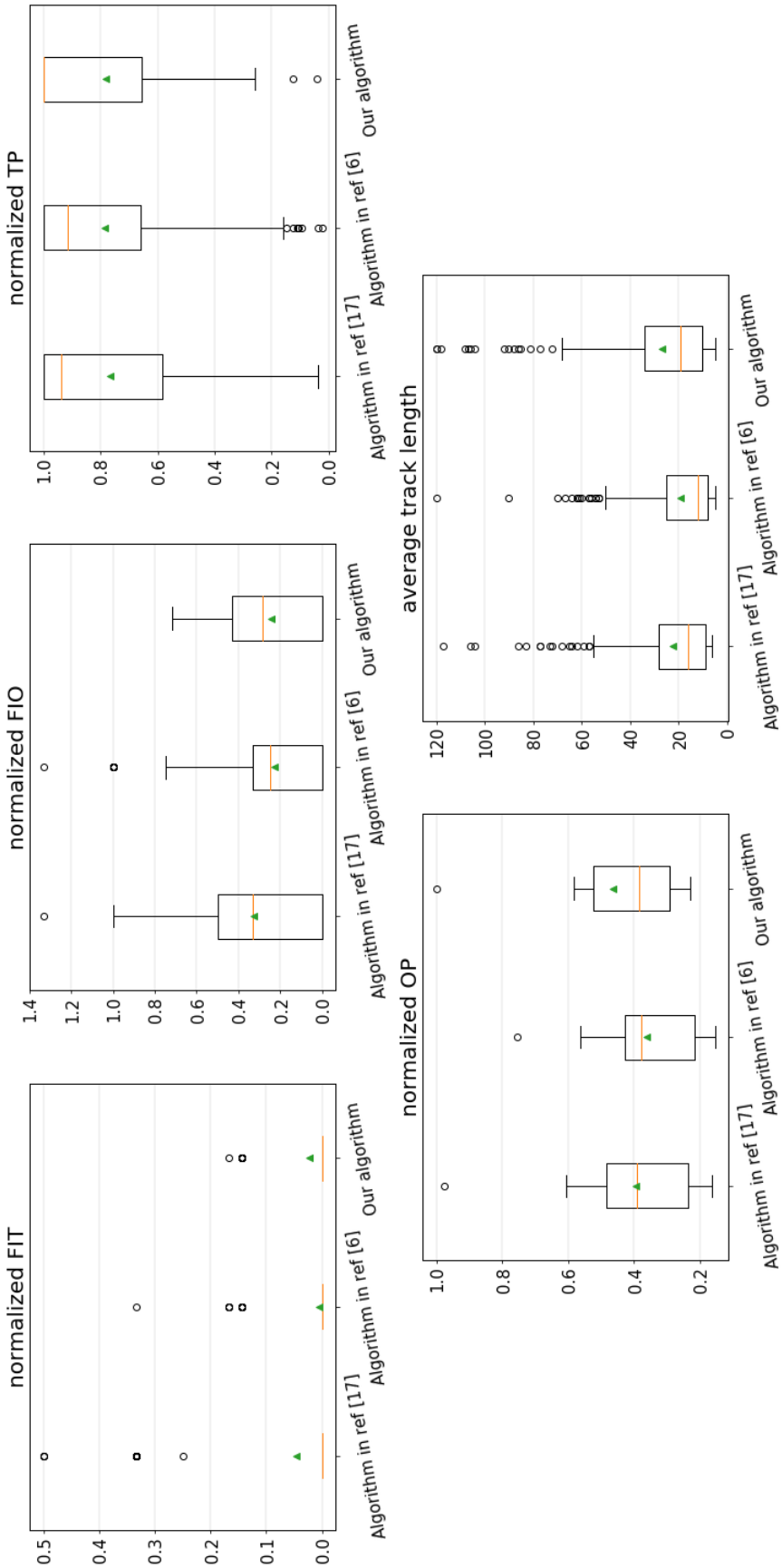


Fig. 5.8 The performance comparison on raw dataset with size of (512, 512, 29, 120). In the boxplots, green triangle shows the mean values, the orange line is the median value, the box represents the first to third quartile of the data, the whiskers show the lower and upper extreme (1.5 times the interquartile range) and the black circles are the outliers.

than [165]. *FIT* and *FIO* were decreased to 0.023 and 0.241, respectively. Furthermore, both purity rates (*TP* and *OP*) are higher than [165], along with 0.013, 0.071 were improved. Although [127] achieved the lowest false rate of 0.005 for *FIT* and 0.228 for *FIO*, our algorithm yielded a higher performance with *TP*, *OP* and average track length.

Table 5.2 The Mean Value of Each Evaluation for Two Methods Comparison on the interpolated dataset with size of (512, 512, 29, 120).

Algorithms	***FIT***	***FIO***	***TP***	***OP***	**Average Track Length**	**Time** (s)
Algorithm in ref [165]	0.046	0.326	0.767	0.393	22.490	2531.67
Algorithm in ref [127]	0.005	0.228	0.783	0.360	19.131	642.85
Our algorithm	0.023	0.241	0.780	0.464	26.633	619.64

The Average Track Length of the interpolated dataset is not only 4.143 longer than [165] but is also 6.395 longer than the result derived from the raw dataset. The reason is that the interpolation improves the representation of each 3D cell. Therefore, fewer cells were missed by segmentation procedure.

The computing costs of the two datasets were shown in Table 5.1 and 5.2. Our algorithm has lower computing time compared with a graph-based method in [165]. With the increase in the data size, graph-based method is much more computationally expensive while our feature-based algorithm works more efficient. The Hungarian method has a comparable computing costs compared with the simple linkage strategy in [127].

The experiments show that our method improved the performance of tracking on our neutrophil datasets compared to the available tracking algorithm in [165] and the simplistic associated method in [127].

5.6 Conclusions

This paper aims to develop a method to track neutrophils in 3D + T space accurately, so as to help biology experts to understand its movement behavior better. We started from data capturing and improved the low-quality raw data using image enhancement, denoising, and smoothing methods. A linear interpolation method was applied to increase the dimension along Z-axis to increase the sampling resolution. 3D U-Net segmentation models were trained and, together with a watershed algorithm, the cells' locations were detected in 3D space. A novel feature weighted tracking method was developed which is tailored to the

specific characteristics of our neutrophil dataset. The cell distance, cell movement direction, and average cell movement distance were three features selected according to the analysis of neutrophils' movement behavior. Based on these features, weighted similarity score matrices were generated and the Hungarian algorithm was used to associate cells frame by frame. The experimental results proved that our method outperformed the available state-of-art algorithm. The rule-based tracking method was selected because deep learning models require large ground truth data to fit in. Annotation in 3D + T space is still a challenge and there is not a user-friendly way to help labelling yet. Deep learning tracking models were proved to outperform the rule-based methods in state-of-arts [157][164]. However, it is difficult for other researchers to reproduce the algorithm if their work is not open-source. At present, the rule-based tracking methods are much more straightforward to follow. In the future, we would focus on improving annotation methods in 3D + T space as well as implementing tracking tasks with deep learning models.

Acknowledgment

This work was supported by the Chinese Scholarship Council through Leiden University.

Chapter 6

Conclusions and Discussion

In this thesis, we have addressed five research questions regarding two applications: pollen classification and neutrophil tracking in the biomedical domain. The themes investigated include image classification, image segmentation, and object tracking. In this chapter, we summarize the main findings from our research. Additionally, we discuss the limitations of our methods and the possible solutions to address them. Lastly, we point out the trends for future work.

6.1 Main Findings

In each research chapter, we have proposed new strategies/approaches to answer the corresponding research questions. Here, we summarize these approaches and present the main findings inspired by experimental results.

(1) **Chapter 2** answers **RQ 1: Can the existing deep learning-based classification models work with images from morphologically similar pollen grains of related species and what is the performance of the different models? Chapter 2** reports on the first time that CNN models are applied to classify morphologically similar unacetolyzed pollen grains of two common genera and a species in the Urticaceae family. In this Chapter, the captured raw pollen data is a 3D stack that has 20 through-focus layers. In order to gain good performance from classification models, therefore, we select three projections of the raw 3D pollen images and treat them as 2D data as the input of the CNNs. In this manner, projections incorporate as many representative features as possible. Consequently, we compare the classification performances of three popular CNNs including VGG16, light-weighted MobileNet V1, and MobileNet V2. CNNs extract features from images automatically via the convolutional layer and these feature vectors are fed into fully connected layers for classification. We,

respectively, train the three mentioned models with random initialization parameters, i.e. model from scratch, as well as pre-trained parameters, i.e. pre-trained model, also known as transfer learning technique. Data augmentation and hard voting techniques are adopted so as to further improve the performance. We observe that for the pre-trained model, VGG16, MobileNet V1, and V2 achieve comparable results. For the model from scratch, VGG16 has around 10% accuracy higher than the MobileNet. It proves that VGG16 is more robust than the light-weighted MobileNet and the transfer learning technique has a more significant effect on the MobileNet. In addition, the newly trained VGG16 model based on the new aerobiological samples is further used in a case study, which proves the deep learning-based VGG16 works successfully on our complex pollen classification task.

(2) **Chapter 3** is extended work from **Chapter 2**. We intend to integrate more models not only from deep learning but also from traditional machine learning approaches and explore the deeper insights further from those approaches.

In **Chapter 3**, we first involve three extra deep learning models: a shallow AlexNet, a comparable VGG19, and a deeper ResNet50. Among the six deep learning-based models, we find that the classification performance does increase from shallow AlexNet to deeper ResNet50. The VGG16, VGG19, MobileNet V1, and V2 reach comparable performance within the performance range of AlexNet and ResNet50. Secondly, we construct traditional machine learning-based models. We explore and extract the handcrafted features from our pollen images and use different classifiers such as SVM, RF, MLP, Adaboost to classify pollen categories. In this process, feature selection and reduction methods are applied to remove the irrelevant and redundant features. The flat models in which only one classifier is used to classify all classes are constructed. Compared to flat models, a hierarchical strategy merges the classes which are more similar into subgroups and classifies these subgroups separately with selected classifiers at different stages. The experimental results imply the hierarchical-structured classification model outperforms the flat-structured model.

The research question **RQ 2: How does the performance of the traditional machine learning-based classification models compare to that of deep learning-based models?** is answered after exploring the performance of deep learning-based models and traditional machine learning-based models. We observe that all the investigated flat classification models yield lower performance compare to the deep learning-based models, even to the shallow AlexNet. The hierarchical model is able to accomplish a comparable performance as AlexNet after fine-tuning all hyper-parameters to optimal settings. However, it could not keep pace with other deeper deep learning-based models investigated. The possible reason is the handcrafted features are designed from expertise and experience instead of learning from the ground truth automatically. There is a huge chance that the extracted features cannot

thoroughly cover the representations of pollen images. Therefore, we conclude that deep learning-based models have powerful capacities and better performance on similar pollen classification tasks.

In addition, to answer **RQ 3: To what extent is it possible that both the traditional machine learning-based and the deep learning-based classification models perform well on a relatively small amount of data?**, we conduct experiments on two small-size datasets, which have around 500/1000- images. All the aforementioned classification models are implemented and compared on the two datasets, respectively. The experimental results show: with the decreasing number of images in the dataset, the classification accuracy decreases accordingly. However, the trend is the same in that deep learning-based models outperform all flat models. The hierarchical models can achieve comparable performance with a shallow AlexNet, but not with other deeper CNNs. Moreover, we observe the decreasing extent of different models. For the 1000- image dataset, the accuracy is reduced by 2-5% varies from different models, while a 4-8% decrease on the 500- image dataset. The decreasing accuracy is fluctuating more or less but follows the same trend.

According to the analysis above, we are able to guarantee the power of the deep learning-based models for handling our pollen classification task. To explore more details of deep learning models, an ablation study is conducted. Peeling the whole model from transfer learning, data augmentation, and hard voting techniques step by step, we discover that the transfer learning technique has a significant impact on the improvement of the classification accuracy with 12-13%. While data augmentation and hard voting improve the performance of 1-2% and 1-3% further, respectively. The ablation study reveals the reason why the deep learning model works well for pollen classification.

(3) In **Chapter 4**, we answer the fourth research question **RQ 4: To what extent is it possible to develop an automated algorithm that provide accurate support in the tracking of neutrophils from time-lapse sequences in the 2D spatial domain?** In order to address this question, we first project the 3D images containing neutrophils to 2D images for all time points of the time-lapse recording. Subsequently, we load the images in our processing pipeline to conduct cell tracking in 2D + T space. We divide a tracking task into three parts: cell segmentation, cell motion tracking, and trajectory linkage. We compare several 2D segmentation approaches including both rule-based (Watershed) and deep learning-based (U-Net-based) methods. We aim to find a segmentation model that can maximally benefit the cell tracking part. The ground truth data for segmentation are prepared manually. We provide a ground truth dataset of 240 images. One more U-Net model is applied to learn the cell movement from the ground truth annotation. This ground truth dataset with hundreds of labeled trajectories is annotated by biological experts by "Manual tracking". Last but not

least, we propose an extended Viterbi algorithm to link the trajectories. We add different heuristics to the basic Viterbi algorithm in order to deal with complex cell behavior such as cell merging and splitting. We find that our extended Viterbi algorithm achieves superior performance compared to other straightforward linkage methods.

(4) In **Chapter 5**, we expand our cell tracking study to 3D + T space so as to answer question **RQ 5: To what extent is it possible to develop an automated algorithm that provide accurate support in the tracking of neutrophils from time-lapse sequences in the 3D spatial domain?** We divide 3D cell tracking into three parts: 3D cell segmentation, feature extraction, and trajectory linkage. 3D segmentation is conducted using a 3D U-Net model. The 3D segmentation ground truth data is labeled manually. Next, instead of using a deep learning model to automatically learn the cell movement motion in **Chapter 4**, we design and manually extract the handcrafted cell features. Our proposed rule-based feature-weighted approach has three features which include cell distance, cell movement direction, and average cell movement distance. They are selected after observing the moving neutrophils and characterizing them. Subsequently, we link the trajectories with different linkage methods based on the calculated feature similarity between pairs of candidate cells frame by frame. In contrast to **Chapter 4**, using a rule-based method rather than a deep learning method is because deep learning models require a large amount of ground truth data. However, annotation in 3D + T space is not only laborious and time-consuming but also not effective without proper labeling tools. Nevertheless, to evaluate the performance, we still need ground truth data. We come up with the idea of adding an extra Z coordinate on the X, and Y-axis coordinates that have already been annotated in **Chapter 4** and correcting X, and Y coordinates accordingly. In this manner, we annotate 20 ground truth trajectories for our dataset. The experimental results show our specific pipeline designed for neutrophils does improve the tracking performance compared to other state-of-the-art methods.

6.2 Discussion

Our proposed methodologies and pipelines in the thesis have addressed five research questions and achieved promising results in terms of two applications. They, however, still have some limitations which can be discussed from the following perspectives.

6.2.1 Data Perspective

(1) In **Chapter 2** and **Chapter 3**, the 20-layer raw pollen image stack captured using a bright-field microscope, are through-focus-images. The expert has not adjusted the focal

level of each Z-plane and only scans through the Z-axis by automated procedures of the microscope itself. In this case, it can be possible that some representative feature information is lost. This is why we project the 3D data into three different projections to acquire more features and classify them as 2D images. The focused 3D raw pollen data collection will be considered using CLSM in further research, and it supposes to achieve a different classification perspective.

(2) In **Chapter 4** and **Chapter 5**, in fact, the expert captures neutrophils and macrophages in the tail of zebrafish larvae, aiming to track both of the two types of leukocytes. The raw 3D time-lapse data acquisition with a confocal microscope has been set to one-minute intervals over a 2 hr period of time. The longer the time interval, the longer distance the cell migrates, so it is easy to lose the track. All three channels of neutrophil, macrophage, and brightfield images from two groups, experimental and controlled groups, of zebrafish need to be taken at the same time within one minute. This means the number of scanning sections on the Z-axis is limited. This is why we originally obtain an 8-layer 3D image.

Moreover, neutrophils and macrophages have different morphology and movement pattern. It is impossible to design a tracking algorithm to fit both of them well. Furthermore, macrophages have a more irregular shape and they are difficult to be distinguished even by experienced experts, and thus, so far, also much less by algorithms. In contrast, neutrophils can be tracked easily in practice. This motivated us to focus on tracking neutrophils only.

Overall, with these limitations mentioned above, there is a lot of room to improve the methods. Not only by improving data quality, but also, working with macrophage tracking needs to be considered in future research.

(3) In **Chapter 4**, the ground truth preparation for 2D cell tracking trajectories, is accomplished in a straightforward manner, in which a cell candidate is manually tracked through the whole sequence directly. The expert does not take the merge and split events into account during the labeling process. This is why we use the linkage method to subsequently solve this problem. Because the deep learning model can not learn these modes from ground truth data.

In **Chapter 5**, due to the lack of effective and convenient labeling tools in 3D + T space, we use a rule-based method to do tracking. Ideally, deep learning models are the mainstream nowadays, and training a learning model from annotated ground truth data should achieve high tracking ability. There has been research after methods to produce 3D annotations [141][157] but these do not appear to be reproducible in our case. Therefore, designing our annotation tool is essential. It should be further considered for future developments in this area.

6.2.2 Hardware Perspective

Deep learning systems require powerful hardware because they have a large amount of data being processed. In this thesis, all the deep learning models that we used are trained on a dedicated server equipped with two NVidia GeForce GTX 2070 with 8 GB GPUs and a processor with 64 GB RAM in our group. With the two GPUs, we conducted pollen classification in **Chapter 2** and **3** successfully. In **Chapter 4**, We also trained the 2D cell segmentation and 2D cell motion tracking models successfully. However, for 3D cell segmentation in **Chapter 5**, the image size of $512 \times 512 \times 8$ is too big to use as the input and output of end-to-end training on our GPUs, even with the setting "batchsize=1". To solve this, we apply a strategy to divide the original image into small patches to train a deep learning model, after training the small patches are stitched to the original size again. Although the 3D cell segmentation can be done with this strategy, the tracking task requires 3D time-series data as the input and output of the model. This needs more GPU memory than the Setup we have been employing. Except for the difficulties of ground truth data labeling, this is another reason we use a rule-based method rather than a deep learning method to conduct a 3D cell tracking task in this thesis.

6.2.3 Algorithmic Perspective

Despite of the aforementioned limitations we have achieved promising results using our algorithms in terms of our application study, there is still room for improvement.

(1) Cell segmentation. Ideally, no matter with 2D or 3D cell segmentation, the over-segmentation and under-segmentation are the key problems that need to be prevented. The segmented results will directly affect the performance of the subsequent tracking part. However, not one approach from either rule-based or deep learning methods can perfectly solve it. In the research presented in this thesis, we compared several existing segmentation methods in order to find the method with the best performance with respect to our data. In this manner, the subsequent tracking will benefit from the good segmentation performance. So for the analysis of the neutrophils we focused on the tracking rather than designing our own segmentation algorithms. In the future work, it will be possible to take this into account.

(2) Cell tracking. Compared with some state-of-art methods, our pipelines for 2D cell tracking and 3D cell tracking improve the performance in the experiments from **Chapter 4** and **5**. But in fact, the tracking rate of algorithms still cannot accomplish sufficient accuracy compared to the manual annotation, especially in a long-range time-lapse sequence. In

particular, the cell merging and splitting events are not solved satisfactorily even with a lot of rules designed in our pipeline. This is an absolute challenge in many cell tracking tasks.

(3) Trajectory visualization. In the 2D tracking task in **Chapter 4**, we link the position of each tracked trajectory with colored lines and map it to the raw data. The trajectories can be shown through a 2D time-lapse movie easily. However, in **Chapter 5**, it is a much more complex process to visualize the tracked trajectories within a 3D + T space. The process could include but is not restricted to cell detection, cell coordinates calculation, and 3D space rotation. Thus, it could be a future visualization work that can be achieved together with 3D ground truth annotation as we mentioned previously.

(4) Comparison of 2D and 3D tracking results. In **Chapter 4** and **Chapter 5**, we demonstrate that our dedicated algorithms outperform other state-of-art methods. This is because all of the comparisons are based on the same standard, ground truth data. However, to date, it is hard to conclude whether the performance of 3D tracking is better than 2D tracking. In **Chapter 4**, we annotate 110 ground truth data while only 20 ground truth is labeled in **Chapter 5**. The overlapped 20 ground truth data can not realistically represent the performance of our two proposed pipelines due to the small size of sampling. Therefore, enlarging the number of ground truth data for 3D tracking is essential for future research.

(5) Macrophage tracking. All previously mentioned algorithms are designed for neutrophils. Neutrophils have the features of clear cell borders that can be easily distinguished while having a faster motion. Compared with neutrophils, macrophages move slower but are more irregular. It is even difficult for biologists to tell them apart. With these completely different characteristics, the proposed algorithms for not only segmentation but also tracking are definitely not suitable to use on the two types of cells simultaneously. For macrophages, segmentation should be the more challenging work compared with the tracking part. A slower movement and velocity could make it easier to trace these macrophages.

6.3 Future Research Directions

In the previous Chapters, we have presented many approaches to address the research questions regarding two applications. A wide variety of future work is also encouraged to advance them further. In this section, we briefly discuss the possible future research directions regarding the aforementioned themes.

(1) We have labeled the ground truth data for 2D cell tracking without taking the merging and splitting events into account. If we can consider all of these possible cell events during

ground truth labeling, the deep learning model used in the tracking part could learn more potential cell motion patterns so as to achieve a better performance in theory.

(2) At present, specifical focus on the difficulties of labeling of ground truth trajectories in 3D + T space is essential, and the design dedicated interactive tools very much needed. It will be very valuable to realize labeling of 3D time-series data in the cell tracking field. Suppose we already have a smart tool that can help label ground truth data, we can apply deep learning approaches to conduct 3D cell tracking tasks easily. Consequently, visualizing the cell trajectories in 3D + T space from different angles is also possible and it reduces the dependency on commercial software. A complete working route as mentioned above might be feasible in the future.

(3) Except for the neutrophils' tracking task explored in our thesis, tracking macrophages will be a more challenging task. Nevertheless, investigating the migration behaviors of macrophages is a necessary step to take in future research.

References

[1] Fons J. Verbeek. Seeing small things big. In *Leiden University*, 2018.

[2] Georgios C. Manikis, Kostas Marias, Eleftherios Alissandrakis, Louis Perrotto, Elisavet Savvidaki, and et al. Pollen grain classification using geometrical and textural features. In *2019 IEEE International Conference on Imaging Systems and Techniques (IST)*, pages 1–6, 2019.

[3] Frida Sommer, Vincenzo Torraca, and Annemarie H. Meijer. Chemokine receptors and phagocyte biology in zebrafish. *Frontiers Immunology*, 2020.

[4] Yufei Xie, Annemarie H. Meijer, and Marcel J. M. Schaaf. Modeling inflammation in zebrafish for the development of anti-inflammatory drugs. *Frontiers in Cell and Developmental Biology*, 8, 2021.

[5] Leiba Jade, Resul Özbilgiç, Liz Hernández, Maria Demou, Georges Lutfalla, Laure Yatime, and Mai Nguyen-Chi. Molecular actors of inflammation and their signaling pathways: Mechanistic insights from zebrafish. *Biology*, 12, 2023.

[6] Xiaoqin Tang. *Computational optimisation of optical projection tomography for 3D image analysis*. PhD thesis, Leiden University, 6 2020.

[7] Douglas E Chandler and Robert W Roberson. *Bioimaging: current concepts in light and electron microscopy*. Jones and Bartlett Publishers, 2009.

[8] Lu Cao. *Biological model representation and analysis*. PhD thesis, Leiden University, 11 2014.

[9] Fuyong Xing, Yuanpu Xie, Hai Su, Fujun Liu, and Lin Yang. Deep learning in microscopy image analysis: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 29(10):4550–4568, 2018.

[10] Gaudenz Danuser. Computer vision in cell biology. *Cell*, 147(5):973–978, 2011.

[11] Thomas M Ward, Pietro Mascagni, Yutong Ban, Guy Rosman, Nicolas Padoy, Ozanan Meireles, and et al. Computer vision in surgery. *Surgery*, 169(5):1253–1256, 2021.

[12] Jiechao Ma, Yang Song, Xi Tian, Yiting Hua, Rongguo Zhang, and Jianlin Wu. Survey on deep learning for pulmonary medical imaging. *Frontiers of medicine*, 14:450–469, 2020.

[13] Dinggang Shen, Guorong Wu, and Heung-II Suk. Deep learning in medical image analysis. *Annual Review of Biomedical Engineering*, 21(19):221–248, 2017.

[14] Mohamed A. Abdou. Literature review: efficient deep neural networks techniques for medical image analysis. *Neural Computing and Applications*, 34:5791–5812, 2022.

[15] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, 2018.

[16] Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, and et al. Deep learning vs. traditional computer vision. In *In Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference*, volume 943, pages 128–144, 2020.

[17] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. Deep learning in bioinformatics. *Briefings in Bioinformatics*, 18(5):851–869, 2017.

[18] Bram van Ginneken. Fifty years of computer analysis in chest imaging: rule-based, machine learning, deep learning. *Radiological Physics and Technology*, 10:23–32, 2017.

[19] Zhichao Liu, Luhong Jin, Jincheng Chen, Qiuyu Fang, Sergey Ablameyko, Zhaozheng Yin, and Yingke Xu. A survey on applications of deep learning in microscopy image analysis. *Computers in Biology and Medicine*, 134, 2021.

[20] Vebjorn Ljosa, Katherine L Sokolnicki, and Anne E Carpenter. Annotated high-throughput microscopy image sets for validation. *Nature Methods*, 9(7):637–637, 2012.

[21] Vladimír Ulman and et al. An objective comparison of cell-tracking algorithms. *Nature Methods*, 14(12):1141–1152, 2017.

[22] Anubha Gupta and et al. Segpc-2021: A challenge dataset on segmentation of multiple myeloma plasma cells from microscopic images. *Medical Image Analysis*, 83, 2023.

[23] Cell tracking challenge. http://celltrackingchallenge.net/, 2012–2020.

[24] Fabrice Cordelieres. Manual tracking. https://imagej.nih.gov/ij/plugins/track/track.html, 2004–2005.

[25] Wei Zhou, Zhiwu Xia, Peng Dou, Tao Su, and Haifeng Hu. Aligning image semantics and label concepts for image multi-label classification. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 19(2):1–23, 2023.

[26] Rui Liu, Wei Dai, Tianyi Wu, Min Wang, Song Wan, and Jun Liu. Aimic: Deep learning for microscopic image classification. *Computer Methods and Programs in Biomedicine*, 226, 2022.

[27] Hadi Rezaeilouyeh, Ali Mollahosseini, and Mohammad H. Mahoor. Microscopic medical image classification framework via deep learning and shearlet transform. *Journal of Medical Imaging*, 3(4), 2016.

[28] Long D. Nguyen, Dongyun Lin, Zhiping Lin, and Jiuwen Cao. Deep cnns for microscopic image classification by exploiting transfer learning and feature concatenation. In *2018 IEEE International Symposium on Circuits and Systems*, pages 1–5, 2018.

[29] William Meiniel, Jean-Christophe Olivo-Marin, and Elsa D. Angelini. Denoising of microscopy images: A review of the state-of-the-art, and a new sparsity-based method. *IEEE Transactions on Image Processing*, 27(8):3842–3856, 2018.

[30] Souad Larabi-Marie-Sainte, Reham Alskireen, and Sawsan Alhalawani. Emerging applications of bio-inspired algorithms in image segmentation. *Electronics*, 10(24):3226, 2021.

[31] Zhaozheng Yin, Kang Li, Takeo Kanade, and Mei Chen. Understanding the optics to aid microscopy image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2010*, volume 6361, 2010.

[32] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:3523–3542, 2022.

[33] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4):13–es, 2006.

[34] Martin Maška and et al. A benchmark for comparison of cell tracking algorithms. *Bioinformatics*, 30(11):1609–1617, 2014.

[35] Erik Meijering, Oleh Dzyubachyk, and Ihor Smal. Methods for cell and particle tracking. *Methods in Enzymology*, 504:183–200, 2012.

[36] Thomas Bieber and et al. Global allergy forum and 3rd davos declaration 2015: Atopic dermatitis/eczema: Challenges and opportunities toward precision medicine. *Allergy*, 71(5):588–592, 2016.

[37] Oliver Pfaar, Hongfei Lou, Yuan Zhang, Ludger Klimek, and Luo Zhang. Recent developments and highlights in rhinitis and allergen immunotherapy. *Allergy*, 73(12):2306–2313, 2018.

[38] Innes Asher and et al. World allergy organization guidelines for prevention of allergy and allergic asthma. *International Archives of Allergy and Immunology*, 135(1):83–92, 2004.

[39] Kostas D. Karatzas, Marina Riga, and Matt Smith. Presentation and dissemination of pollen information. In *Allergenic Pollen*, pages 217–247. Springer, 2012.

[40] Carmi Geller-Bernstein and Jay M Portnoy. The clinical utility of pollen counts. *Clinical Reviews in Allergy and Immunology*, 57(3):340–349, 2019.

[41] JM Hirst. An automatic volumetric spore trap. *Annals of Applied Biology*, 39:257–265, 1952.

[42] Beug Hans-Jürgen. Leitfaden der pollenbestimmung für mitteleuropa und angrenzende gebiete. In *Verlag Dr. Friedrich Pfeil*. 2004.

[43] Chiara Ziello, Tim H. Sparks, Nicole Estrella, Jordina Belmonte, Karl C. Bergmann, Edith Bucher, and et al. Changes to airborne pollen counts across europe. *Plos One*, 7(4):e34076, 2012.

[44] Erdtman G. The acetolysis method-a revised description. *Svensk Botanisk Tidskrift*, 54(4):561–564, 1960.

[45] Gretchen D. Jones. Pollen analyses for pollination research, acetolysis. *Journal of Pollination Ecology*, 13:203–217, 2014.

[46] Angelica Tiotiu, Andrea Brazdova, Cyril Longé, Patrice Gallet, Martine Morisset, Virginie Leduc, and et al. Urtica dioica pollen allergy: Clinical, biological, and allergomics analysis. *Annals of Allergy, Asthma Immunology*, 117(5):527–534, 2016.

[47] D'Amato G. and Liccardi G. Pollen-related allergy in the european mediterranean area. *Clinical and Experimental Allergy*, 24(3):210–219, 1994.

[48] Giorgio Ciprandi, Paola Puccinelli, Cristoforo Incorvaia, and Simonetta Masieri. Parietaria allergy: An intriguing challenge for the allergist. *Medicina*, 54(6):106, 2018.

[49] Bass D.A. and Bass D.J. Parietaria judaica l. a cause of allergic disease in sydney. a study of habit and spread of the weed. *Review of Palaeobotany and Palynology*, 64(1):97–101, 1990.

[50] Christina Fotiou, Athanasios Damialis, Nikolaos Krigas, John M Halley, and Despoina Vokou. Parietaria judaica flowering phenology, pollen production, viability and atmospheric circulation, and expansive ability in the urban environment: impacts of environmental factors. *Internation Journal of Biometeorology*, 55(1):35–50, 2011.

[51] D'Amato G, Ruffilli A, Sacerdoti G, and Bonini S. Parietaria pollinosis: a review. *Allergy*, 47(5):443–449, 1992.

[52] Roser GuardiaCentre and Jordina BelmonteBotanical. Phenology and pollen production of parietaria judaica l. in catalonia (ne spain). *Grana*, 43(1):57–64, 2010.

[53] Corbi A.L., Pelaez A., Errigo E., and Carreira J. Cross-reactivity between parietaria judaica and parietaria officinalis. *Annals of Allergy*, 54(2):142–147, 1985.

[54] Bousquet J., Hewitt B., Guérin B., Dhivert H., and Michel F.B. Allergy in the mediterranean area. ii: Cross-allergenicity among urticaceae pollens (parietaria and urtica). *Clinical Allergy*, 16(1):57–64, 1986.

[55] D'Amato G., Cecchi L., Bonini S., Nunes C., Annesi-Maesano I., Behrendt H., and et al. Allergenic pollen and pollen allergy in europe. *Allergy*, 62(9):976–990, 2007.

[56] Rodríguez A.M., Palacios I.S., Molina R.T, and Corchero A.M. Urtica membranacea and the importance of its separation from the rest of the urticaceae in aeropalynological studies carried out in the mediterranean region. *Plant Biosystems*, 140(3):321–332, 2006.

[57] Punt W. and Malotaux Marieke. Cannabaceae, moraceae and urticaceae. *Review of Palaeobotany and Palynology*, 42(1-4):23–44, 1984.

[58] Holt K.A. and Bennett K.D. Principles and methods for automated palynology. *New Phytologist*, 203(3):735–742, 2014.

[59] De Sá otero Ma Pilar, González Amelia, Rodríguez-Damián M., and Cernadas E. Computer-aided identification of allergenic species of urticaceae pollen. *Grana*, 43(4):224–230, 2004.

[60] Ariadne Barbosa Gonçalves, Junior Silva Souza, Gercina Gonçalves da Silva, Marney Pascoli Cereda, Arnildo Pott, and Marco Hiroshi Naka. Feature extraction and machine learning for the classification of brazilian savannah pollen grains. *Plos One*, 11(6):e0157044, 2016.

[61] Hanane Menad, Farah Ben-Naoum, and Abdelmalek Amine. Deep convolutional neural network for pollen grains classification. In *National Study Day on Research on Computer Sciences*, 2019.

[62] Sevillano Víctor and Aznarte José L. Improving classification of pollen grain images of the polen23e dataset through three different applications of deep learning convolutional neural networks. *Plos One*, 13(9):e0201807, 2018.

[63] Sevillano Víctor, Holt Katherine, and Aznarte José L. Precise automatic classification of 46 different pollen types with convolutional neural networks. *Plos One*, 15(6):e0229751, 2020.

[64] Eraldo Ribeiro Amar Daood and Mark Bush. Pollen grain recognition using deep learning. *Advances in Visual Computing*, 10072(6):321–330, 2016.

[65] Surangi W. Punyasena, David K. Tcheng, Cassandra Wesseln, and Pietra G. Mueller. Classifying black and white spruce pollen using layered machine learning. *New Phytologist*, 196(3):937–944, 2012.

[66] Yılmaz Kaya, S. Mesut Pınar, M. Emre Erez, Mehmet Fidan, and James B. Riding. Identification of onopordum pollen using the extreme learning machine, a type of artificial neural network. *Palynology*, 38(1):129–137, 2014.

[67] Amirreza Mahbod, Gerald Schaefer, Rupert Ecker, and Isabella Ellinger. Pollen grain microscopic image classification using an ensemble of fine-tuned deep convolutional neural networks. *arXiv preprint arXiv:2011.07428*, 2020.

[68] Duistermaat Leni. Heukels' flora van nederland 24th edition. Groningen/Utrecht, Noordhoff Uitgevers, 2020.

[69] Rasband W.S. Imagej, US National Institutes of Health, Bethesda, MD, U.S.A. http://rsb.info.nih.gov/ij/, 1997–2006.

[70] Wheeler R. Extended depth of field. http://www.richardwheeler.net.

[71] Simonyan Karen and Zisserman Andrew. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[72] Howard G. Andrew, Zhu Menglong, Chen Bo, Kalenichenko Dmitry, Wang Weijun, and Weyand Tobias. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[73] Sandler Mark, Howard Andrew, Zhu Menglong, Zhmoginov Andrey, and Chen Liang-Chieh. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[74] Chollet F.K. https://github.com/fchollet/keras, 2015.

[75] Geus André R. de, Barcelos Celia A.Z., Batista Marcos A., and Silva Sérgio F. da. Large-scale pollen recognition with deep learning. In *2019 27th European Signal Processing Conference (EUSIPCO)*, pages 1–5, 2019.

[76] Amar I. Daood, Eraldo Ribeiro, and Mark B. Bush. Sequential recognition of pollen grain z-stacks by combining cnn and rnn. In *The Florida AI Research Society*, 2018.

[77] Ingrid C. Romero, Shu Kong, Charless C. Fowlkes, and Surangi W. Punyasena. Improving the taxonomy of fossil pollen using convolutional neural networks and superresolution microscopy. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 117, pages 28496–28505, 2020.

[78] Rodriguez-Damian M., Cernadas E., Formella A., Fernandez-Delgado M., and Pilar De Sa-Otero. Automatic detection and classification of grains of pollen based on shape and texture. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 36(4):531–542, 2006.

[79] Amar Daood, Eraldo Ribeiro, and Mark Bush. Classifying pollen using robust sequence alignment of sparse z-stack volumes. In *International Symposium on Visual Computing*, pages 331–340, 2016.

[80] Chloe He, Gerard Glowacki, and Alexis Gkantiragas. Unsupervised representations of pollen in bright-field microscopy. *arXiv preprint arXiv:1908.01866*, 2019.

[81] Kadaikar Aysha, Pan Yan, Zhang Qiaoxi, Conde-Céspedes Patricia, Trocan Maria, Amiel Frédéric, and et al. Variable complexity neural networks comparison for pollen classification. *International Journal of Biology and Biomedical Engineering*, 2019.

[82] Shaoning Zeng, Bob Zhang, Jianping Gou, and Yong Xu. Regularization on augmented data to diversify sparse representation for robust image classification. *IEEE Transactions on Cybernetics*, 52(6):4935–4948, 2022.

[83] Ramón Gallardo-Caballero, Carlos J García-Orellana, Antonio García-Manso, Horacio M González-Velasco, Rafael Tormo-Molina, and Miguel Macías-Macías. Precise pollen grain detection in bright field microscopy using deep learning techniques. *Sensors*, 19(16):3583, 2019.

[84] Katherine Angharad Holt, G. P. Allen, R.M. Hodgson, and et al. Progress towards an automated trainable pollen location and classifier system for use in the palynology laboratory. *Review of Palaeobotany and Palynology*, 167(3-4):175–183, 2011.

[85] Jose Oteros, Gudrun Pusch, Ingrid Weichenmeier, Ulrich Heimann, Rouven Möller, Stefani Röseler, and et al. Automatic and online pollen monitoring. *International Archives of Allergy and Immunology*, 167(3):158–166, 2015.

[86] Eric Sauvageat, Yanick Zeder, Kevin Auderset, Bertrand Calpini, Bernard Clot, Benoît Crouzy, and et al. Real-time pollen monitoring using digital holography. *Atmospheric Measurement Techniques*, 13(3):1539–1550, 2020.

[87] Gennaro D'Amato, Herberto Jose Chong-Neto, Olga Patricia Monge Ortega, and et al. The effects of climate change on respiratory allergy and asthma induced by pollen and mold allergens. *In Allergy: European Journal of Allergy and Clinical Immunology*, 75(9):2219–2228, 2020.

[88] Biedermann T., Winther L., Till S.J., Panzner P., Knulst A., and Valovirta E. Birch pollen allergy in europe. In *Allergy: European Journal of Allergy and Clinical Immunology*, volume 74, pages 1237–1248, 2019.

[89] Ariadne Barbosa Gonçalves, Junior Silva Souza, Gercina Gonçalves Da Silva, Marney Pascoli Cereda, Arnildo Pott, Marco Hiroshi Naka, and Hemerson Pistori. Feature extraction and machine learning for the classification of brazilian savannah pollen grains. *Plos One*, 11(6), 2016.

[90] Battiato Sebastiano, Ortis Alessandro, Trenta Francesca, Ascari Lorenzo, Politi Mara, and Siniscalco Consolata. Detection and classification of pollen grain microscope images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 4220–4227, 2020.

[91] Dunker Susanne, Motivans Elena, Rakosy Demetra, Boho David, Mäder Patrick, Hornick Thomas, and Knight Tiffany M. Pollen analysis using multispectral imaging flow cytometry and deep learning. *New Phytologist*, 229(1):593–606, 2021.

[92] Pospiech Matej, Javůrková Zdeňka, Hrabec Pavel, Štarha Pavel, Ljasovská Simona, Bednář Josef, and Tremlová Bohuslava. Identification of pollen taxa by different microscopy techniques. *Plos One*, 16(9):e0256808, 2021.

[93] Manikis G.C., Marias K., Alissandrakis E., Perrotto L., Savvidaki E., and Vidakis N. Pollen grain classification using geometrical and textural features. In *IST 2019 - IEEE International Conference on Imaging Systems and Techniques, Proceedings*, pages 1–6, 2019.

[94] Daood Amar, Ribeiro Eraldo, and Bush Mark. Pollen grain recognition using deep learning. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10072, pages 321–330, 2016.

[95] Schiele Julian, Rabe Fabian, Schmitt Maximilian, Glaser Manuel, Haring Franziska, Brunner Jens O., Bauer Bernhard, Schuller Bjorn, Traidl-Hoffmann Claudia, and Damialis Athanasios. Automated classification of airborne pollen using neural networks. In *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, pages 4474–4478, 2019.

[96] Marcel Polling, Chen Li, Lu Cao, Fons J. Verbeek, Letty A. de Weger, Jordina Belmonte, Concepción De Linares, Joost Willemse, Hugo de Boer, and Barbara Gravendeel. Neural networks for increased accuracy of allergenic pollen monitoring. *Scientific Reports*, 11, 2021.

[97] del Pozo-Baños Marcos, Ticay-Rivas Jaime R., Alonso Jesús B., and Travieso Carlos M. Features extraction techniques for pollen grain classification. *Neurocomputing*, 150:377–391, 2015.

[98] Marcos J. Víctor, Nava Rodrigo, Cristóbal Gabriel, Redondo Rafael, Escalante-Ramírez Boris, Bueno Gloria, Déniz Óscar, González-Porto Amelia, Pardo Cristina, Chung François, and Rodríguez Tomás. Automated pollen identification using microscopic imaging and texture analysis. *Micron*, 68:36–46, 2015.

[99] Astolfi Gilberto, Gonçalves Ariadne Barbosa, Menezes Geazy Vilharva, Borges Felipe Silveira Brito, Astolfi Angelica Christina Melo Nunes, Matsubara Edson Takashi, Alvarez Marco, and Pistori Hemerson. Pollen73s: An image dataset for pollen grains classification. *Ecological Informatics*, 60, 2020.

[100] Rodrìuez-Damián Marìa, Cernadas Eva, Formella Arno, Fernández-Delgado Manuel, and De Sá-Otero Pilar. Automatic detection and classification of grains of pollen based on shape and texture. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 36(4):531–542, 2006.

[101] Aguet François, Van De Ville Dimitri, and Unser Michael. Model-based 2.5-d deconvolution for extended depth of field in brightfield microscopy. *IEEE Transactions on Image Processing*, 17(7):1144–1153, 2008.

[102] Bountris P, Farantatos E, and Apostolou N. Advanced image analysis tools development for the early stage bronchial cancer detection. *World Academy of Science, Engineering and Technology*, 1(9), 2007.

[103] Lu Cao, Marjo de Graauw, Kuan Yan, Leah Winkel, and Fons J. Verbeek. Hierarchical classification strategy for phenotype extraction from epidermal growth factor receptor endocytosis screening. *BMC Bioinformatics*, 17(196), 2016.

[104] Dunn G.A. and Brown A.F. Alignment of fibroblasts on grooved surfaces described by a simple geometric transformation. *Journal of Cell Science*, 83:313–340, 1986.

[105] Chudyk Celeste, Castaneda Hugo, Leger Romain, Yahiaoui Islem, and Boochs Frank. Development of an automatic pollen classification system using shape, texture and aperture features. In *LWA 2015 Workshops: KDML, FGWM, IR, and FGDB*, pages 65–74, 2015.

[106] Rodríguez-Damián M., Cernadas E., Formella A., and Sá-Otero P. Pollen classification using brightness-based and shape-based descriptors. In *Proceedings - International Conference on Pattern Recognition*, volume 2, pages 212–215, 2004.

[107] Dalal Navneet and Triggs Bill. Histograms of oriented gradients for human detection. In *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, 2005.

[108] Haralick Robert M., Dinstein Its'hak, and Shanmugam K. Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics*, 3(6):610–621, 1973.

[109] Hu Ming Kuei. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8(2):179–187, 1962.

[110] Jiliang Tang, Salem Alelyani, and Huan Liu. Feature selection for classification: A review. In *Data Classification: Algorithms and Applications*, number 28, pages 37–64. 2014.

[111] Jain Divya and Singh Vijendra. Feature selection and classification systems for chronic disease prediction: A review, 2018.

[112] Cossetin Marcelo J., Nievola Julio C., and Koerich Alessandro L. Facial expression recognition using a pairwise feature selection and classification approach. In *Proceedings of the International Joint Conference on Neural Networks*, pages 5149–5155, 2016.

[113] Popescu Madalina Cosmina and Sasu Lucian Mircea. Feature extraction, feature selection and machine learning for image classification: A case study. In *2014 International Conference on Optimization of Electrical and Electronic Equipment, OPTIM 2014*, pages 968–973, 2014.

[114] Silla Carlos N. and Freitas Alex A. A survey of hierarchical classification across different application domains, 2011.

[115] Kiritchenko Svetlana, Matwin Stan, and Famili A. Fazel. Functional annotation of genes using hierarchical text categorization. *In Proc. of the BioLINK SIG: Linking Literature, Information and Knowledge for Biology (held at ISMB-05*, 2005.

[116] Krizhevsky Alex, Sutskever Ilya, and Hinton Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[117] Simonyan Karen and Zisserman Andrew. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[118] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[119] Amirreza Mahbod, Gerald Schaefer, Rupert Ecker, and Isabella Ellinger. Pollen grain microscopic image classification using an ensemble of fine-tuned deep convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[120] David Gutierrez Arias, Marcos Vinicius Mussel Cirne, Josimar Edinson Chire, and Helio Pedrini. Classification of pollen grain images based on an ensemble of classifiers. In *Proceedings - 16th IEEE International Conference on Machine Learning and Applications, ICMLA 2017*, pages 234–240, 2017.

[121] Wanbin Hu, Leonie van Steijn, Chen Li, Fons J. Verbeek, Lu Cao, Roeland M. H. Merks, and Herman P. Spaink. A novel function of tlr2 and myd88 in the regulation of leukocyte cell migration behavior during wounding in zebrafish larvae. *Frontiers in Cell and Developmental Biology*, 9, 2021.

[122] Junya Hayashida and Ryoma Bise. Cell tracking with deep learning for cell detection and motion estimation in low-frame-rate. In *International Conference on Medical Image Computing and Computer-Assisted Intervention–MICCAI 2019*, volume 11764, pages 397–405, 2019.

[123] Akram Saad Ullah, Kannala Juho, Eklund Lauri, and Heikkilä Janne. Joint cell segmentation and tracking using cell proposals. In *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, pages 920–924, 2016.

[124] Hernandez David E., Chen Steven W., Hunter Elizabeth E., Steager Edward B., and Vijay Kumar. Cell tracking with deep learning and the viterbi algorithm. In *2018 International Conference on Manipulation, Automation and Robotics at Small Scales (MARSS)*, pages 1–6, 2018.

[125] Tim Scherr, Katharina Löffler, Moritz Böhland, and Ralf Mikut. Cell segmentation and tracking using cnn-based distance predictions and a graph-based matching strategy. *Plos One*, 15(12):e0243219, 2020.

[126] Kuan Yan, Fons J. Verbeek, Sylvia Le Devedec, and Bob van de Water. Cell tracking and data analysis of in vitro tumour cells from time-lapse image sequences. In *Proceedings of the Fourth International Conference on Computer Vision Theory and Applications–VISAPP 2009*, volume 1, 2009.

[127] Jean-Baptiste Lugagne, Haonan Lin, and Mary J. Dunlop. Delta: Automated cell segmentation, tracking, and lineage reconstruction using deep learning. *Plos Computational Biology*, 16(4):e1007673, 2020.

[128] Assaf Arbelle, Jose Reyes, Jia-Yun Chen, Galit Lahav, and Tammy Riklin Raviv. A probabilistic approach to joint cell tracking and segmentation in high-throughput microscopy videos. *Medical Image Analysis*, 47:140–152, 2018.

[129] Yousef Al-Kofahi, Alla Zaltsman, Robert Graves, Will Marshall, and Mirabela Rusu. A deep learning-based algorithm for 2-d cell segmentation in microscopy images. *BMC Bioinformatics*, 19(365), 2018.

[130] Tomas Vicar, Jan Balvan, Josef Jaros, Florian Jug, Radim Kolar, Michal Masarik, and Jaromir Gumulec. Cell segmentation methods for label-free contrast microscopy: review and comprehensive comparison. *BMC Bioinformatics*, 20(360), 2019.

[131] Fuyong Xing and Lin Yang. Robust nucleus/cell detection and segmentation in digital pathology and microscopy images: A comprehensive review. *IEEE Reviews in Biomedical Engineering*, 9:234–263, 2016.

[132] Otsu Nobuyuki. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[133] Vincent Luc and Soille Pierre. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:583–598, 1991.

[134] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of the IEEE conference on computer vision and pattern Recognition (CVPR), arXiv:1505.04597*, 2015.

[135] Xiao Xiao, Shen Lian, Zhiming Luo, and Shaozi Li. Weighted res-unet for high-quality retina vessel segmentation. In *2018 9th International Conference on Information Technology in Medicine and Education (ITME)*, pages 327–331, 2018.

[136] Steven Guan, Amir A. Khan, Siddhartha Sikdar, , and Parag V. Chitnis. Fully dense unet for 2-d sparse photoacoustic tomography artifact removal. *IEEE Journal of Biomedical and Health Informatics*, 2(2):568–576, 2020.

[137] Tao He, Hua Mao, Jixiang Guo, and Zhang Yi. Cell tracking using deep neural networks with multi-task learning. *Image and Vision Computing*, 60:142–153, 2017.

[138] Engin Turetken, Xinchao Wang, Carlos J. Becker, Carsten Haubold, and Pascal Fua. Network flow integer programming to track elliptical cells in time-lapse sequences. *IEEE Transaction on Medical Imaging*, 36(4):942–951, 2017.

[139] Magnusson Klas E.G. and Jaldén Joakim. A batch algorithm using iterative application of the viterbi algorithm to track cells and construct cell lineages. In *2012 9th IEEE International Symposium on Biomedical Imaging (ISBI)*, pages 382–385, 2012.

[140] Magnusson Klas E.G., Jalden Joakim, Gilbert Penney M., and Blau Helen M. Global linking of cell tracks using the viterbi algorithm. *IEEE Transaction on Medical Imaging*, 34(4):911–929, 2015.

[141] Christian Payer, Darko Štern, Marlies Feiner, Horst Bischof, and Martin Urschler. Segmenting and tracking cell instances with cosine embeddings and recurrent hourglass networks. *Medical Image Analysis*, 57:106–119, 2019.

[142] Marzieh R. Moghadam and Yi-Ping Phoebe Chen. Tracking neutrophil migration in zebrafish model using multi-channel feature learning. *IEEE Journal of Biomedical Health Informatics*, 25(4):1197–1205, 2021.

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

[144] Jean-Yves Tinevez, Nick Perry, Johannes Schindelin, Genevieve M. Hoopes, Gregory D. Reynolds, Emmanuel Laplantine, and et al. Trackmate: An open and extensible platform for single-particle tracking. *Methods*, 115:80–90, 2017.

[145] Katherine M. Henry, Luke Pase, Carlos Fernando Ramos-Lopez, Graham J. Lieschke, Stephen A. Renshaw, and Constantino Carlos Reyes-Aldasoro. Phagosight: An open-source matlab package for the analysis of fluorescent neutrophil and macrophage migration in a zebrafish model. *Plos One*, 8(8):e72636, 2013.

[146] J.A. Cornwell, J. Li, S. Mahadevan, J.S. Draper, G.L. Joun, H. Zoellner, and et al. Trackpad: Software for semi-automated single-cell tracking andlineage annotation. *SoftwareX*, 11, 2020.

[147] Anne E. Carpenter, Thouis R. Jones, Michael R. Lamprecht, Colin Clarke, In Han Kang, Ola Friman, and et al. Cellprofiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biology*, 7(10), 2006.

[148] Tashita Atsuki, Kobashi Syoji, Mori Yuki, Morimoto Masakazu, Aikawa Satoru, Yoshioka Yoshichika, and et al. Macrophage tracking using the hungarian algorithm in time lapse mr images. In *2015 7th International Conference on Emerging Trends in Engineering Technology (ICETET)*, pages 169–173, 2015.

[149] Réka Hollandi, Ákos Diósdi, Gábor Hollandi, Nikita Moshkov, and Péter Horváth. Annotatorj: an imagej plugin to ease hand annotation of cellular compartments. *Molecular Biology of the Cell*, 31(20):2179–2186, 2020.

[150] Michael Yeung, Evis Sala, Carola-Bibiane Schönlieb, and Leonardo Rundo. A mixed focal loss function for handling class imbalanced medical image segmentation. *ArXiv*, abs/2102.04525, 2021.

[151] Carole H Sudre, Wenqi Li, Tom Vercauteren, Sébastien Ourselin, and M. Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Proceedings of the IEEE conference on computer vision and pattern Recognition (CVPR)*, 2017.

[152] Kuan Yan and Fons J. Verbeek. Segmentation for high-throughput image analysis: Watershed masked clustering. *Leveraging Applications of Formal Methods, Verification and Validation. Applications and Case Studies*, 7610:25–41, 2012.

[153] Juan C. Caicedo, Jonathan Roth, Allen Goodman, Tim Becker, Kyle W. Karhohs, and Matthieu Broisin. Evaluation of deep learning strategies for nucleus segmentation in fluorescence images. *Cytometry Part A: the Journal of the International Society for Analytical Cytology*, 95(9):953–965, 2019.

[154] Kevin Smith, Daniel Gatica-Perez, Jean-Marc Odobez, and Sileye Ba. Evaluating multi-object tracking. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, pages 36–36, 2005.

[155] Martin Maška, Vladimír Ulman, David Svoboda, Pavel Matula, Petr Matula, Cristina Ederra, and et al. A benchmark for comparison of cell tracking algorithms. *Bioinformatics*, 30:1609–1617, 2014.

[156] Wanbin Hu, Shuxin Yang, Yasuhito Shimada, Magnus Münch, Rubén Marín-Juez, Annemarie H. Meijer, and Herman P. Spaink. Infection and rna-seq analysis of a zebrafish tlr2 mutant shows a broad function of this toll-like receptor in transcriptional and metabolic control and defense to mycobacterium marinum infection. *BMC Genomics*, 20:1–18, 2019.

[157] Marzieh R. Moghadam and Yi-Ping Phoebe Chen. Tracking leukocytes in intravital time lapse images using 3d cell association learning network. *Artificial Intelligence in Medicine*, 118, 2021.

[158] Duncan Carradice and Graham J Lieschke. Zebrafish in hematology: sushi or science? *Blood*, 111(7):3331–3342, 2008.

[159] Hsieh-Fu Tsai, Joanna Gajda, Tyler F.W. Sloan, Andrei Rares, and Amy Q. Shen. Usiigaci: Instance-aware cell tracking in stain-free phase contrast microscopy enabled by machine learning. *SoftwareX*, 9:230–237, 2019.

[160] Claire Mitchell, Lauryanne Caroff, Jose Alonso Solis-Lemus, Constantino Carlos Reyes-Aldasoro, Alessandra Vigilante, Fiona Warburton, and et al. Cell tracking profiler – a user-driven analysis framework for evaluating 4d live-cell imaging data. *Journal of Cell Science*, 133(22), 2020.

[161] Austin E.Y.T. Lefebvre, Dennis Ma, Kai Kessenbrock, Devon A. Lawson, and Michelle A. Digman. Automated segmentation and tracking of mitochondria in live-cell time-lapse images. *Nature Methods*, 18:1091–1102, 2021.

[162] Seifedine Kadry, Venkatesan Rajinikanth, David Taniar, Robertas Damaševičius, and Xiomara Patricia Blanco Valencia. Automated segmentation of leukocyte from hematological images—a study using various cnn schemes. *The Journal of Supercomputing*, 78:6974–6994, 2022.

[163] Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation, arxiv:1606.06650. In *Proceedings of the IEEE conference on computer vision and pattern Recognition (CVPR)*, volume 9901, pages 424–432, 2016.

[164] Chentao Wen, Takuya Miura, Venkatakaushik Voleti, Kazushi Yamaguchi, Motosuke Tsutsumi, Kei Yamamoto, and et al. 3deecelltracker, a deep learning-based pipeline for segmenting and tracking cells in 3d time lapse images. *Computational and Systems Biology, Neuroscience*, pages 1–36, 2021.

[165] S. Shailja, Jiaxiang Jiang, and B.S. Manjunath. Semi supervised segmentation and graph-based tracking of 3d nuclei in time-lapse microscopy. In *Proceedings of the IEEE conference on computer vision and pattern Recognition (CVPR), arXiv:2010.13343*, pages 385–389, 2021.

[166] Petit Manuel, Cerutti Guillaume, Godin Christophe, and Malandain Grégoire. Robust plant cell tracking in fluorescence microscopy 3d+t series. In *2022 IEEE 19th International Symposium on Biomedical Imaging (ISBI)*, pages 1–4, 2022.

[167] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE conference on computer vision and pattern Recognition (CVPR), arXiv:1712.00080*, 2018.

[168] David Borland, Carolyn M. McCormick, Niyanta K. Patel, Oleh Krupa, Jessica T. Mory, Alvaro A. Beltran, and et al. Segmentor: a tool for manual refinement of 3d microscopy annotations. *BMC Bioinformatics*, 22:1–12, 2021.

[169] H.W. Kuhn. The hungarian method for the assigment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1995.

[170] Johannes Schindelin, Ignacio Arganda-Carreras, Erwin Frise, Verena Kaynig, Mark Longair, Tobias Pietzsch, and et al. Fiji: an open-source platform for biological-image analysis. *Nature Methods*, 9:676–682, 2012.

Appendix A

Supplementary Materials in Chapter 2

Supplementary Table S1

Locations of all Urticaceae specimens and number of images. NL = the Netherlands, SP = Spain and PO = Portugal. *collected in 2018 and 2019, deposited in the Naturalis Biodiversity Center herbarium.

Species (n=total images)	**Geographical origin**	**Collection date**	**No. of images used**	**Deposition number**
Parietaria judaica L. (n = 1670)	Montejaque (SP)	17/10/2011	54	WAG.1186948
	Leiden, Stationsweg (NL)	19/11/2019	168	L.3993376*
	Huizen (NL)	20/09/2014	174	L.4303913
	Leiden, Robijnstraat (NL)	23/07/2012	139	L.2071680
	Den Haag (NL)	05/10/2018	392	L.3993377*
	Leiden, Paterstraatje	09/10/2018	250	L.3993378*
	Sassenplaat (NL)	03/07/2013	233	L.4304093
	Rotterdam, Hartelkanaal (NL)	27/09/2014	260	L.4304136
Parietaria officinalis L. (n = 1359)	Middelburg (NL)	26/06/2014	234	L.3974371
	Haarlem (NL)	13/07/2013	191	L.2073373
	Wageningse Polder (NL)	19/07/2012	64	WAG.1186992
	Leiden (NL)	07/2012	369	L.3963901
	Den Haag, Escamplaan (NL)	12/10/2018	383	L.3993379*
	Den Haag, Bosjes van Poot (NL)	01/08/2012	248	L.2071818
Urtica dioica L. (n = 1055)	Leiden, Hogeschool 1 (NL)	06/11/2019	316	L.3993380*
	Leiden, Hogeschool 2 (NL)	07/11/2019	299	L.3993381*
	Den Haag (NL)	17/11/2019	182	L.3993382*

	Leiden, Sandifortdreef (NL)	15/11/2019	191	L.3993383*
	Arnhem (NL)	29/05/2001	67	WAG.1188104
Urtica membranacea	Amsterdam (NL)	11/2018	521	L.3993384*
Poir. ex Savigny	Overloon (NL)	17/06/2014	135	L.3959964
(n = 1118)	Cape st. Vincent (PO)	03/1995	87	L.1629741
	Leiden, Sandifortdreef (NL)	15/11/2019	191	L.3993383*
	Den Haag (NL)	06/03/2019	375	L.3993385*
Urtica urens L.	Leiden (NL)	01/11/2019	128	L.3993386*
(n = 1270)	Castilla-la-Mancha (SP)	27/05/2016	165	WAG.1962413
	Zandvoort (NL)	05/08/2012	201	L.2071917
	Meijendel (NL)	12/08/2011	140	L.2074446
	Zwolle (NL)	29/04/2005	134	L.4271105
	Wassenaar (NL)	15/09/2002	219	L.4233917
	Den Haag (NL)	13/03/2020	283	L.3993387*

Supplementary Table S2

Probability scores for Urticaceae pollen grains scanned from aerobiological samples using the pre-trained VGG16 model with 5-fold cross-validation. *U. mem* = *Urtica membranacea*

Lleida (16-06-2019), n=63

Image No.	**Probability *Parietaria***	**Probability *Urtica***	**Probability *U.mem***	**Final ID Threshold 0.6**	**Final ID Threshold 0.7**
1	0.95	0.05	0.00	*Parietaria*	*Parietaria*
2	0.98	0.02	0.00	*Parietaria*	*Parietaria*
3	0.29	0.70	0.01	*Urtica*	*Urtica*
4	0.98	0.02	0.00	*Parietaria*	*Parietaria*
5	0.24	0.76	0.00	*Urtica*	*Urtica*
6	1.00	0.00	0.00	*Parietaria*	*Parietaria*
7	0.94	0.06	0.00	*Parietaria*	*Parietaria*
8	0.99	0.01	0.00	*Parietaria*	*Parietaria*
9	0.12	0.88	0.00	*Urtica*	*Urtica*
10	0.99	0.01	0.00	*Parietaria*	*Parietaria*

11	0.99	0.01	0.00	*Parietaria*	*Parietaria*
12	0.96	0.04	0.00	*Parietaria*	*Parietaria*
13	1.00	0.00	0.00	*Parietaria*	*Parietaria*
14	0.96	0.04	0.00	*Parietaria*	*Parietaria*
15	1.00	0.00	0.00	*Parietaria*	*Parietaria*
16	0.90	0.09	0.01	*Parietaria*	*Parietaria*
17	0.90	0.01	0.09	*Parietaria*	*Parietaria*
18	0.73	0.16	0.10	*Parietaria*	*Parietaria*
19	0.95	0.04	0.00	*Parietaria*	*Parietaria*
20	0.98	0.00	0.02	*Parietaria*	*Parietaria*
21	0.16	0.83	0.00	*Urtica*	*Urtica*
22	0.67	0.31	0.02	*Parietaria*	unknown
23	0.02	0.98	0.00	*Urtica*	*Urtica*
24	0.95	0.04	0.00	*Parietaria*	*Parietaria*
25	0.99	0.01	0.00	*Parietaria*	*Parietaria*
26	0.99	0.01	0.00	*Parietaria*	*Parietaria*
27	0.95	0.05	0.00	*Parietaria*	*Parietaria*
28	0.02	0.98	0.00	*Urtica*	*Urtica*
29	0.34	0.66	0.00	*Urtica*	unknown
30	1.00	0.00	0.00	*Parietaria*	*Parietaria*
31	0.98	0.02	0.00	*Parietaria*	*Parietaria*
32	1.00	0.00	0.00	*Parietaria*	*Parietaria*
33	0.99	0.01	0.00	*Parietaria*	*Parietaria*
34	0.92	0.02	0.06	*Parietaria*	*Parietaria*
35	0.57	0.41	0.02	unknown	unknown
36	1.00	0.00	0.00	*Parietaria*	*Parietaria*
37	0.87	0.13	0.00	*Parietaria*	*Parietaria*
38	0.99	0.00	0.00	*Parietaria*	*Parietaria*
39	0.97	0.03	0.00	*Parietaria*	*Parietaria*
40	0.58	0.41	0.01	unknown	unknown
41	0.98	0.02	0.00	*Parietaria*	*Parietaria*
42	0.70	0.29	0.00	*Parietaria*	*Parietaria*
43	0.84	0.16	0.00	*Parietaria*	*Parietaria*
44	0.97	0.02	0.01	*Parietaria*	*Parietaria*
45	0.83	0.17	0.00	*Parietaria*	*Parietaria*
46	0.99	0.00	0.00	*Parietaria*	*Parietaria*

47	1.00	0.00	0.00	*Parietaria*	*Parietaria*
48	0.99	0.00	0.00	*Parietaria*	*Parietaria*
49	0.96	0.04	0.01	*Parietaria*	*Parietaria*
50	0.00	1.00	0.00	*Urtica*	*Urtica*
51	0.99	0.01	0.00	*Parietaria*	*Parietaria*
52	0.99	0.00	0.01	*Parietaria*	*Parietaria*
53	0.91	0.04	0.05	*Parietaria*	*Parietaria*
54	0.95	0.04	0.00	*Parietaria*	*Parietaria*
55	0.90	0.10	0.00	*Parietaria*	*Parietaria*
56	0.95	0.05	0.00	*Parietaria*	*Parietaria*
57	0.99	0.01	0.00	*Parietaria*	*Parietaria*
58	1.00	0.00	0.00	*Parietaria*	*Parietaria*
59	0.99	0.01	0.00	*Parietaria*	*Parietaria*
60	0.17	0.82	0.00	*Urtica*	*Urtica*
61	0.41	0.56	0.02	unknown	unknown
62	0.98	0.02	0.00	*Parietaria*	*Parietaria*
63	0.76	0.21	0.03	*Parietaria*	*Parietaria*

Vielha (09-08-2019), n=26

Image No.	**Probability *Parietaria***	**Probability *Urtica***	**Probability *U.mem***	**Final ID Threshold 0.6**	**Final ID Threshold 0.7**
1	0.03	0.97	0.00	*Urtica*	*Urtica*
2	0.07	0.86	0.07	*Urtica*	*Urtica*
3	0.10	0.90	0.00	*Urtica*	*Urtica*
4	0.02	0.98	0.00	*Urtica*	*Urtica*
5	0.09	0.91	0.00	*Urtica*	*Urtica*
6	0.26	0.74	0.00	*Urtica*	*Urtica*
7	0.00	1.00	0.00	*Urtica*	*Urtica*
8	0.41	0.04	0.55	unknown	unknown
9	0.61	0.39	0.01	*Parietaria*	unknown
10	0.81	0.10	0.09	*Parietaria*	*Parietaria*
11	0.02	0.98	0.00	*Urtica*	*Urtica*
12	0.01	0.99	0.00	*Urtica*	*Urtica*
13	0.49	0.13	0.38	unknown	unknown

14	0.00	1.00	0.00	*Urtica*	*Urtica*
15	0.14	0.84	0.02	*Urtica*	*Urtica*
16	0.63	0.10	0.27	*Parietaria*	unknown
17	0.12	0.88	0.00	*Urtica*	*Urtica*
18	0.09	0.90	0.10	*Urtica*	*Urtica*
19	0.24	0.76	0.00	*Urtica*	*Urtica*
20	0.04	0.96	0.00	*Urtica*	*Urtica*
21	0.85	0.12	0.03	*Parietaria*	*Parietaria*
22	0.80	0.14	0.07	*Parietaria*	*Parietaria*
23	0.00	1.00	0.00	*Urtica*	*Urtica*
24	0.17	0.83	0.00	*Urtica*	*Urtica*
25	0.02	0.98	0.00	*Urtica*	*Urtica*
26	0.57	0.43	0.00	unknown	unknown

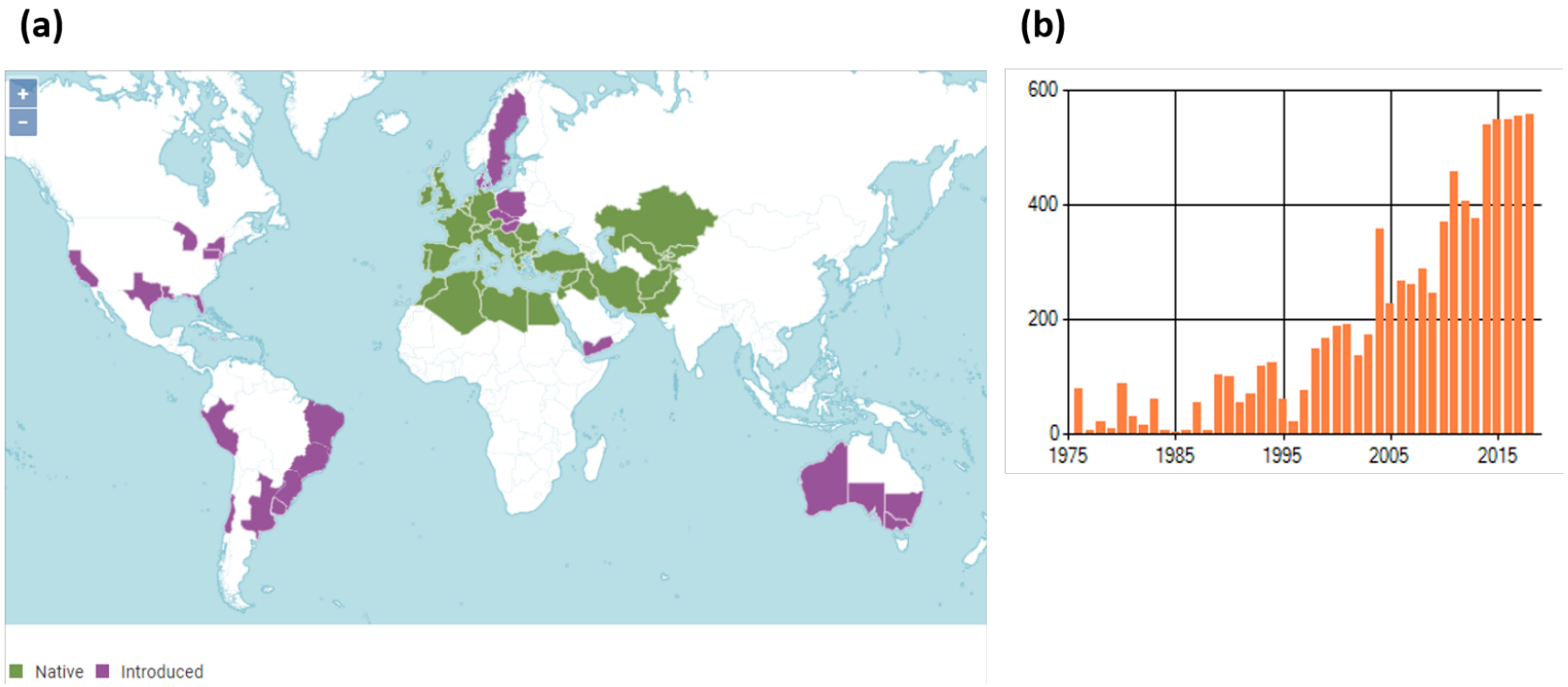
Leiden (23-08-2019), n=112

Image No.	**Probability *Parietaria***	**Probability *Urtica***	**Probability *U.mem***	**Final ID Threshold 0.6**	**Final ID Threshold 0.7**
1	0.04	0.96	0.00	*Urtica*	*Urtica*
2	0.01	0.99	0.00	*Urtica*	*Urtica*
3	0.07	0.93	0.00	*Urtica*	*Urtica*
4	0.16	0.83	0.00	*Urtica*	*Urtica*
5	0.19	0.81	0.00	*Urtica*	*Urtica*
6	0.02	0.98	0.00	*Urtica*	*Urtica*
7	0.00	1.00	0.00	*Urtica*	*Urtica*
8	0.28	0.72	0.00	*Urtica*	*Urtica*
9	0.11	0.89	0.00	*Urtica*	*Urtica*
10	0.34	0.66	0.00	*Urtica*	unknown
11	0.04	0.96	0.00	*Urtica*	*Urtica*
12	0.18	0.81	0.00	*Urtica*	*Urtica*
13	0.00	1.00	0.00	*Urtica*	*Urtica*
14	0.47	0.53	0.00	unknown	unknown
15	0.11	0.89	0.00	*Urtica*	*Urtica*
16	0.01	0.99	0.00	*Urtica*	*Urtica*
17	0.20	0.80	0.00	*Urtica*	*Urtica*

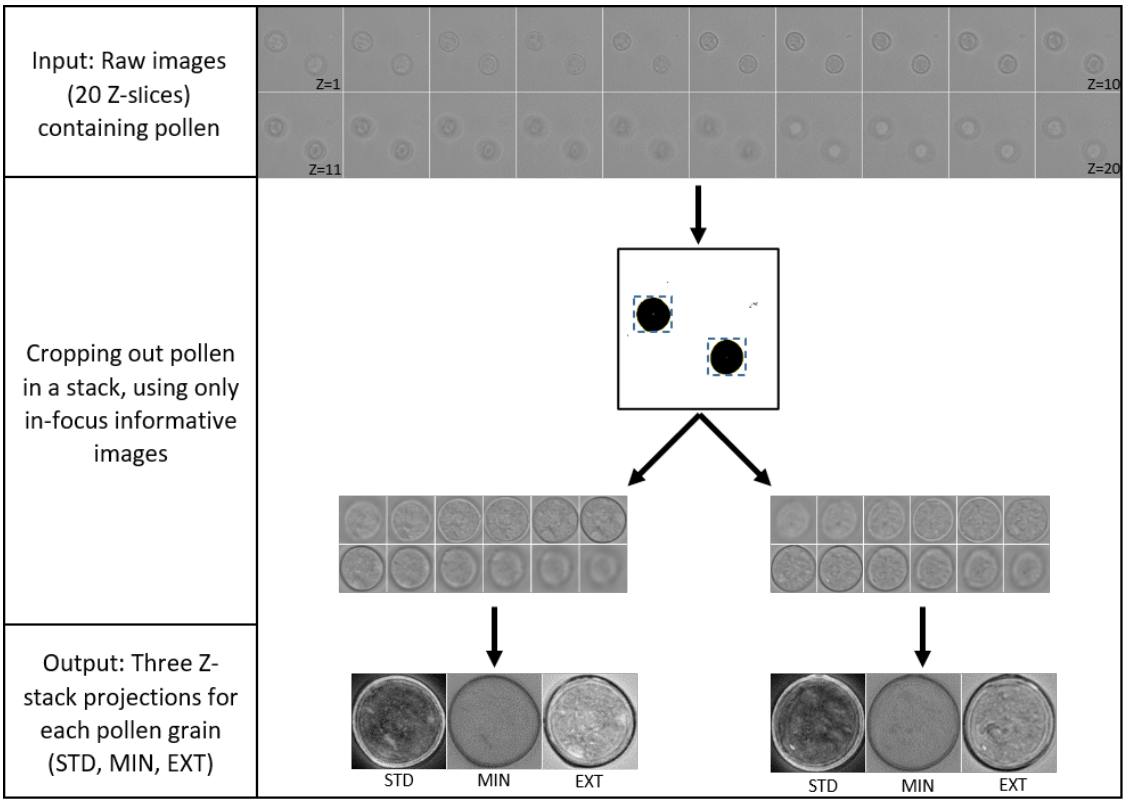
18	0.00	1.00	0.00	*Urtica*	*Urtica*
19	0.00	1.00	0.00	*Urtica*	*Urtica*
20	0.01	0.99	0.00	*Urtica*	*Urtica*
21	0.75	0.25	0.00	*Parietaria*	*Parietaria*
22	0.00	1.00	0.00	*Urtica*	*Urtica*
23	0.03	0.97	0.00	*Urtica*	*Urtica*
24	0.01	0.99	0.00	*Urtica*	*Urtica*
25	0.69	0.31	0.00	*Parietaria*	unknown
26	0.11	0.89	0.00	*Urtica*	*Urtica*
27	0.12	0.88	0.00	*Urtica*	*Urtica*
28	0.17	0.83	0.00	*Urtica*	*Urtica*
29	0.09	0.91	0.00	*Urtica*	*Urtica*
30	0.00	1.00	0.00	*Urtica*	*Urtica*
31	0.48	0.52	0.00	unknown	unknown
32	0.24	0.76	0.00	*Urtica*	*Urtica*
33	0.06	0.94	0.00	*Urtica*	*Urtica*
34	0.29	0.71	0.00	*Urtica*	*Urtica*
35	0.14	0.86	0.00	*Urtica*	*Urtica*
36	0.38	0.62	0.00	*Urtica*	unknown
37	0.06	0.94	0.00	*Urtica*	*Urtica*
38	0.55	0.45	0.00	unknown	unknown
39	0.01	0.99	0.00	*Urtica*	*Urtica*
40	0.00	1.00	0.00	*Urtica*	*Urtica*
41	0.00	1.00	0.00	*Urtica*	*Urtica*
42	0.02	0.98	0.00	*Urtica*	*Urtica*
43	0.03	0.97	0.00	*Urtica*	*Urtica*
44	0.21	0.79	0.00	*Urtica*	*Urtica*
45	0.02	0.98	0.00	*Urtica*	*Urtica*
46	0.00	1.00	0.00	*Urtica*	*Urtica*
47	0.01	0.99	0.00	*Urtica*	*Urtica*
48	0.79	0.20	0.07	*Parietaria*	*Parietaria*
49	0.54	0.46	0.00	unknown	unknown
50	0.01	0.99	0.00	*Urtica*	*Urtica*
51	0.00	1.00	0.00	*Urtica*	*Urtica*
52	0.01	0.99	0.00	*Urtica*	*Urtica*
53	0.01	0.99	0.00	*Urtica*	*Urtica*

54	0.00	1.00	0.00	*Urtica*	*Urtica*
55	0.00	1.00	0.00	*Urtica*	*Urtica*
56	0.00	1.00	0.00	*Urtica*	*Urtica*
57	0.02	0.98	0.00	*Urtica*	*Urtica*
58	0.00	1.00	0.00	*Urtica*	*Urtica*
59	0.54	0.46	0.00	unknown	unknown
60	0.45	0.55	0.00	unknown	unknown
61	0.09	0.91	0.00	*Urtica*	*Urtica*
62	0.00	1.00	0.00	*Urtica*	*Urtica*
63	0.00	1.00	0.00	*Urtica*	*Urtica*
64	0.00	1.00	0.00	*Urtica*	*Urtica*
65	0.06	0.94	0.00	*Urtica*	*Urtica*
66	0.05	0.95	0.00	*Urtica*	*Urtica*
67	0.01	0.99	0.00	*Urtica*	*Urtica*
68	0.23	0.77	0.00	*Urtica*	*Urtica*
69	0.21	0.79	0.00	*Urtica*	*Urtica*
70	0.72	0.28	0.00	*Parietaria*	*Parietaria*
71	0.49	0.51	0.00	unknown	unknown
72	0.06	0.94	0.00	*Urtica*	*Urtica*
73	0.33	0.67	0.00	*Urtica*	unknown
74	0.00	1.00	0.00	*Urtica*	*Urtica*
75	0.28	0.72	0.00	*Urtica*	*Urtica*
76	0.00	1.00	0.00	*Urtica*	*Urtica*
77	0.03	0.97	0.00	*Urtica*	*Urtica*
78	0.05	0.95	0.00	*Urtica*	*Urtica*
79	0.21	0.79	0.00	*Urtica*	*Urtica*
80	0.00	1.00	0.00	*Urtica*	*Urtica*
81	0.00	1.00	0.00	*Urtica*	*Urtica*
82	0.03	0.97	0.00	*Urtica*	*Urtica*
83	0.02	0.98	0.00	*Urtica*	*Urtica*
84	0.12	0.88	0.00	*Urtica*	*Urtica*
85	0.17	0.83	0.00	*Urtica*	*Urtica*
86	0.01	0.99	0.00	*Urtica*	*Urtica*
87	0.90	0.10	0.00	*Parietaria*	*Parietaria*
88	0.11	0.89	0.00	*Urtica*	*Urtica*
89	0.02	0.98	0.00	*Urtica*	*Urtica*

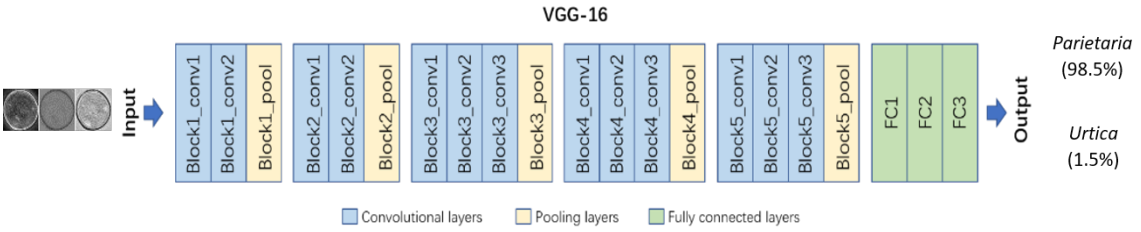
90	0.00	1.00	0.00	*Urtica*	*Urtica*
91	0.29	0.71	0.00	*Urtica*	*Urtica*
92	0.11	0.89	0.00	*Urtica*	*Urtica*
93	0.12	0.88	0.00	*Urtica*	*Urtica*
94	0.01	0.99	0.00	*Urtica*	*Urtica*
95	0.01	0.99	0.00	*Urtica*	*Urtica*
96	0.00	1.00	0.00	*Urtica*	*Urtica*
97	0.00	1.00	0.00	*Urtica*	*Urtica*
98	0.02	0.98	0.00	*Urtica*	*Urtica*
99	0.00	1.00	0.00	*Urtica*	*Urtica*
100	0.21	0.79	0.00	*Urtica*	*Urtica*
101	0.55	0.45	0.00	unknown	unknown
102	0.00	1.00	0.00	*Urtica*	*Urtica*
103	0.48	0.52	0.00	unknown	unknown
104	0.00	1.00	0.00	*Urtica*	*Urtica*
105	0.00	1.00	0.00	*Urtica*	*Urtica*
106	0.57	0.43	0.00	unknown	unknown
107	0.11	0.89	0.00	*Urtica*	*Urtica*
108	0.23	0.77	0.00	*Urtica*	*Urtica*
109	0.26	0.74	0.00	*Urtica*	*Urtica*
110	0.12	0.88	0.00	*Urtica*	*Urtica*
111	0.58	0.42	0.00	unknown	unknown
112	0.00	1.00	0.00	*Urtica*	*Urtica*



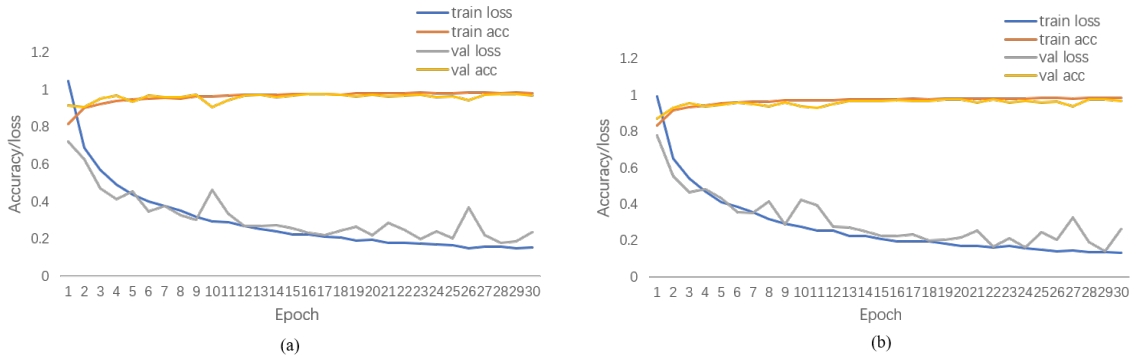
Supplementary Fig. S1. (a) Global native (green) and introduced (purple) distribution of *Parietaria judaica* and *P. officinalis* (POWO (2019). "Plants of the World Online. Map taken from the Royal Botanic Gardens, Kew. http://www.plantsoftheworldonline.org/ Retrieved 05 October 2020"). (b) Trend in Pellitory of the wall (*Parietaria judaica*) plant sightings per square kilometre in the Netherlands over the past 45 years. Index number = 100 for 1990 © NEM (CBS FLORON) 2019.



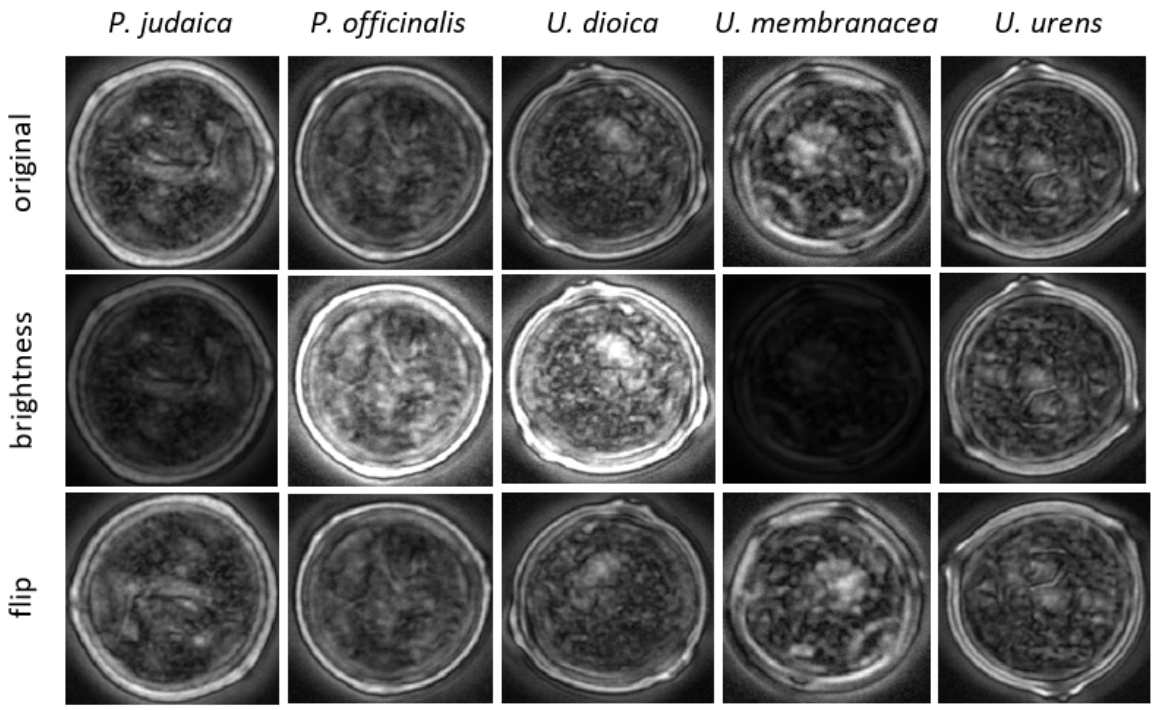
Supplementary Fig. S2. Pollen image acquisition and processing workflow carried out with in-house designed Pollen_Projector script. Once raw images are obtained at 20 different focal levels ('Z-slices'), subsequent steps involve cropping of whole individual pollen grains and producing three different projections from the Z-stacks. Abbreviations of projections: STD = Standard Deviation, MIN = Minimum Intensity and EXT = Extended Focus.



Supplementary Fig. S3. Schematic overview of the structure of VGG-16 with an example of three-channel input image of a *Parietaria judaica* pollen grain (known label) and the output generated, where it confidently identifies the images as *Parietaria* (98% probability). Adapted from Simonyan et al., (2014)[1]



Supplementary Fig. S4. Figures showing the accuracy/loss plots for the VGG16 model with 5- and 10-fold cross-validation.



Supplementary Fig. S5. Examples of data augmentation on the Standard Deviation Projection (STD) of selected pollen grains of all Urticaceae pollen species used in this study.

Appendix B

Supplementary Materials in Chapter 3

Supplementary Table S1 The classification performance of deep learning models on 5-species dataset. Standard deviation, training each model three times, is given in brackets.

	Precision	**Recall**	**F1 score**
VGG16	0.978(±0.001)	0.977(±0.001)	0.977(±0.001)
VGG19	0.974(±0.002)	0.973(±0.002)	0.973(±0.002)
ResNet50	0.980(±0.001)	0.979(±0.001)	0.979(±0.001)
MobileNet V1	0.958(±0.003)	0.957(±0.003)	0.957(±0.003)
MobileNet V2	0.970(±0.006)	0.970(±0.006)	0.970(±0.006)

Supplementary Table S2 The details of the dataset used in our study.

Classes	**Species**	**The number of images**	**The number of individual plants**
Parietaria	*Parietaria Judaica*	1670	8
	Parietaria officinalis	1359	6
Urtica	*Urtica dioica*	1055	5
	Urtica urens	1270	7
Urtica membranacea	*Urtica membranacea*	1118	4
Total images/individual plants were analyzed		6472	30

Supplementary Table S3 The final size of feature vector after feature selection/reduction.

Classifier	Feature selection/reduction	The final size after feature selection/reduction
SVM	PCA (0.8)	179×1
RF	SelectFromModel (mean)	2064×1
MLP	PCA (0.85)	337×1
Adaboost	Mutual information (2000)	2000×1

Supplementary Table S4 The number of images of predicted results for Urticaceae pollen grains scanned from aerobiological samples using the VGG16 and ResNet50 model with 5-fold cross-validation. *U. mem* = *Urtica membranacea*.

Location	**Leiden, the Netherlands**				**Lleida, Spain**			
Class labels	Label 1: *Parietaria*	Label 2: *Urtica*	Label 3: *U. mem*	unknown	Label 1: *Parietaria*	Label 2: *Urtica*	Label 3: *U. mem*	unknown
VGG16	5	96	0	11	51	9	0	3
ResNet50	8	97	0	7	60	3	0	0

Supplementary Table S5 Average performance comparison of different models on 5-fold and 10-fold cross-validation subset selection, respectively. Standard deviation of five/ten subsets, is given in brackets.

	VGG16	VGG19	ResNet50	Flat model	Hierarchical model
Average performance of 5-fold cross-validation selection with Standard Deviation	0.958 (±0.007)	0.958 (±0.007)	0.975 (±0.002)	0.867 (±0.012)	0.891 (±0.023)
Average performance of 10-fold cross-validation selection with Standard Deviation	0.940 (±0.008)	0.938 (±0.008)	0.955 (±0.010)	0.840 (±0.019)	0.873 (±0.030)

Supplementary Table S6 Ablation study with ResNet50. The average performance of ResNet50 based on ten (about) 500-images subsets via 10-fold cross-validation selection method is given. Standard deviation, of ten subsets, is given in brackets. Numbers in italics refer to training without transfer learning and data augmentation, respectively.

	Training from scratch	With/*without* transfer learning	With/*without* data augmentation	With hard voting
Accuracy	0.793(±0.034)	0.919(±0.019)	0.933(±0.011)	0.955(±0.010)
	0.793(±0.034)	0.919(±0.019)	*0.919(±0.019)*	0.939(±0.024)
	0.793(±0.034)	*0.793(±0.034)*	0.804(±0.032)	0.841(±0.033)

Summary

Microscopy supports us in observing small objects like cells, tissues, and micro-organisms. In Bio-Imaging images are captured to study biological phenomena. Often large collections of images are captured so that these images can be analysed in order to understand a phenomenon. In order to do these analyses, we need computational methods so that we can generate knowledge from these images. In this manner, imaging can accelerate studies in the biomedical domain. In this thesis, we have studied cell images from which analysis gives insight in the understanding of the content of these images. These are pollen grains and immune cells. For these cells the image analysis comprised analysis of 2D and 3D images. Moreover, the dynamics is considered in time-lapse sequences. The analysis of the pollen images focusses on the classification of the pollen whereas the analysis of immune cells focusses on the dynamics of neutrophils through tracking in time-lapse sequences. The approach that is required for these analyses is based on techniques from artificial intelligence in which computers are trained to learn recognizing specific patterns.

Pollen images are often made from airborne samples. From these samples different types of pollen can be recognised by a specialist. Sometimes it is very difficult, if not impossible to tell different species apart. In our study, we address a problem where this is typically the case and attempt to distinguish the allergenic pollen *Parietaria* from the non-allergenic pollen *Urtica*, and *Urtica membranacea* in the same Urticaceae family. We use images from pollen to see if we can train a computer to tell these species apart. In order to have sufficient detail of the pollen we capture a pollen grain in 20 optical slices and use a projection method from these images. For each pollen image, we compute three different projections to obtain images as input for a classification model. In this manner, we aim to integrate as many representative features as possible. Moreover, with these three projections as input images, the performance of the classification has proven to increase.

The amount of images was sufficient to probe deep learning-based classification models. Initially, we have trained three of these models to categorize the three pollen species. These are VGG16, MobileNet V1 and MobileNet V2. To further explore possibilities for deep learning, we extended our studies with extra three deep-learning models. These are AlexNet,

VGG19 and ResNet50. In daily practice, it is not always guaranteed that an abundant amount of samples is available. Therefore, in addition to the deep learning methods, we investigated the performance of traditional machine learning-based classification models such as Support Vector Machine (SVM) and Random Forest (RF). In convolutional neural networks (CNN), features are automatically extracted and used in the training of the model. For traditional machine learning we have to design a set of handcrafted features to train a model. Feature selection methods have been adopted to reduce the unrelated features. After feature selection, different flat classification models were constructed, respectively, to identify each category. In addition, a hierarchical strategy was conducted by categorizing all classes into subgroups. For each subgroup, different classifiers were trained separately. We have demonstrated that this hierarchical strategy improves the performance compared to flat models.

From the implementation of different classification models for both deep learning-based and traditional machine learning-based methods, we conclude that: compared to machine learning-based methods, deep learning-based models achieve higher classification accuracy, in particular, i.e. ResNet50. The experimental results on both large and small datasets have proven the robustness of deep learning models. An ablation study explains the success of deep learning models by integrating different techniques, i.e. data augmentation, transfer learning and hard voting. We have elaborated and implemented a workflow to find a solution to accurately classify those high-similar pollen categories. This study provides a good insight in complex classification of pollen thereby allowing healthcare professionals to better advise hay fever patients with their treatments and outdoor activity planning.

Next to pollen imagery, we studied a completely different domain, in which we can also easily obtain images in different dimensions. We address the monitoring of immune cells in response to a possible infection. So, here we study life-images of cells while they are moving in a body; i.e. neutrophils in a zebrafish larvae. We monitor these neutrophils by tracking them in time-lapse sequences. Neutrophils are involved in the first line of defense against pathogens to protect the organism, in our case a tail wound in a zebrafish larvae. As part of the understanding of the cell behavior of the immune system, the monitoring of the migration of neutrophils *in vivo* is important. Neutrophil tracking, however, is a challenging problem. In our studies, we have analysed time-lapse sequences of neutrophils that have been imaged with a confocal microscope. From the time-lapse sequences computational methodologies and algorithms are designed with the aim to solve the tracking problems in both 2D and 3D spatial domains.

For the neutrophil tracking in a 2D time-lapse sequence, we propose a pipeline involving cell segmentation, cell motion tracking between adjacent frames, and trajectory linkage. The neutrophils are fluorescently labelled and a first step for the identification is a segmentation

of these cells. For segmentation of cells we use three U-Net-based segmentation models and a rule-based Watershed Masked Clustering (WMC) segmentation algorithm. The models are constructed and compared for performance with respect to accurate cell location for the tracking procedure. We intend to select the best performing segmentation model. Subsequently, from the segmentation, the cell migration motion needs to be understood. Therefore, we construct an additional U-Net model that is trained to learn the cell migration motion from annotated ground truth data. In this manner, we can predict the cell position in the next frame. In order to obtain cell trajectories, we need to solve the challenging cell merging/splitting problems that arise from the sequences. To that end, we have made a key contribution in that we designed an extension to the Viterbi linkage algorithm to be able to objectively solve this. Our pipeline has achieved satisfying performance on the neutrophil tracking task.

After our accomplishments in 2D time-lapse sequences, we investigated neutrophil tracking in 3D time-lapse sequences. We designed and implemented a pipeline including 3D cell segmentation, a feature-weighted similarity estimation and trajectory linkage. We select a 3D U-Net deep learning model to segment 3D cell images at each time point. A similarity estimation is done by extracting cell features, i.e. cell travel distance, direction of cell movement and average cell travel distance over trajectory. Next, a similarity score is calculated by weighting these features at different scales. Based on the similarity score, we adapted the Hungarian algorithm to link the cell trajectories. Overall, compared to the existing state-of-the-art methods, our method demonstrates a good performance according to our experimental results.

In this thesis, we have demonstrated the use of deep learning methods for different kinds of cell imagery and shown the potential for future use whist illustrating pitfalls, but also presenting efficient new algorithmic approaches.

Nederlandse Samenvatting

Microscopie ondersteunt ons bij het observeren van kleine objecten zoals cellen, weefsels en micro-organismen. In het werkveld van BioImaging worden beelden opgenomen om biologische fenomenen te bestuderen. Vaak worden grote verzamelingen beelden opgenomen zodat deze beelden kunnen worden geanalyseerd om een fenomeen te begrijpen. Om deze analyses te kunnen doen zijn computationele methoden nodig waarmee we kennis uit de beelden kunnen genereren. Imaging kan, met deze werkwijze, studies in het biomedische domein versnellen. In dit proefschrift hebben we beelden van cellen bestudeerd waarbij de analyse inzicht verschaft in begrip van de inhoud van deze beelden. We hebben in het bijzonder naar beelden van pollenkorrels en immuuncellen gekeken. Bij deze cellen betrof de beeldanalyse van zowel 2D als 3D beelden. Daarenboven is gekeken naar dynamiek uit beeldreeksen. De beeldanalyse van de pollenkorrels heeft een focus op classificatie van de pollen terwijl de aandacht bij de analyse van de immuuncellen vooral gaat naar de dynamiek van neutrofielen door deze te volgen in een tijdinterval opgenomen videosequentie. De aanpak die nodig is voor deze analyses berust op technieken uit de kunstmatige intelligentie, waar computers leren om bepaalde patronen te herkennen.

Beelden van pollen worden vaak gemaakt van uit de lucht genomen monsters. In beelden van deze luchtmonsters kan een specialist verschillende soorten pollen herkennen. Echter, soms is het moeilijk, zo niet onmogelijk om sommige soorten uit elkaar te houden. In onze studie hebben we ons beziggehouden met soorten waar dit typisch het geval is. We doen daarmee een poging om allergeen pollen van Parietaria van niet-allergeen pollen van Urtica en Urtica membranacea, allen uit dezelfde Urticaceae familie, te kunnen onderscheiden. We gebruiken beelden van pollen om na te gaan of we een computer(-programma) kunnen leren de soorten uit elkaar te houden. Om voldoende detail in de beelden te verkrijgen, maken we gebruik van de mogelijkheid om de pollenkorrel in 20 parallelle optische aanzichten in beeld vast te leggen, en uit die 20 aanzichten een projectie te genereren in een beeld. Van ieder pollenbeeld verkrijgen we 3 verschillende projecties die we als invoerbeelden gebruiken voor een classificatiemodel. Langs deze weg proberen we zoveel mogelijk representatieve

kenmerken van pollen te integreren. Bovendien is gebleken dat met deze 3 types van projecties, de prestatie van de classificatie aanzienlijk verbeterd.

De hoeveelheid beelden die we beschikbaar hebben is voldoende om zogenaamde "deep learning" leermodellen te gebruiken. Eerst hebben we dit type modellen getraind om drie soorten pollen te classificeren. De modellen die we hiervoor hebben gebruikt zijn: VGG16, MobileNet V1 en MobileNet V2. Teneinde de mogelijkheden van "deep learning" verder te exploreren, hebben we onze studie uitgebreid met nog 3 andere "deep learning" modellen. Deze zijn: AlexNet, VGG19 en ResNet50. In de dagelijkse praktijk is het echter niet altijd gegarandeerd dat er een grote hoeveelheid monsters van pollen, in beeldvorm, aanwezig is. Daarom hebben we naast de "deep learning" methoden, ook de prestaties van classificatie modellen gebaseerd op traditioneel machinaal leren; zoals bijvoorbeeld de Support Vector Machine (SVM) en Random Forest (RF). In convolutionele neurale netwerken (CNN), worden kenmerken automatisch geëxtraheerd en gebruikt in een model voor training. Voor traditioneel machinaal leren moeten we zelf de kenmerken ontwerpen, zogenaamde "handcrafted" kenmerken, om het model te trainen. Methoden voor kenmerkselectie zijn overgenomen om het aantal ongerelateerde kenmerken te reduceren. Na het toepassen van kenmerkselectie zijn verschillende rechttoe-rechtaan modellen gemaakt om iedere pollencategorie te herkennen. Daarenboven is er een hiërarchische classificatie-strategie toegepast door eerst de verschillende klassen in subgroepen te categoriseren. Voor iedere subgroep worden de verschillende classificatiemodellen dan afzonderlijk getraind. Met deze experimenten hebben we laten zien dat de prestatie verbeterd door gebruik te maken van de hiërarchische strategie.

Van de implementatie van verschillende classificatie modellen, zowel gebaseerd op "deep learning" als traditioneel machinaal leren, concluderen we dat in vergelijking met machinaal leren, "deep learning" methoden een hogere nauwkeurigheid in de classificatie bereiken, ResNet 50 in het bijzonder. Onze experimentele resultaten, op zowel grote als kleine datasets, hebben de robuustheid van de "deep learning" methoden laten zien. Een ablatie studie demonstreert het succes van "deep learning" modellen door verschillende technieken, zoals data-augmentatie, "transfer learning" en "hard voting" te integreren. We hebben een werkwijze uitgewerkt en geïmplementeerd om een oplossing te vinden voor het nauwkeurig kunnen classificeren van zeer gelijkende pollencategorieën. Deze studie geeft een goed inzicht in de complexe classificatie van pollen en geeft gezondheidsdiensten de mogelijkheid om hooikoortspatiënten beter te adviseren voor hun behandeling en plannen van activiteiten buitenshuis.

Naast het verwerken van beelden van pollen, hebben we nog een compleet verschillend domein bestudeerd waar we eveneens op gemakkelijke weg beelden in meerder dimensies

kunnen verkrijgen. We bestuderen het volgen van immune cellen in hun respons op een mogelijke infectie. We kijken daarom naar live-beelden van cellen zoals ze door het lichaam bewegen; i.e. neutrofielen in de larve van een zebravis. We monitoren deze neutrofielen door ze te volgen in een videosequentie die in vaste tijdintervallen is opgenomen (time-lapse). Het volgen van een subject in een videosequentie wordt "tracking" genoemd. De neutrofielen zijn betrokken in de eerstelijnsverdediging bij het beschermen van een organisme tegen een pathogeen; in het geval van onze studie betreft het een verwonding aan de staartvin van de larve van een zebravis. Het monitoren van de migratie van neutrofielen is een belangrijk onderdeel voor het verkrijgen van begrip van het gedrag van de cellen van het immuunsysteem. Echter, het volgen (tracking) van neutrofielen is een uitdaging. In onze studie hebben we tijdreeks-sequenties van neutrofielen geanalyseerd die zijn opgenomen met een Confocal Laser Scanning Microscope (CLSM). Met de tijdreeks-sequenties als uitgangspunt zijn computationele methoden en algoritmen ontworpen met het specifieke doel het "tracking" probleem in zowel het 2D als het 3D domein op te kunnen lossen.

Voor het volgen van neutrofielen in een 2D tijdreeks videosequentie, stellen we een protocol voor waarin opgenomen de segmentatie van de cel, het volgen van de bewegingen van de cel tussen opeenvolgende "frames" en het aaneenrijgen van de cel uit de tijdreeks-sequenties in een traject over de tijd. De neutrofielen zijn fluorescent gelabeld en de eerste stap in het identificeren van deze cellen door is een segmentatie. Voor deze segmentatie gebruiken we U-Net gebaseerde segmentatie modellen en een algoritme dat is gebaseerd op regels, nl. Watershed Masked Clustering (WMC). Deze twee methoden voor segmentatie zijn geïmplementeerd en hun prestaties zijn onderling vergeleken met betrekking tot accurate cel-lokalisatie voor de tracking procedure. Het is onze bedoeling het best presterende segmentatiemodel op te nemen in ons protocol. Vanuit de resultaten van de segmentatie moet, vervolgens de cel beweging kunnen worden begrepen. Daartoe hebben we een extra U-net model geconstrueerd dat is getraind om cel beweging en migratie te leren uit geannoteerde grondwaarheid data. Op deze manier kunnen we de positie van een cel in een volgend frame van de tijdreeks-sequentie voorspellen. Om vervolgens trajecten van cellen te verkrijgen moeten we ons buigen over de uitdagende taak van het ontrafelen van cellen in de tijdreeksen-beelden die zich lijken samen te voegen of op te splitsen. We hebben hier een belangrijke bijdrage geleverd die bestaat uit een uitbreiding van het "Viterbi Linkage Algoritme" waarmee de ambigue situaties objectief kunnen worden opgelost. Met deze drie componenten levert ons protocol bevredigende prestaties voor het volgen van neutrofielen *in vivo*.

In navolging van onze resultaten in de analyse van 2D video-sequenties, hebben we het volgen van neutrofielen in 3D tijdreeks-beelden onderzocht. We hebben een protocol ontworpen waarin opgenomen 3D cel segmentatie, een gewogen kenmerk-similariteitschatting

en wederom het linken van de cel-posities in een traject (trajectory linkage). Voor de 3D celsegmentatie hebben we een 3D U-Net model geselecteerd, om zo voor ieder punt in de tijdreeks een segmentatie van een cel te krijgen. Een similariteitschatting wordt gedaan door een aantal kenmerken van de cel te extraheren: i.e. afgelegde cel afstand, bewegingsrichting van de cel en de gemiddelde afstand over het afgelegde traject. Door deze kenmerken op verschillende schaal te wegen wordt er een similariteitscore berekend. Op basis van deze similariteitscore hebben we het zogenaamde Hongaarse Algoritme aangepast om de cel-posities te linken in trajecten van de cellen. In vergelijk met bestaande moderne methoden, kunnen we concluderen dat ons protocol, gegeven onze experimentele resultaten, een goede prestatie levert.

In dit proefschrift hebben we gedemonstreerd hoe “deep learning” methoden kunnen worden toegepast in uiteenlopende taken voor de analyse van beelden van cellen en daarbij een inkijk gegeven in de potentie van deze methoden voor de toekomst. We illustreren de valkuilen maar presenteren eveneens nieuwe en efficiënte algoritmische benaderingswijzen.

Curriculum Vitae

Chen Li was born in Shandong, China, in 1992. In 2011, she began her bachelor's studies in Information Engineering at Ludong University, located near the east coast of Shandong province, China. In 2015, Chen pursued a master's degree in computer science at China Agricultural University in Beijing, China. During her three-year master's program, she focused on data prediction models and developed an interest in computer vision and image processing. She earned her master's degree in 2018. Aligned with her research interests in computer vision, Chen joined the Computational Bio-imaging group at Leiden University, supervised by Prof. Dr. Ir. Fons J. Verbeek. Her four-year study, supported by the Chinese Scholarship Council (CSC), involved cell image analysis in both 2D and 3D domains, as well as time-lapse data analysis. She completed projects in collaboration with Naturalis and the Institute of Biology (IBL) of Leiden University.

Acknowledgements

On 3rd September 2018, my flight arrived at Schipol Airport, in the Netherlands. The journey of my Ph.D. study at Leiden University started. Time flies. Five and a half years have passed since I was here. During this period, I experienced challenges with aspects of language, culture, research, etc. It was sometimes frustrating, however, it was full of excitement and joy in most cases. For me, conquering difficulties, or sharing my achievements, would not have been possible without the support and guidance from my supervisors, friends, colleagues, and family members. Thus, I would like to express my appreciation to them.

First of all, I would like to express my gratitude to my supervisor, Prof. Fons J. Verbeek, who gave me the opportunity be part of the Bio-imaging group for my research. It is my great honor to work with you throughout my doctoral studies. What I have learned is not only the specialized knowledge in the aspects of bioimaging techniques, microscopy techniques, cells, and computational methods but also the spirit of being an independent academic researcher. Thank you for your supervision and insightful suggestions during my doctoral studies. It will benefit me in my next career and life.

I would like to express my appreciation to my daily supervisor, Dr. Lu Cao. She has been a guide who always helps me with my research, and she is also like an older sister who encourages and supports me in overcoming difficulties. During the last period of my Ph.D. journey, which was filled with the pressures of graduation and job-hunting, she cared about me, comforted me, and gave me many helpful suggestions to help me escape from stress, emotions, and frustrations. I greatly appreciate everything she has done for me.

Dr. Erwin van Soest, who holds a doctoral degree in computer science and two master's degrees in computer science and astronomy, has been the smartest and nicest roommate I could wish for. From him, I have learned many aspects of physics, astronomy, history, and more. He made me realize that there are people in this world with vast amounts of knowledge. Moreover, as a Dutch native, he has greatly assisted me in living in the Netherlands. I cannot count how many Dutch letters he helped me read, how many times he accompanied me to hospitals, repair shops, and other places, and how often he helped communicate with people from city hall, the GP, NS company, and others via phone calls. Thanks to his warm help, my

life in the Netherlands has been easy and cozy. Erwin, I promise I will miss you no matter where I am in China.

In addition, I would like to thank my other roommate Feibo Duan and my Chinese colleagues: Xiaoqin Tang, Zhan Xiong, Yi Chen, Xue Wang, Jia Li, Danyi Liu, Jingmin Long, Bei Chen, and Wanbin Hu. As PhD candidates in our group, we have often discussed research questions and helped each other with living in the Netherlands. This has made the experience memorable for all of us. Furthermore, I would like to thank my friends Jia Zhang, Wenjing Chu, and Qihui Liang. Whether during the COVID pandemic or in normal times, we have accompanied each other, gone shopping, traveled, cooked, and had meals together. I wish all of you great success in your future careers.

I would also like to thank Marloes van der Nat for her help with my studies at Leiden University. She has helped me solve many problems regarding courses, residence permits, graduation logistics, and more. Additionally, many thanks to my other colleagues in the Bio-imaging group: Katy Wolstencroft, Shima Javanmardi, Erick Paulus, Solomiia Kurchaba, Ana Arcos, Mariam Basajja, Irene Martorelli, and Sacha Goultiaev, for being there for me.

The great gratitude goes to my parents for their financial support, encouragement, and unconditional love. Last but not least, I would like to show my appreciation to my boyfriend Dr. Bing Wang. The time for us being together was short, however, we are still much closer by sharing our daily life even academic life. We help each other stay strong. Thanks for your love, company, encouragement, and support during these years. We were far apart, but we persevered. The best is yet to come.

Chen Li
May 2024
Leiden, the Netherlands