Detection and Differentiation of Circulating Tumor Cells in Liquid Biopsy using Self-Supervised Learning and Human-in-the-Loop Methods

Dissertation
zur Erlangung des Doktorgrades
an der Fakultät für Mathematik,
Informatik und Naturwissenschaften

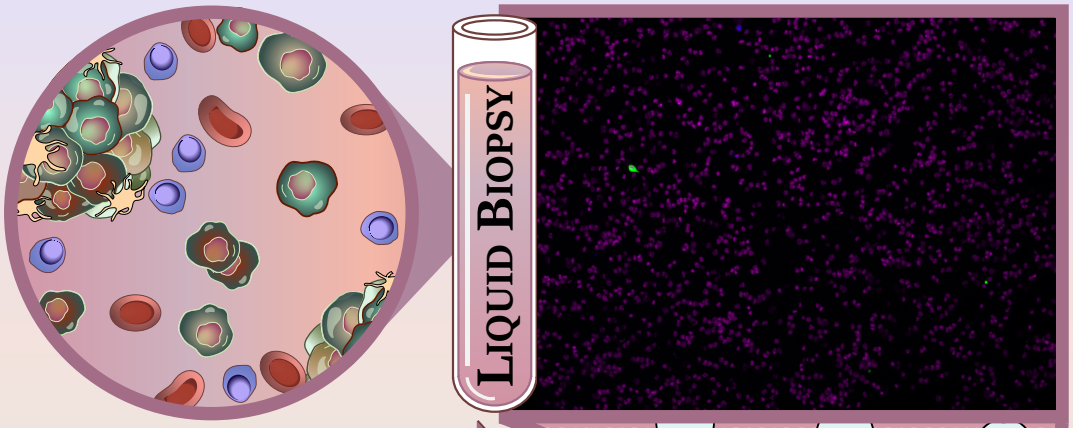
Fachbereich Biologie
der
Universität Hamburg

vorgelegt von
Hümeyra Husseini-Wüsthoff (geb. Husseini)

Hamburg
2025

Gutachter der Dissertation:	Prof. Dr. Arp Schnittger Prof. Dr. René Werner
Zusammensetzung der Prüfungskommission:	Prof. Dr. Arp Schnittger Prof. Dr. René Werner Prof. Dr. Julia Kehr Prof. Dr. med. Klaus Pantel
Vorsitzender der Prüfungskommission:	Prof. Dr. Julia Kehr
Datum der Disputation:	30.01.2026
Vorsitzender Fach-Promotionsausschusses Biologie:	Prof. Dr. Wolfgang Streit
Leiter des Fachbereichs Biologie:	Prof. Dr. Stefan Hoth
Dekan der Fakultät MIN:	Prof. Dr.-Ing. Norbert Ritter

Detection and Differentiation of Circulating Tumor Cells in



using Self-Supervised Learning and Human-In-The-Loop Methods

Für Anne und Baba

Danksagung

Nach vier Jahren intensiver Forschungsarbeit habe ich nun den Abschluss meiner Dissertation am Universitätsklinikum Hamburg-Eppendorf (UKE) erreicht. Ich möchte diesen besonderen Moment nutzen, mich von Herzen bei all jenen zu bedanken, die mich auf diesem Weg begleitet und zum Gelingen dieser Arbeit beigetragen haben.

Mein aufrichtiger Dank gilt in erster Linie meinem äußerst kompetenten Doktorvater Prof. Dr. René Werner: *Lieber René, bereits seit meiner Masterarbeit hast Du mich nicht nur fachlich, sondern auch persönlich mit großem Engagement begleitet. Deine offene Art, Deine verlässliche Unterstützung und Deine beständige Ermutigung haben meine Entwicklung entscheidend geprägt und waren für mich von unschätzbarem Wert.*

Ebenso danke ich herzlich meinem Zweitgutachter Prof. Dr. Arp Schnittger, den ich bereits seit meinem Biologiestudium kenne und der mich als Co-Betreuer in der Masterarbeit unterstützt hat: *Lieber Arp, ich habe Deine Unterstützung immer sehr geschätzt.*

Ein besonderer Dank gebührt auch den engagierten Kooperationspartner:innen der Tumorbiologie am UKE, Prof. Sabine Riethdorf, Prof. Harriet Wikmann-Kocher und Prof. Klaus Pantel, ohne deren Expertise, Hilfe und Begeisterung dieses interdisziplinäre Projekt nie in dieser Form hätte realisiert werden können. Besonders hervorheben möchte ich: *Liebe Sabine, vielen Dank für die tolle Zusammenarbeit und die fachlichen Diskussionen. Dein wertvoller Beitrag und Deine Bereitschaft, Dich voll einzubringen, haben unser gemeinsames Projekt maßgeblich bereichert.*

Ebenso möchte ich allen herzlich danken, die dazu beigetragen haben, dass ich während meiner Promotion durch die Erich und Gertrud Roggenbuck-Stiftung gefördert werden konnte. Mein besonderer Dank gilt auch der Roggenbuck-Stiftung selbst für die Unterstützung meiner wissenschaftlichen Arbeit.

Darüber hinaus gilt mein großer Dank all meinen Kolleg:innen, die mich während der Promotion begleitet und unterstützt haben. Für die stets angenehme Arbeitsatmosphäre danke ich insbesondere meinen Büropartnern Lukas Wimmert und Johannes Gebauer. Hervorheben möchte ich außerdem Maximilian Nielsen: *Lieber Max, Dein wertvoller, insbesondere methodischer Beitrag war für meine Arbeit von großer Bedeutung. Die Zusammenarbeit mit Dir hat mir nicht nur viel Freude bereitet, sondern mir als Biologin auch die Möglichkeit gegeben, viele neue methodische Einblicke zu gewinnen und meinen Horizont deutlich zu erweitern.*

Ein besonderer Dank gilt außerdem der Studienstiftung des deutschen Volkes für ihre langjährige Unterstützung. Bereits während meines Studiums und später als Promotionsstipendiatin durfte ich von der Förderung profitieren.

Mein Dank richtet sich auch an die verschiedenen Verlage, Zeitschriften und Autor:innen, die mir die Rechte zur Nutzung von Abbildungen in dieser Dissertation gewährt haben. Sämtliche urheberrechtlichen Hinweise zu den verwendeten Abbildungen finden sich in den entsprechenden Abbildungsbeschreibungen.

Darüber hinaus danke ich meinen Freundinnen, die mir stets mit Rücksicht, Verständnis und liebevoller Unterstützung begegnet sind, auch wenn ich aufgrund der Promotion manchmal anderes priorisieren musste.

Zuletzt möchte ich mich von ganzem Herzen bei meiner Familie bedanken: *Liebe Eltern, Ihr habt einen langen, weiten und nicht immer leichten Weg hinter Euch; ich wünschte, Ihr hättet die gleichen Chancen und eine ebenso liebevolle Unterstützung erfahren wie ich. Liebe Eltern, lieber Sofian - Euer unerschütterlicher Glaube an mich, Eure Liebe, Eure ständige Ermutigung und Eure bedingungslose Unterstützung haben mir diesen Weg eröffnet. Lieber Sofian, ich danke Dir, dass Du mir jeden Tag nach der Arbeit zugehört, geholfen und mir immer wieder Mut zugesprochen hast. Liebe Sousou, auch Dir möchte ich von Herzen danken. Deine emotionale und fachliche Unterstützung ist unersetzlich. Ich wünsche Dir, dass Du deine Promotion bald erfolgreich abschließt!*

Abstract

Over the past decade, advances in liquid biopsy - which involves analyzing blood samples for tumor cells or their products - have contributed to significant progress in cancer diagnostics and precision medicine. Among the most prominent biomarkers are circulating tumor cells (CTCs), which are cancer cells present in the bloodstream that play a crucial role in the metastatic process. Numerous studies using the CellSearch® (CS) system, which has been approved by the U.S. Food and Drug Administration (FDA) for clinical application in metastatic breast, colorectal, and prostate cancer, focus particularly on the enumeration of CTCs, as the number of CTCs serve as a strong predictor of patients' progression-free and overall survival. For instance, the detection of five or more CTCs per 7.5 mL of blood in metastatic breast cancer patients correlates with poor clinical outcomes.

Despite their clinical relevance, CTCs are rare and highly heterogeneous, presenting considerable challenges for their detection. Currently, the identification of CTCs in peripheral blood samples by CS remains only partially automated, highlighting the need for fully automated solutions. Although recent deep learning (DL)-based methods offer promising advances, they typically require large annotated data sets, which are expensive and often impractical to obtain, whereas abundant unlabeled data remain largely unused. Furthermore, the shift toward full automation increases the risk of overlooking model uncertainties, which need to be addressed.

Using fluorescence microscopy images acquired by the CS system from a cohort of metastatic breast cancer patients, the key contributions of this thesis are as follows:

(1) Application of a reliable DL-based segmentation method to detect CTCs.

(2) Demonstration that self-supervised learning using unlabeled image data enables the model to learn generalized representations, and, when combined with a supervised conventional machine learning classifier, allows for efficient and robust binary classification (CTCs vs. non-CTCs) using only a small set of well-annotated images.

(3) Development of a human-in-the-loop framework to identify and target model uncertainties during classification, enabling improved differentiation between cell classes by having experts annotate an automatically selected, limited yet meaningful set of previously unlabeled images from regions of high model uncertainty.

(4) Comparative analysis of CTC detection and classification performance between the CS system and the optimized classifier from (3) on a metastatic cohort with progressive and stable disease, representing varying CTC burdens, as well as on healthy donors. Expert review of events and candidate cell images of both systems revealed that distinguishing between CTCs and non-CTCs can be challenging, leading to the definition of additional categories for unsure cells.

It is concluded that the methods and strategies presented here pave the way for more reliable, automated, and efficient CTC detection, while also providing initial approaches for addressing uncertainties arising from both the model and expert annotation.

Kurzfassung

In den vergangenen zehn Jahren haben Fortschritte in der Liquid Biopsy (Flüssigbiopsie) - darunter die Blutprobenanalyse von Tumorzellen oder deren Produkte - zu signifikanten Entwicklungen in der Krebsdiagnostik und der Präzisionsmedizin geführt. Zu den wichtigsten Biomarkern zählen zirkulierende Tumorzellen (CTCs), bei denen es sich um Tumorzellen im Blutkreislauf handelt, die eine entscheidende Rolle im Metastasierungsprozess spielen. Zahlreiche Studien, die das von der US-amerikanischen Food and Drug Administration (FDA) für die klinische Anwendung bei metastasiertem Brust-, Kolorektal- und Prostatakrebs zugelassene CellSearch® (CS)-System verwenden, legen den Fokus besonders auf die Zählung von CTCs, da die CTC Anzahl als starker Prädiktor für das progressionsfreie und das Gesamtüberleben der Patient:innen gilt. So korreliert beispielsweise der Nachweis von fünf oder mehr CTCs pro 7,5 ml Blutprobe bei Patient:innen mit metastasiertem Brustkrebs mit einer ungünstigen Prognose.

Trotz ihrer klinischen Relevanz sind CTCs im Vergleich zu anderen Blutzellen selten und zeichnen sich durch eine hohe Heterogenität aus, was ihre Detektion erheblich erschwert. Die Identifikation von CTCs in peripheren Blutproben mittels CS-System ist derzeit nur teilweise automatisiert, wodurch der Bedarf an vollständig automatisierten Lösungen besonders hoch ist. Zwar zeigen aktuelle Deep Learning (DL)-basierte Verfahren vielversprechende Fortschritte, diese erfordern jedoch meist große Mengen annotierter Datensätze, deren Erstellung aufwendig und oftmals nicht praktikabel ist, wohingegen zahlreiche nicht annotierte Daten oftmals ungenutzt bleiben. Darüber hinaus birgt die vollständige Automatisierung das Risiko, Unsicherheiten des Modells zu übersehen, weshalb diese gezielt adressiert werden müssen.

Auf Basis von Fluoreszenzmikroskopie-Bildern, die mit dem CS-System von einer Kohorte metastasierter Brustkrebspatient:innen aufgenommen wurden, ergeben sich die folgenden zentralen Beiträge dieser Arbeit:

(1) Die Anwendung eines zuverlässigen, DL-basierten Segmentierungsalgorithmus zur Detektion von CTCs.

(2) Die Anwendbarkeit von selbstüberwachtem Lernen auf unannotierten Bilddaten im Bereich der Flüssigbiopsie. Dieser Ansatz ermöglichte es zunächst, generalisierte Repräsentationen der Bilder zu erlernen und, in Kombination mit einem nachfolgenden konventionellen Machine-Learning-Klassifikator, eine effiziente und robuste binäre Klassifikation (CTCs vs. Nicht-CTCs) mit nur einer kleinen Anzahl sorgfältig annotierter Bilder zu erzielen.

(3) Entwicklung eines Human-in-the-Loop-Frameworks zur Identifikation und Reduzierung von Modellunsicherheiten während der Klassifikation, indem gezielt und automatisiert eine begrenzte Anzahl zuvor nicht annotierter Bilder aus Bereichen hoher Modellunsicherheit von Expert:innen annotiert und dem Modell zur weiteren Verbesserung zurückgeführt wird.

(4) Vergleichsanalyse der Detektions- und Klassifikationsleistung für CTCs zwischen dem CS-System und dem optimierten Klassifikator aus (3) an einer metastasierten Kohorte mit progressiver und stabiler Erkrankung, die unterschiedliche CTC-Belastungen repräsentiert, sowie an gesunden Spender:innen. Die Überprüfung der Ereignisse und vorgeschlagenen Kandidatenzellen beider Systeme zeigte, dass die Unterscheidung zwischen CTCs und Nicht-CTCs selbst für Expert:innen mitunter schwierig ist, was zur Definition zusätzlicher Kategorien für unsichere Zellen führte.

Abschließend lässt sich sagen, dass die hier vorgestellten Methoden und Strategien den Weg für eine zuverlässigere, automatisierte und effizientere CTC-Detektion ebnen und zugleich erste Ansätze zur Adressierung von Unsicherheiten sowohl auf Modell- als auch auf Expertenseite liefern.

Content

List of Acronyms

ACCEPT	Automated CTC Classification, Enumeration, and PhenoTyping
Adam	adaptive momentum estimation
AF	activation function
AI	artificial intelligence
APC	allophycocyanin
(B)CE	(binary) cross-entropy
BYOL	Bootstrap Your Own Latent
CC BY	Creative Commons Attribution license
CD45	cluster of differentiation 45; leukocyte marker
cfRNA	circulating free ribonucleic acid
CK	cytokeratin
CNN	convolutional neural network
CS	CellSearch®
CTC	circulating tumor cell
ctDNA	circulating tumor deoxyribonucleic acid
CV	cross-validation
D	dimensional
DAPI	4',6-diamidino-2-phenylindole, dihydrochloride
DBSCAN	Density Based Spatial Clustering of Applications with Noise
DINO	self-distillation with no labels
DL	deep learning
DSB	Data Science Bowl

EDT	Euclidean Distance Transform
EMA	exponential moving average
EMT	epithelial-to-mesenchymal transition
EOM	excess of mass
EpCAM	epithelial cell adhesion molecule
Eq	equation
EV	extracellular vesicle
FDA	Food and Drug Administration
Fig	figure
FN	false negative
FP	false positive
GT	ground truth
HER2	human epidermal growth factor receptor 2
HDBSCAN	Hierarchical Density Based Spatial Clustering of Applications with Noise
HiL	human-in-the-loop
IoU	Intersection over Union
KNN	k-nearest neighbors
LB	liquid biopsy
LPI	Local Patch Interaction
LR	logistic regression
MAE	mean absolute error
MC	Monte Carlo
MET	mesenchymal-to-epithelial transition
MHC	major histocompatibility complex
ML	machine learning
MLP	multilayer perceptron
MoCo	Momentum Contrast

MSE	mean squarred error
NMS	non-maximum suppression
PCA	principal component analysis
PD-L1	programmed death-ligand 1
PE	phycoerythrin
RBF	radial basis function
ReLU	rectified linear unit
SimCLR	Simple Framework for Contrastive Learning of Visual Representations
SGD	stochastic gradient descent
SSL	self-supervised learning
SVM	support vector machine
TEP	tumor-educated platelet
TN	true negative
TP	true positive
t-SNE	t-distributed stochastic neighbour embedding
UKE	University Medical Center Hamburg-Eppendorf
UMAP	uniform manifold approximation and projection
ViT	Vision Transformer
WBC	white blood cell
XCA	Cross-Covariance Attention
XCiT	Cross-Covariance image Transformer

Chapter 1

Introduction

Cancer ranks among the top causes of death. According to global cancer statistics [Sun+21], nearly 10 million individuals died from cancer in 2020. As of 2022, it was estimated that approximately one in five people will have cancer during their lifetime, with about one in nine men and one in twelve women dying from the disease [Bra+24]. In 2022, breast cancer remained the most commonly diagnosed cancer among women, both in terms of incidence and mortality; in contrast, lung cancer predominated among men for both incidence and mortality [Bra+24]. Other significant cancer incidences in women included lung and colorectal cancer, while in men, prostate and colorectal cancers had the next highest incidence rates [Bra+24].

Although precision medicine remains an achievable goal in medical oncology, the substantial hurdle posed by tumor cell heterogeneity complicates treatment efforts. Tumor heterogeneity arises from various factors, including genetic and epigenetic changes, transcriptional and protein modifications, alterations in the microenvironment, and changes in cellular phenotypes [McQ+17]. It encompasses both temporal (progression over time) and spatial (variation across regions) aspects, each presenting major challenges for effective treatment. Temporal heterogeneity is characterized by the transition from localized to metastatic disease, which includes variations between the primary tumor and its metastatic sites within the same patient. For instance, a patient initially diagnosed with human epidermal growth factor receptor 2 (HER2)-negative primary breast cancer might later develop HER2-positive metastases [Lin+12], significantly impacting the patient's survival. Spatial heterogeneity, on the other hand, refers to the presence of distinct tumor clones, which can be found within a single primary or metastatic tumor (intratumoral), as well as among different tumors in the same patient (intrapatient) [McQ+17].

1.1. Cancer detection and classification in liquid biopsy

Diagnostic imaging and tissue biopsies are the gold standards for cancer detection, diagnosis, treatment decision-making, and response evaluation. However, these methods have limitations in terms of sensitivity and their ability to detect early-stage disease. In the metastatic setting, tissue biopsies, in particular, are highly invasive, posing discomfort and risk to patients, and are often expensive. Moreover, some anatomical locations

are difficult to access [Ma+24]. Since the deaths of cancer patients are often caused by metastases at distant sites, repeated biopsies would be necessary to sample tissue from both the primary and metastatic sites to monitor disease progression. However, the invasive nature of these procedures makes them unsuitable for continuous monitoring [DKB19].

To overcome some of the aforementioned limitations, liquid biopsy (LB) offers a promising alternative. LB is a minimally invasive method for collecting bodily fluids such as blood (see Fig. 1.1), allowing for the detection of cancer-derived material circulating in the bloodstream that is released from tumor lesions.

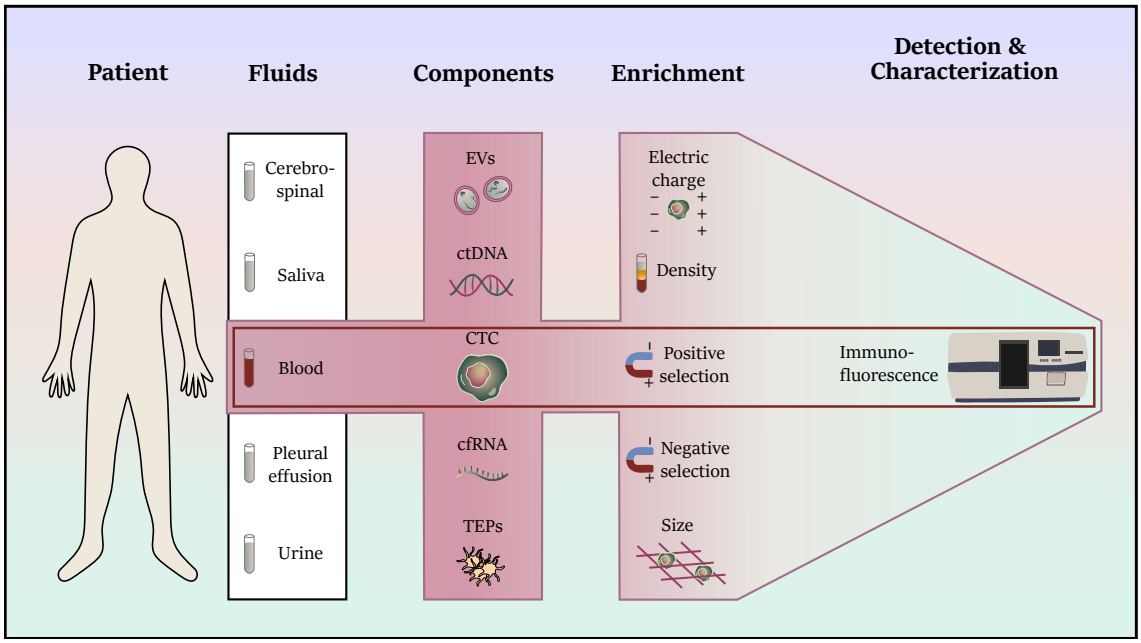


Figure 1.1.: Overview of LB. This figure illustrates five examples of fluids used in LB. When focusing on blood, it consists of various components, five of which are depicted and highlighted in the components column; among these are CTCs. Various strategies for the enrichment, detection, and characterization of CTCs are depicted. CTCs can be enriched using physical properties, employing technologies based on size, density, and electric charge. Alternatively, they can also be enriched using biological properties through marker-dependent strategies: CTCs may be either positively or negatively selected using antibodies targeting specific proteins. The red horizontal box highlights the enrichment and detection strategy used by the gold standard. Abbreviations and explanations: CTC: Circulating tumor cell; EV: Extracellular vesicle; ctDNA: Circulating tumor DNA; cfRNA: Circulating free RNA; TEPs: Tumor-educated platelets.

LB plays a role in early screening and aids in diagnosis, and through repeated sampling, this method allows for close real-time monitoring of tumor progression and therapy responses, and guides treatment decisions [AP25]. Several tumor-derived and tumor-induced circulating biomarkers can be detected by LB (Fig. 1.1), including tumor-derived extracellular vesicles (EVs), tumor-educated platelets (TEPs), circulating free RNA (cfRNA), circulating tumor DNA (ctDNA), and circulating tumor cells (CTCs), among

others. CTCs are tumor cells that have intravasated into the bloodstream and are one of the most investigated markers [AP25]. Most clinically relevant data are derived from studies on CTCs [KP19] and their clinical relevance has been shown in numerous studies [Bid+18; Eff+18; Abd+21]. Research indicates that both quantitative information (CTC counts), and qualitative information of CTCs, such as phenotypic characteristics, offer valuable insights.

The latter is particularly relevant due to the significant heterogeneity of CTCs; depending on the type and stage of cancer, they can differ greatly in size [Wen+24], display distinct physical characteristics [HNZ13], and express varying surface proteins [PA16; Tan+23]. Targeting surface proteins for CTC detection has been shown to be an effective method, as reported by studies with the CellSearch® (CS) system (Menarini Silicon Biosystems Inc., Huntingdon Valley, PA, USA), which is the only FDA (Food and Drug Administration)-cleared clinical application of CTCs and is considered the gold standard in CTC detection for patients with metastatic breast, colorectal and prostate cancer [MDT10; Swe+16]. Since most epithelial tumors express the epithelial cell adhesion molecule (EpCAM) on their surface, CS enriches for CTCs using anti-EpCAM antibodies, as epithelial cells typically do not circulate under normal conditions. CTCs are then identified through immunofluorescence staining for cytokeratins (CK), 4',6-diamidino-2-phenylindole, dihydrochloride (DAPI; marker for nucleus), and CD45 (positive marker for leukocytes). The expression levels of EpCAM, both up- ([GS14]) and downregulation ([Wen+04; Rao+05]), can further serve as an indicator of the aggressiveness of CTCs. For example, downregulation of EpCAM in CTCs is often accompanied by the upregulation of mesenchymal markers. This upregulation has been shown to increase tumor invasiveness and is linked to poorer survival [Zha+17].

As for the quantitative aspect of analyzing CTCs, CTCs are significantly less common in the blood, with approximately one CTC found among 10^6 to 10^8 leukocytes [BT13]. When analyzing a standard 7.5 mL blood sample using the CS system, the frequency of CTCs varies depending on the type of metastatic cancer. CTC frequency is relatively low in metastatic colorectal cancer, whereas it is comparatively higher in metastatic prostate and breast cancers [All+04]. Studies with the CS system (primarily in the metastatic setting) have demonstrated that the CTC count serves as a prognostic indicator for different types of epithelial cancers, such as metastatic breast, colon, and prostate cancers [Rie+18]. For metastatic colorectal cancer, having 3 or more CTCs per 7.5 mL of blood indicates a poor prognosis, while the consensus cut-off threshold in metastatic prostate and breast cancer is 5 or more CTCs per 7.5 mL of blood, which is linked to reduced overall and progression-free survival [Rie+18]. Once treatment has started, assessing

CTC counts at follow-up timepoints can also predict treatment effectiveness and response [Hay+06].

Challenges in CTCs by the CS system have been reported by Andree et al. [AVT16], including the system's semi-automated nature. After the enrichment and staining of cells, a software that is part of the CS system automatically generates a gallery of cropped images from fluorescence microscopy images. These cropped images are selected based on the presence of positive signals in the DAPI and phycoerythrin (PE; for CK) channels that are located in the same predefined square [Swe+16], rather than by specifically proposing CTC candidates. For enumeration of CTCs, a trained operator has to manually examine the presented images in the gallery and select CTCs [LTK16] based on the defined CS criteria [Rie+18]: DAPI and CK positivity, CD45 negativity, and a round or oval cell shape with a diameter of at least 4 μm. However, manual classification is constrained by user fatigue and can become very tedious. Especially in cases involving patients with advanced metastatic disease, extensive image galleries are generated by the system. To extend the capabilities of the CS system and in response to the challenges in CTC detection, differentiation, and enumeration, Zeune et al. [Zeu+17] introduced the Automated CTC Classification, Enumeration, and PhenoTyping (ACCEPT) tool. This tool facilitates the analysis of previously identified CTCs and also enables the detection and classification of CTCs in fluorescence images of blood samples. ACCEPT achieves this by applying linear gates to object features derived from multi-scale segmentation, with these features initially selected based on user observations. In more recent work by Zeune et al. [Zeu+20], the developers of ACCEPT acknowledged the pressing need for a fully automated and robust method for CTC enumeration and proposed the use of artificial intelligence (AI) to classify CTCs. They demonstrated the capability of AI to efficiently analyze large data sets in an automated manner, thereby providing a potential solution for the time-consuming tasks of CTC evaluation and sorting, and allowing faster and more consistent results.

The effectiveness of machine learning (ML) and deep learning (DL) techniques in detecting and classifying CTCs from other blood cells has been demonstrated, with some research efforts proceeding independently of the CS system [Che+16; Iye+20; ALL21], whereas others, such as those by Zeune et al. [Zeu+17; Zeu+20], have sought to address the limitations posed by the CS system. Although these methods typically achieve over 90% accuracy, they often require substantial amounts of ground truth (GT) data for effective classifier training in a supervised learning context. However, obtaining large labeled data sets is often not feasible, and greater volumes of unlabeled data remain available but underutilized. Initial efforts to reduce dependence on extensive labeled data have been undertaken for instance by Svensson et al. [Sve+14], who compared

unsupervised, semi-supervised and supervised classifiers for CTC detection. In the field of computer vision, Caron et al. [Car+21] presented a self-supervised learning (SSL) training method that leverages unlabeled data in a purely data-driven approach.

Although full automation is desirable, the involvement of human experts remains crucial. For instance, Wang et al. [Wan+16] demonstrated that the success rate in detecting metastatic breast cancer in whole-slide images of sentinel lymph node biopsies improved from 0.925 (using DL alone) and 0.966 (pathologist) to 0.995 when both approaches were combined. This illustrates the benefits of integrating ML with human expertise. Similarly, the integration of human expertise could be applied to enhance the accuracy of CTC differentiation.

1.2. Aims of this work

Given that the CS system is currently the gold standard for CTC detection and enumeration, yet possesses certain limitations, this work focuses on addressing these shortcomings. The overall aim is to develop an AI-based image-processing system for the detection and differentiation of CTCs and non-CTCs in LB data obtained from CS. Using metastatic breast cancer as an exemplary case, the approach involves applying SSL and human-in-the-loop methods.

A.1 The first aim of this thesis is to achieve robust detection of cells in the raw fluorescence microscopy images from the CS system by utilizing a DL-based segmentation method as an alternative to the existing, but not publicly available, CS segmentation approach.

A.2 Building on recent developments and success in the field of SSL [Car+21], the second aim is to achieve reliable, label-efficient binary classification of detected cells using the principles of SSL. The objective is to leverage the abundance of unlabeled data while requiring only a limited amount of labeled data for downstream CTC classification, thereby assessing label efficiency compared to state-of-the-art supervised DL models, and exploring the extracted feature maps of the learned representations.

A.3 Recognizing that a model cannot always be fully confident in its predictions, the third aim is to expand the developed system by integrating human expertise to efficiently label automatically selected training samples, particularly from latent space regions of classifier uncertainty. Utilizing a human-in-the-loop approach, this strategy aims to iteratively improve the classifier's

accuracy while reducing the annotation and evaluation workload for experts, particularly when compared to the effort required by the CS system.

A.4 The final aim of this thesis is to extend the comparison of CTC detection performance between the CS system and the model trained and optimized via the human-in-the-loop approach to an additional cohort, comprising both metastatic patients and healthy donors. This includes evaluating concordance, sensitivity, and the handling of potential ambiguous cells, thereby assessing the generalizability of the developed model.

1.3. Structure of this thesis

The thesis begins with chapter 2, which provides an introduction to the biological background necessary for this work. This chapter offers an overview of the role of CTCs in the metastatic cascade and elaborates on the sampling and processing procedures of the CS system used for CTC detection. Given the significance of CTC detection in the clinical setting and the limitations of the CS system, the chapter concludes with a comprehensive review of current state-of-the-art DL methods developed to address these limitations.

Chapter 3 delves into the principles of AI, covering the fundamentals of ML and DL, as well as concepts for latent space analysis relevant to this work. Further sections focus on key topics such as image segmentation and advanced AI models that are applied in this thesis.

Chapter 4 encompasses the experiments conducted and the results obtained throughout this study. At the beginning of the chapter, an overview of the underlying publications is provided. The chapter further introduces the in-house LB patient data and explains the application of a DL-based segmentation method on it. Subsequently, a label-efficient CTC classification strategy using SSL is introduced. Based on these methodological approaches, further sections expand the system by incorporating a human-in-the-loop strategy to improve the classification of CTCs. The final section focuses on the application of the developed method to additional patients and healthy donors.

Chapter 5 concludes the results, places them in the context of current literature, and provides an outlook for further possible research opportunities.

Chapter 2

Biological principles

2.1. Circulating tumor cells in the metastatic cascade

The metastatic cascade is a multi-stage process by which tumor cells invade the bloodstream or lymphatic system, survive transit, and ultimately colonize distant organs to form metastases. The process where CTCs act as the migratory units that travel through the blood stream is illustrated in Fig. 2.1. Since the primary tumor rarely causes the death of cancer patients, but rather metastases at distant sites do, understanding this process is crucial for improving patient prognosis and identifying strategies to prevent the formation of metastases.

The initiation begins with cancer cells acquiring changes in genes associated with various processes known as the hallmarks of cancer, such as evading growth suppressors, resisting cell death, and activating invasion and metastasis [HW11]. The metastatic cascade starts when primary tumor cells locally invade their surrounding microenvironment and migrate across the endothelial barrier to enter the bloodstream [Law+23]. One possible explanation for this invasive and migratory capability of cancer cells is the epithelial-to-mesenchymal transition (EMT), during which they lose epithelial traits such as cell-cell adhesion [Mit18] and acquire mesenchymal characteristics, including the expression of vimentin [YY17], an intermediate filament protein that is a crucial component of the cytoskeleton. Satelli and Li [SL11] reported that vimentin is frequently overexpressed, for instance, in breast cancer, and is associated with cancer cell invasiveness and metastasis.

Once in the bloodstream, CTCs face environmental stressors such as oxidative stress and shear forces [MM21]. While the majority of CTCs die [Lin+21], some undergo complex, adaptive processes to survive. Moderate shear stress has been shown to induce transcriptional changes that promote cell motility and migration [Lee+17]. Simultaneously, CTCs strive to evade detection by the immune system [Wan+18b]. CTCs can escape immune destruction, for example, by expression of inhibitory molecules such as programmed death-ligand 1 (PD-L1), or through downregulation of major histocompatibility complex (MHC) class I on the surface, which causes reduced recognition by immune cells [TB22]. Further, in compositions where CTCs form clusters (Fig. 2.1) - whether comprising only CTCs or a mix of CTCs with stromal or immune cells - they

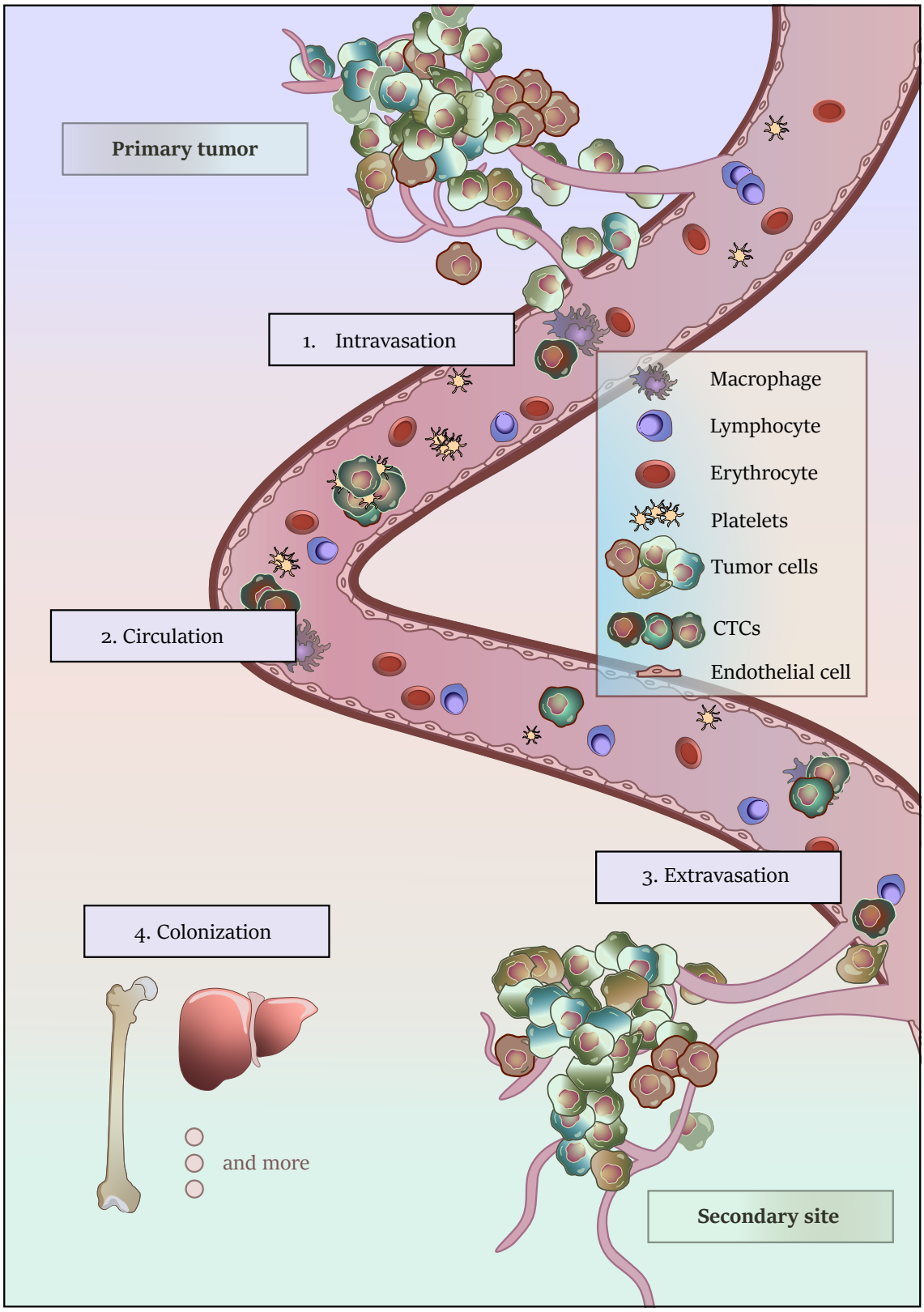


Figure 2.1.: Overview of the metastatic cascade: (1) Intravasation - cancer cells detach from the primary tumor and enter the bloodstream; (2) Circulation - survival as circulating tumor cells and interaction with blood cells; (3) Extravasation - exit from the bloodstream; (4) Colonization - tumor cells spread to distant sites (e.g., bone, liver). Not to scale; proportions and cell frequency are illustrative.

exhibit greater metastatic potential compared to individual CTCs [Ace+14]. This is partly due to their shorter circulation times of 6-10 minutes, compared to 25-30 minutes for individual CTCs [Rin+23], which enhances their survival and offers better protection from immune attacks.

Ultimately, their metastatic success relies on the ability to rapidly extravasate through the endothelial barrier into surrounding tissues. The efficiency of extravasation is influenced by various factors, including the expression of adhesion molecules for direct interaction with the endothelial cell wall, the release of chemokines, the physical characteristics of CTCs - such as cluster size and deformability, which play a role in their mechanical trapping in capillaries - as well as the involvement of supporting cells such as neutrophils [Rin+23].

After extravasation, reverse EMT, also known as mesenchymal-to-epithelial transition (MET), may restore their epithelial characteristics to facilitate metastatic growth [JZX17], enabling cancer cells to colonize distant sites (e.g., bones, liver) and form metastases. However, cancer cells also face various challenges, such as evading immune responses and adapting to new microenvironments. The colonization process can stretch across several years, and cancer cells may even enter a latent state [MO16] before they potentially break out and successfully establish new growths at these distant locations.

2.2. CellSearch®

The presence of CTCs in peripheral blood is associated with poor prognosis. Using the CS system for isolation and enumeration of CTCs, this correlation has been demonstrated in prospective multicenter studies of metastatic prostate cancer [De +08], colorectal cancer [Coh+08], and breast cancer [Cri+04]. In light of these findings, this chapter seeks to provide a more comprehensive understanding of the CS procedure and principles. This section begins with an overview of specimen collection and preparation, followed by the processing performed using the CELLTRACKS® AUTOPREP® System. Subsequently, the analysis process using the CELLTRACKS ANALYZER II® is described, covering the steps from scanning a sample to reviewing and reporting the results. Although the following sections provide an overview of the respective CS procedures, for detailed instructions, the respective CS User's Guides should be consulted.

2.2.1. Whole blood collection, preparation, and processing

Whole blood is collected into a CellSave Preservative Tube. From this, 7.5 mL of blood is transferred into a 15 mL CS Conical Centrifuge Tube, and a dilution buffer is added. The

tube is then capped, gently mixed, and centrifuged, leading to the separation of plasma and red blood cells, with the plasma forming a layer at the top.

The prepared sample is then processed by the automated sample-processing CELLTRACKS® AUTOPREP® System, used with a kit. In the case of the CS Circulating Tumor Cell Kit, this includes a ferrofluid-based capture reagent and immunofluorescent reagents. In the first step, the system aspirates the plasma layer, then adds ferrofluids with a magnetic core coated with epithelial cell-specific EpCAM antibodies, that bind to EpCAM antigens for capturing CTCs. The sample is then exposed to a strong magnetic field, causing the antibody-coated immunomagnetic cells to be drawn to the side of the tube, and allowing for the separation of bound cells. Although the process enriches for EpCAM-positive cells, many leukocytes still remain among them [Swe+13]. Following enrichment, fluorescently labeled antibodies are added to bind to antigens on the target cells in order to label them. DAPI is used to stain the nucleus, PE labeled antibodies are used to recognize CKs 8, 18, and 19, among others, which are intracellular proteins present in most cells derived from epithelial cancer, and allophycocyanin (APC) is used to label CD45, which is found on the surface of leukocytes, serving as an exclusion marker. The magnetic separation is then repeated after incubation, excess staining reagents are aspirated, and a fixative is added. The processed sample is lastly transferred by the CELLTRACKS® AUTOPREP® System into a cartridge, which is then placed in a MAGNEST® Cartridge Holder. This holder contains a strong magnetic field to magnetically align the labeled epithelial cells at the cartridge surface. The sample surface is then scanned by the CELLTRACKS ANALYZER II®.

2.2.2. Circulating tumor cell analysis using the CELLTRACKS ANALYZER II®

The CELLTRACKS ANALYZER II® is a partly automated fluorescence microscope designed to capture the fluorescently labeled, aligned cells and transfer the resulting image data to a connected computer running the CELLTRACKS® software. This software is used for enumeration and enables an expert to view, analyze, and conduct a CTC selection of the presented events.

The system begins by performing a coarse focus on the ferrofluid, followed by edge detection of the cartridge, and then scans the entire surface of the cartridge frame by frame, and an image is taken with each filter. The filters are DAPI, an additional filter, PE, and APC. The scheme in Fig. 2.2 visualizes the scanning process.

The system divides the cartridge usually into 175 frames (for each channel), organized into 5 rows with 35 frames each. The scanning process begins at the top left corner of

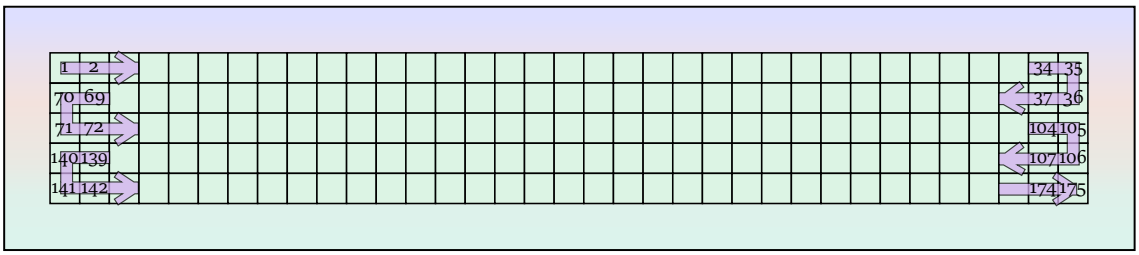


Figure 2.2.: CS scanning procedure for a cartridge. The surface is organized into 175 frames, arranged in 5 rows of 35, following a zigzag pattern from top left to bottom right.

the first row and moves across to the rightmost frame. It then continues directly below, starting from the rightmost frame in the second row and moving to the left. This zigzag pattern continues with the third row being scanned from left to right, the fourth row from right to left, and finally, the fifth row from left to right, ending with the last frame, frame 175.

After the scanning process, the software identifies objects where DAPI and CK staining are close to one another. It then displays these objects as cropped images for each channel separately, along with a composite color image combining the DAPI and CK channel (the latter is sometimes referred to as the CK-PE channel), in a gallery format as shown in Fig. 2.3A.

CK is displayed in green and the cell nucleus is shown in pink in the composite image, but can sometimes appear white depending on the signal intensities of both channels. The images in the gallery are initially presented as unassigned events and are sorted by event number and frame number. An expert then classifies an image as a CTC by clicking on the composite image, which creates an orange box around the displayed composite image. If a more detailed image of the displayed cells is desired, the expert can examine the image in each channel on the cartridge frame by clicking on the event number of the corresponding cell. The screen then switches to the Cell Select view, depicting yellow boxes around the events of the respective frame, as shown in Fig. 2.3B. When a box is clicked, the selected image is highlighted with a red dashed box and can be enlarged. The typical image size in the gallery is 80×80 px, but some images may differ in size, as indicated by the varying sizes of the yellow boxes. The current cartridge image in Fig. 2.3B suggests that the gallery will be extensive, as numerous yellow boxes are already displayed for a single frame. This scenario is typical in cases of metastatic disease, where the operator reviews the images in such galleries, demanding substantial effort and time to classify them as CTCs. Not only is the number of images that need to be reviewed a challenge for the operator, but interpreting the respective stainings also presents difficulties. Fig. 2.4 shows that bright signals in the CK-PE channel can

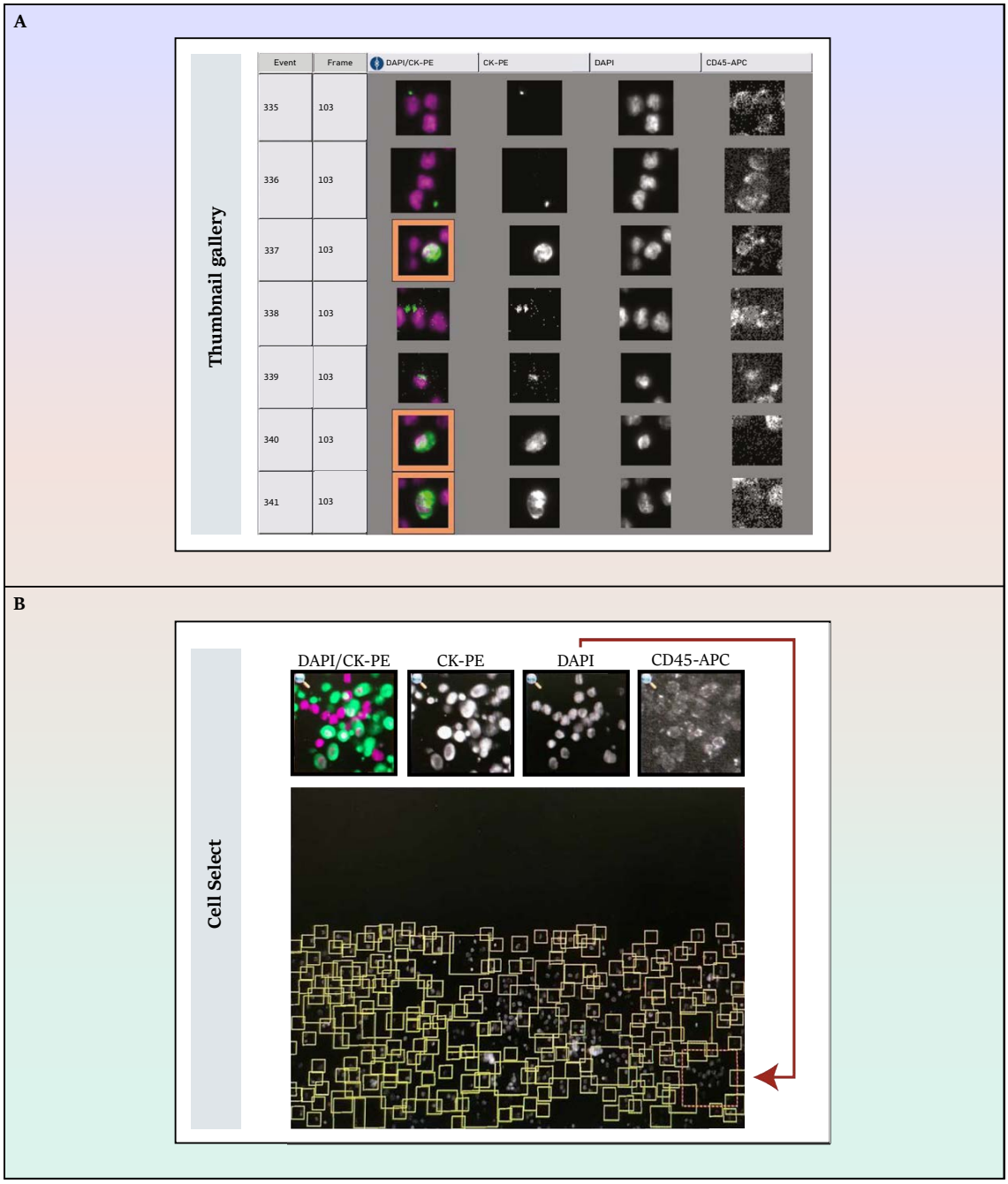


Figure 2.3.: CS image gallery and exemplary features of the Cell Select view. (A) In each row of the gallery, from left to right, the following are displayed: Event number, frame number, composite image (DAPI/CK-PE), and images for each channel (CK-PE, DAPI, CD45-APC). The green color in the composite image represents CK-PE, while pink stands for DAPI. Composite images with an orange border (events 337, 340, and 341) are manually assigned as CTCs (positive for CK-PE, DAPI, and negative for CD45) by the operator. Figure adapted from publication [WDT14], licensed under Creative Commons Attribution (CC BY). (B) The Cell Select view allows for detailed inspection of gallery cell images. Cartridge frames can be examined for each channel, with the presented cell images in the gallery highlighted in yellow boxes. The location of the clicked gallery image (see top row) is highlighted with a red dashed box in the underlying cartridge channel image, as shown here using the DAPI channel. Abbreviations and explanations: DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK-PE: Cytokeratin-phycoerythrin; CD45-APC: Exclusion marker-allophycocyanin.

lead to shine-through effects (spectral spillover) in the APC channel, requiring closer examination by the operator.

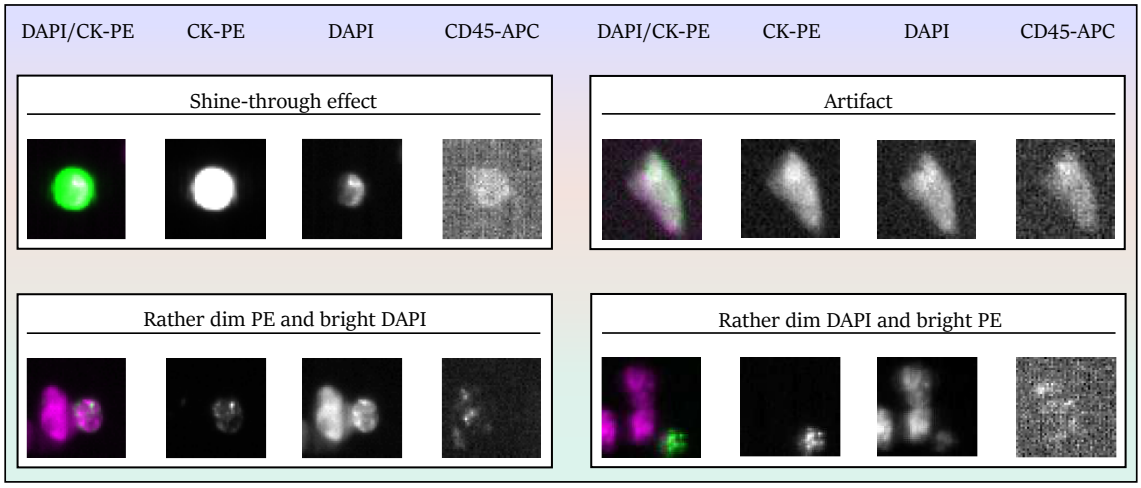


Figure 2.4.: Appearances of cell images in the gallery. Shine-through effects, artifacts, and variations in intensity across the CK-PE and DAPI channels can occur in the CS gallery. Abbreviations and explanations: DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK-PE: Cytokeratin-phycoerythrin; CD45-APC: Exclusion marker-allophycocyanin.

Furthermore, there are cases where the CK-PE signal is relatively bright while the DAPI signal is rather dim, or vice versa. In such instances, the signals in the composite image must be examined more carefully. Additionally, artifacts, apoptotic cells, and cell clusters have been also reported by Andree et al. [AVT16]. In general, studies using the CS system have shown consistent results in identifying CTCs [Cum+13; Kra+11; Rie+07]. However, despite this concordance, the aforementioned variability in cell appearances continues to cause differing object or cell assignments during manual image review by various operators, especially across platforms. One of the first attempts to address this issue was made by Zeune et al. [Zeu+17] through the development of the ACCEPT tool.

2.3. State of the art and beyond CellSearch®

Despite the urgent need for fully automated CTC detection, there is little research with DL-based methods in this field. Chen et al. [Che+16] proposed a label-free detection approach based on a fully-connected neural network using optical phase imaging, and Mao et al. [MYS16] developed a detection system using a deep convolutional neural network (CNN) on images independent of cell marker expression. The developers of the ACCEPT tool [Zeu+20] also emphasized the use of DL methods for fully automated CTC classification and analysis and explored CNNs; however, their work is the only effort directly aimed at the clinically relevant imaging setup of the CS system. As their

work serves as the motivation for this thesis, this section will present an overview of the methods and the main results by Zeune et al. [Zeu+20]. The theoretical concepts underlying these approaches will be explained in the following chapter 3.

2.3.1. Deep learning of circulating tumor cells

As CTCs are among other cell classes within the blood, the work of Zeune et al. [Zeu+20] aimed to automatically classify fluorescent cell thumbnail images to their respective cell categories. They tackled this multiclass classification task with two different DL approaches: one using a standard CNN (for further details see Section 3.2.5) and the other using an autoencoder CNN. The second aim was then the visualization of the latent space information of both approaches to inspect cell classes and potential CTC subpopulations. An overview of their workflow is shown in Fig. 2.5.

For their study, they used approximately 500 processed cartridge images from the CS system, with high reported CTC counts, originating from patients with metastatic breast, prostate, colon, and lung cancer, as well as benign breast disease and from cultured cancer cell lines. The samples were then processed using the ACCEPT tool, and measurements (e.g., mean intensity) for each cell and fluorescence channel (DAPI, CK, and CD45) were extracted. Linear gates were defined and applied, and candidates falling within the defined gates were then manually scored by several reviewers to determine if the images truly belonged to the four respective categories: CTC, tdEV, white blood cells, and bare nuclei. Images that were left unlabeled by the reviewers were then re-examined and assigned as either artifacts or unknown objects. The four cell categories and the class of unknown objects together formed a set of five classes, and the GT set was reviewed again by another reviewer to reduce the number of noisy images. Lastly, an additional class was added to the GT, consisting of objects that lack the expression of DAPI and CK but are CD45 positive, named CD45EVs. Thus, the fluorescence images of the six cell classes comprised an extensive GT, as depicted in Fig. 2.5.

Before the fluorescent images were used as inputs for the networks, they underwent pre-processing. Both the standard CNN and the autoencoder CNN include a convolutional feature extraction component, known as the encoder, which processes the image data through multiple layers, reaching a low dimensional feature space. In this process, the images are compressed from an $80 \times 80 \times 3$ input format to a 50×1 feature representation. This latent representation is fed into fully connected layers for classifying the input based on the extracted features. The outputs sum to one, allowing them to be interpreted as the respective class probabilities. The network is trained by minimizing the error between

the true class label and predicted class distributions, known as the cross-entropy (CE) loss (for further details see Section 3.2.2).

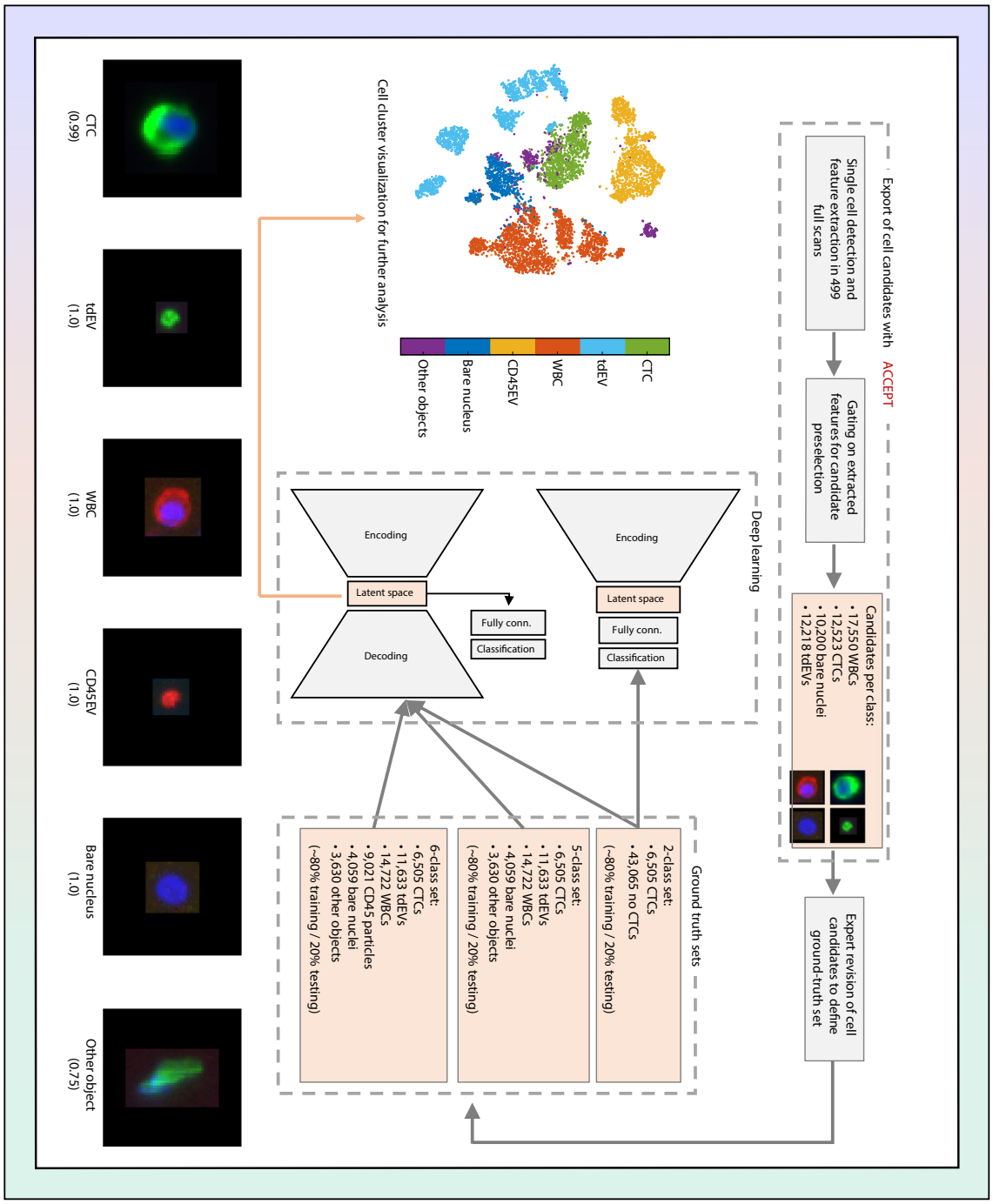


Figure 2.5.: Overview of the worflow of Zeune et al. The workflow starts with the processing of archived CS images using the ACCEPT tool, followed by manual review to generate an extensive GT for training and validation of the two networks. Probabilities assigned to the respective class images are shown. CTC and cell analyses are performed in the latent space of the extracted features. Abbreviations and explanations: tdEV: Tumor-derived EV; WBC: White blood cell. Image adapted from [Zeu+20], reproduced with permission from the copyright holder (Springer Nature).

For the autoencoder CNN, the encoding and classification components remain unchanged, as illustrated in Fig. 2.5. The key difference from the standard CNN is that the latent space representation is additionally fed into a decoding part, consisting of separate convolutional decoding networks for each of the three fluorescence channels, to reconstruct the encoded input image. The network is trained by minimizing a loss function aimed at reducing errors in both reconstruction and classification, with a factor that determines the influence of the classification on the weights.

Throughout their experiments, the data set was divided into either a two-class (CTC vs. No CTC), five-class, or six-class set, and each of the sets was randomly split into 80% training and 20% validation. As for the six-class classification, the results are depicted in Fig. 2.6A.

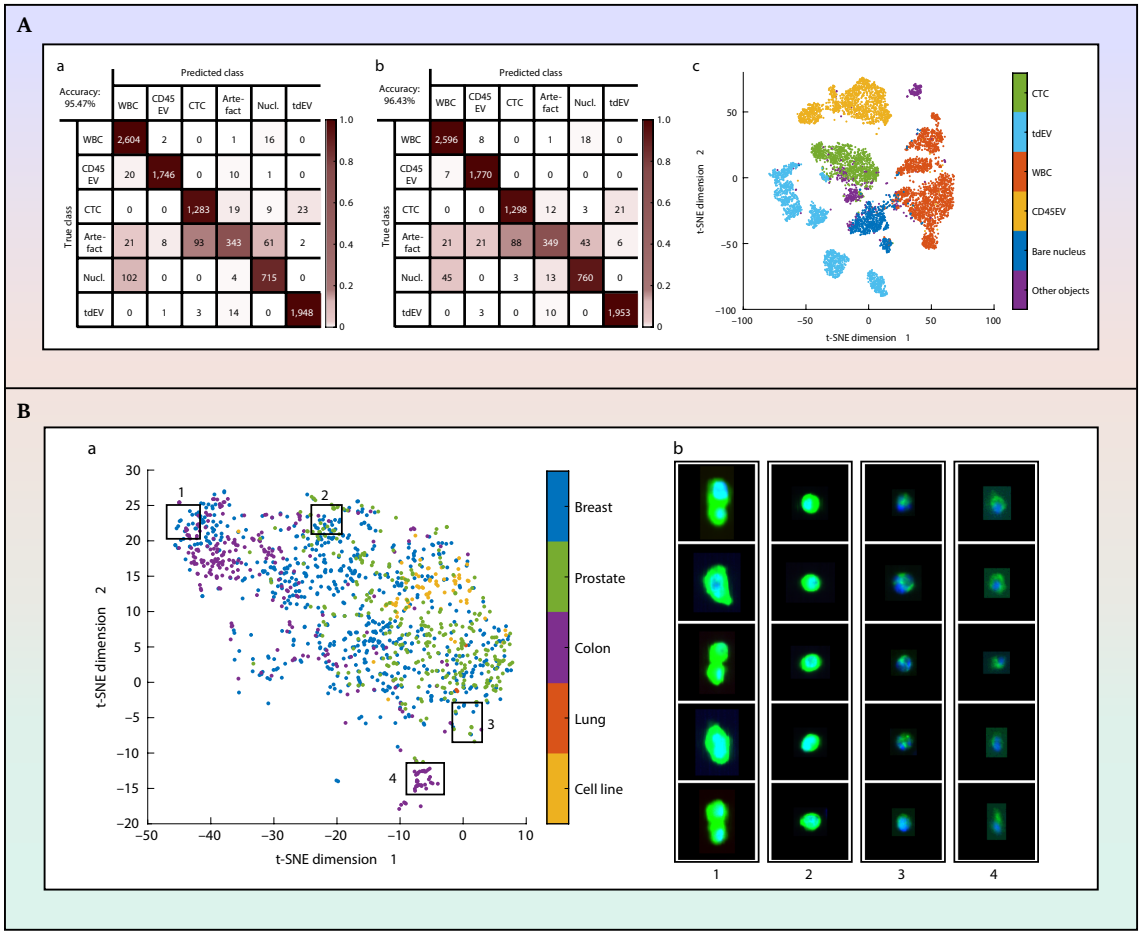


Figure 2.6.: Results from Zeune et al. [Zeu+20]. (A) DL-based CTC image classification. Confusion matrix for six-classes using (a) a standard approach (CNN), (b) autoencoder; (c) 2D t-SNE visualization of autoencoder latent space, revealing CTC subpopulations beyond the six defined classes. (B) CTC heterogeneity in latent space. (a) 2D t-SNE map for CTCs across four cancer types and cell lines, (b) Example CTCs from distinct t-SNE regions, each showing different morphology and staining. Abbreviations and explanations: tdEV: Tumor-derived EV; WBC: White blood cell; t-SNE: t-distributed stochastic neighbor embedding. Images adapted from [Zeu+20], reproduced with permission from the copyright holder (Springer Nature).

Regarding CTC classification, both approaches - the standard CNN as well as the autoencoder CNN - correctly classify the majority of CTCs. Interestingly, many images from the class of artifacts were mistakenly classified as CTCs, yet most of these misclassifications appear to belong to the category of other objects.

The overall classification accuracy is high for both approaches. To visualize the latent space, t-distributed stochastic neighbor embedding (t-SNE), a dimensionality reduction method, is applied to further compress the 50D feature space to 2D. The 2D t-SNE map of the autoencoder CNN is shown in Fig. 2.6A. Upon inspecting the latent space, distinct clustering of all six cell classes becomes apparent, and even subpopulations within the classes of tumor-derived EVs and white blood cells are revealed.

In the next step, the heterogeneity of CTCs across the cancer types was further analyzed, as depicted in Fig. 2.6B. When randomly selecting points from different locations within the latent space (Fig. 2.6B), the CTCs exhibit distinct morphological and staining characteristics. CTCs from location 1 are either large in size or overlap with a neighboring CTC; CTCs from location 2 are round and exhibit a strong CK signal; CTCs from location 3 have a more subdued CK staining; and CTCs from location 4 display dim CK and DAPI staining.

To sum up, Zeune et al. introduced promising CNN-based approaches for the classification of CTCs and other cells in the blood. They further analyzed cell classes in the latent space, revealed distinct clusters of similar cells, identified subpopulations within cell classes, and examined the heterogeneity of CTCs. As the presented results demonstrated the successful application of DL for CTC differentiation, understanding the concepts of ML and DL is of utmost importance to advance research in the area of robust and automated CTC detection, differentiation, and enumeration.

Chapter 3

Principles of artificial intelligence and image processing

In recent years, the concepts of AI, particularly ML and DL, have attracted considerable attention across many research fields. As the availability of data, computational power, and memory increase, the development and application of ML and DL approaches become more feasible. They can act as adjunct tools to human expertise or solve problems independently, often more efficiently and quickly than humans, particularly in the processing and analysis of large data sets to extract useful information and uncover patterns.

Given these capabilities, a particularly promising area for the application of ML and DL is the medical and biomedical field, as these domains are often characterized by highly complex, heterogeneous, and sometimes abundant data that require sophisticated analysis. Tailored ML and DL solutions can support physicians and researchers by enabling more precise and efficient data analysis, reducing the risk of medical errors, and providing better standardization of protocols, such as for diagnosis and prognosis [ZZR19].

In this thesis, approaches from ML and DL are employed to address and overcome biomedical data analysis challenges. Accordingly, the following sections introduce the core concepts and selected methods from ML and DL that are particularly important in the context of this work.

3.1. Machine learning

In the literature, ML is defined as the scientific study that integrates elements of mathematics, statistics, and computer science [SS19], focusing on the development of algorithms and statistical models that perform specific tasks by extracting and learning relevant patterns and inferences from data, rather than relying on a predefined set of rules [Cui+20]. However, ML algorithms are more accurately characterized based on their training methods, the availability of annotated data, and the types of problems they are designed to solve. ML can generally be divided into supervised, unsupervised, and reinforcement learning (where an agent learns from trial-and-error feedback). In the following sections, supervised and unsupervised learning will be presented in more detail.

3.1.1. Supervised learning

Supervised learning deals with labeled data, where both the input and the desired output are known. By fitting a mapping function through training input-output pairs it enables making predictions on new, unseen data. Classification problems are commonly solved using this approach, with k-nearest neighbors (KNN) and support vector machines (SVM) being two of the most well-established algorithms. KNN is considered one of the most straightforward ML algorithms [SK23b], so a brief description will be provided below. SVM was employed for all downstream CTC classification tasks in this thesis and will therefore be presented in greater detail.

K-nearest neighbors: The objective of the KNN method for a new test sample is to find a predefined number of training samples whose feature vectors are closest in distance to it, as measured in the feature space using a specific distance metric - most commonly the Euclidean distance - and then to determine the class based on these neighbors [CD21]. The simplest strategy for classification is to assign the class label that is most frequent among the nearest neighbors (majority vote) to the new point (see Fig. 3.1). The number

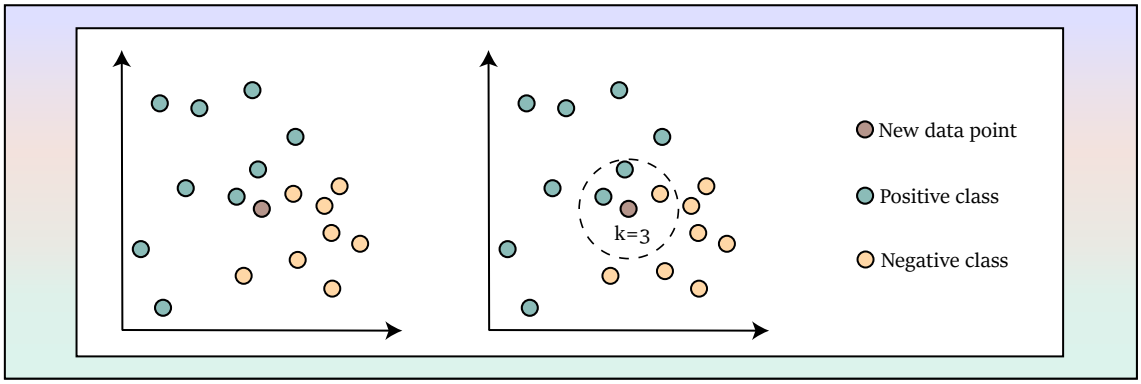


Figure 3.1.: Visual illustration of the KNN classification. Training samples with two classes, along with a new test sample, are shown in a 2D feature space. To classify the new sample, KNN uses a distance measure (e.g., Euclidean distance) to find the nearest neighbors. In this example, k is set to 3. Therefore, for the test sample, the positive class represents the majority class (dashed circle) among these neighbors.

of samples is the user-defined constant k, and if $k = 1$, then the sample is assigned to the class of its nearest neighbor. KNNs have been widely applied and shown to be effective in classification problems [Li+12; MSB13; RM12], but the choice of k is not trivial, as the class assignment may vary depending on the chosen value of k [KK13].

Support vector machine: The primary objective of the SVM is to separate classes in the training set with a decision surface (or hyperplane) in feature space that maximizes

the margin between the classes - the distance from the hyperplane to the closest data points of each class - thereby minimizing the classification error [Cer+20].

The simplest case for an SVM occurs when the training data is linearly separable into two classes, that are the positive (+1) and negative (-1), in a 2D feature space (see Fig. 3.2). Given N training points, the set of input feature vectors is denoted as X, where each $\mathbf{x}_i$ (for $i = 1, ..., N$) belongs to $\mathbb{R}^D$ (D is the dimensionality) [Fle09], and is associated with a class label $y_i \in \{-1, 1\}$ from the set Y of class labels. When $D = 2$, a line can be drawn to separate the two classes; for $D > 2$, the separation occurs via a hyperplane [Fle09]. The equation for the linear hyperplane can be expressed as follows:

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b, \tag{3.1}$$

where $\mathbf{w}$ is the weight vector normal (perpendicular) to the hyperplane, and b is the bias term [Fle09]. The bias enables the hyperplane to be shifted away from the origin [Boo+25]; without it, the decision boundary would always pass through the origin.

To find the optimal plane between the two classes, a subset of training samples that lie on the marginal hyperplane is considered. These samples, known as support vectors, are crucial because they define the separating hyperplane. If any support vector is removed, the position of the hyperplane will change [Cer+20].

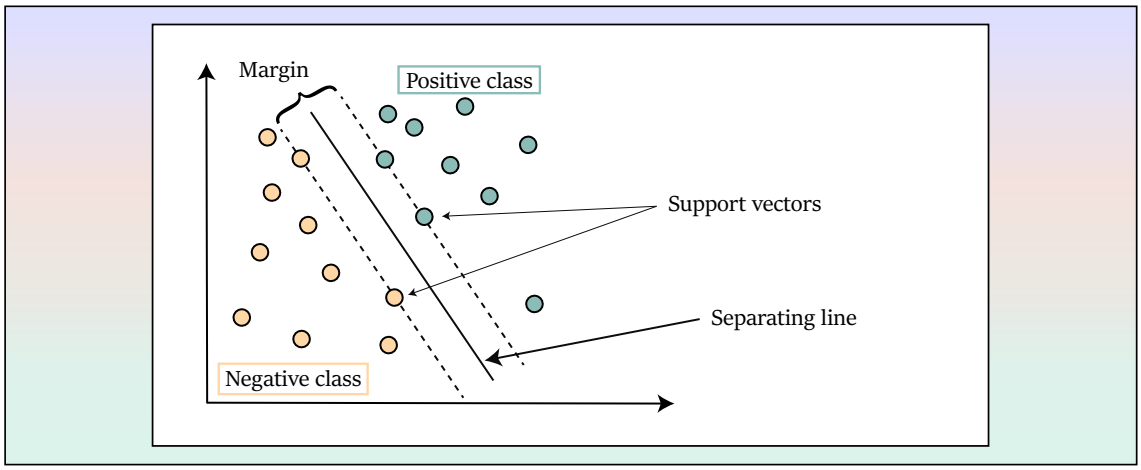


Figure 3.2.: Visual illustration of a linear SVM in a 2D feature space. The support vectors are the data points nearest to the separating hyperplane (here a line), determining the margin with which the two classes are divided.

The two planes H_1 and H_2 on which these points (i.e., the support vectors) lie on can be described by:

$$\mathbf{w} \cdot \mathbf{x}_i + b = +1 \quad \text{for } H_1 \tag{3.2}$$

$$\mathbf{w} \cdot \mathbf{x}_i + b = -1 \quad \text{for } H_2. \tag{3.3}$$

For this linearly separable data set, the optimization problem, where all data points also are correctly classified, can be defined as:

$$\min \frac{1}{2}||\mathbf{w}||^2 \tag{3.4}$$

subject to the constraint:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1. \tag{3.5}$$

When the data contains outliers or is not separable, the SVM can tolerate some misclassifications by favoring a more generalizable margin. To achieve this, the optimization problem is modified as follows:

$$\min \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{N} \xi_i \tag{3.6}$$

subject to the constraints:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \tag{3.7}$$

where ξ_i are slack variables that measure the degree to which the i-th data point violates the margin requirement, and C is a regularization parameter that controls the balance between achieving a wide margin and reducing the classification error [Zho24; Zha12]: a high value of C results in a greater penalty for misclassifications.

The SVM can be extended through the use of kernel functions - a technique known as the kernel trick - which enables the classification of data that are not linearly separable (in the space of the inputs $\mathbf{x}$) by implicitly mapping them into a higher-dimensional feature space [Fle09]. The kernel function avoids explicitly calculating the coordinates in the higher-dimensional space; instead, it measures the similarity between pairs of data points (using their dot product) in the higher-dimensional space. A linear kernel K is simply the dot product of two input vectors $\mathbf{x}$ and $\mathbf{y}$ in the original feature space:

$$K(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}. \tag{3.8}$$

In addition to the linear kernel, commonly used kernels include the polynomial, sigmoid, and radial basis function (RBF) kernels. The RBF kernel is particularly popular, as it tends to perform well across a wide range of problems [PC13; Cla+14]. The RBF kernel allows the SVM to model complex, nonlinear relationships by implicitly mapping the data into

an infinite-dimensional feature space [Cla+14], where the similarity between two data points is measured. It is defined as:

$$K(\mathbf{x}, \mathbf{y}) = e^{-\gamma \|\mathbf{x}-\mathbf{y}\|^2}, \tag{3.9}$$

where γ is a hyperparameter that controls the influence of each individual training sample on the decision boundary. High values of γ result in a more localized (close) effect wherease lower values indicate rather far influence. Depending on the data, optimum parameter search might be necessary to achieve good classification results [KC09].

3.1.2. Unsupervised learning

While supervised learning relies on labeled data, many real-world scenarios lack explicit labels. In these cases, unsupervised learning offers valuable methods for uncovering patterns and underlying structures within the data. When working with large data sets, unsupervised techniques such as feature reduction and clustering become particularly useful. Clustering is a method that groups data into collections based on the characteristics and similarities of data point features [Ezu+22] and it can potentially reveal previously unnoticed subgroups within the data [All+20].

In the following, a brief overview of dimensionality reduction is provided, followed by a more detailed presentation of selected clustering methods.

Dimensionality reduction techniques

Dimensionality reduction seeks to transform data from a high-dimensional space into a meaningful representation with fewer dimensions through mathematical transformation. This can be achieved by preserving only the most essential variables from the data set or by removing irrelevant, redundant, or noisy variables, thereby creating a smaller set of new variables [SVM14]. Dimensionality reduction not only facilitates feature selection and compression of high-dimensional data, but also aids in tasks such as visualization and classification. Among the most commonly used methods are principal component analysis (PCA) [Pea01; Hot33], t-SNE [VH08], and uniform manifold approximation and projection (UMAP) [MHM18].

Depending on the specific problem or task, these methods can be used as alternatives to one another (PCA: linear method; t-SNE and UMAP: non-linear manifold learning methods), or, in certain scenarios, stacked in sequence to further reduce dimensionality - resulting in faster training times and lower computational complexity [Lov+21]. For instance, PCA can initially be applied to compress dimensions to a range such as 30

to 100, before utilizing t-SNE or UMAP [Mar24] for further reduction to two or three dimensions. This approach accelerates the execution of t-SNE or UMAP, and both can be used for data visualization. In another example, Allaoui et al. [AKC20] demonstrated the application of UMAP as a preprocessing step, which aided in clustering within a reduced feature space using algorithms such as K-Means and Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN).

Clustering

Clustering methods can be broadly categorized into partitional, hierarchical, and density-based approaches [BM21]. Partitional clustering divides a set of data points into a specified number of non-overlapping clusters [Ran+23a]. Hierarchical clustering seeks a more complex grouping of data points into a set of nested clusters, such as in a dendogram. Density-based clustering does not require that every data point belongs to a cluster, as it identifies dense regions. Among the most well-known methods in the last category are Density Based Spatial Clustering of Applications with Noise (DBSCAN) [Est+96] and the more recent HDBSCAN [CMS13], which builds upon DBSCAN.

Below, K-Means is presented as a representative example of partitional clustering, both for its simplicity and its requirement to specify the number of clusters in advance. In contrast, HDBSCAN will be explained in greater detail due to the special focus given to this approach during the thesis, as it can automatically determine the number of clusters in a data set.

K-means: In K-means clustering, points are divided into a number of disjoint (distinct and non-overlapping) groups [Nae+23]. This division is achieved through an iterative algorithm, where the user specifies the desired number of clusters (k) in advance. An exemplary overview of the main steps involved in the K-means clustering algorithm is illustrated in Fig. 3.3.

The algorithm begins by randomly selecting k initial cluster centers (see A and B in Fig. 3.3), known as centroids, from the data set. For each data point, the algorithm calculates its distance to each centroid, commonly using the Euclidean distance in the feature space [Nae+23], and assigns the point to the cluster with the closest centroid. Once all points are assigned, the centroids are recomputed as the mean position of the data points within each cluster [Ram+25] (see C in Fig. 3.3). These two steps, assignment and centroid update, are repeated iteratively until the centroids no longer change significantly or a predefined maximum number of iterations is reached (see D in Fig. 3.3).

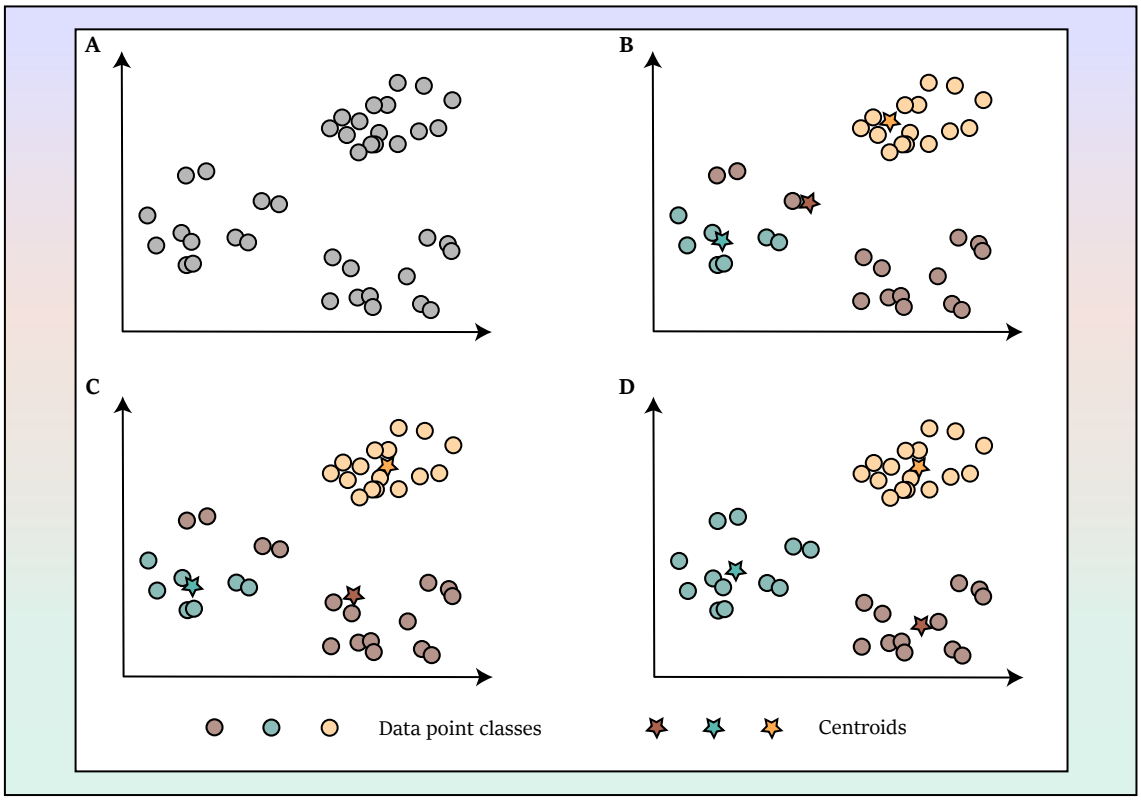


Figure 3.3.: Example of K-means clustering with $k = 3$. (A) The initial data distribution before K-means partitioning. (B) Centroids are randomly initialized, and each data point is assigned to its nearest centroid. (C) The centroids are recomputed to the mean position of all points assigned to each cluster. (D) After several iterations of reassignment and centroid updating, the final k clusters are obtained.

Density-Based Spatial Clustering of Applications with Noise algorithm: DBSCAN is an algorithm capable of finding arbitrarily shaped clusters and defines clusters as regions where points are densely packed together, separated by regions where points are sparse [Est+96; KR16]. If the Euclidean distance in the feature space is used as the similarity measure, the region is defined as a hypersphere with radius ϵ around the given data point p as its center [KR16]. The ϵ-neighborhood of point $p \in D$ is called $N_\epsilon(p)$ and is defined as:

$$N_\epsilon(p) = \{q \in D \mid dist(p, q) \leq \epsilon\}, \tag{3.10}$$

where D is the set of data points and $dist(p, q)$ is the function calculating the distance between points p and q [BF23]. DBSCAN requires two input parameters: ϵ and $MinPts$. The parameter $MinPts$ represents a density threshold, indicating the minimum number of neighboring points belonging to a core point. If the number of points belonging to the ϵ-neighborhood of p meets or exceeds the $MinPts$, then p is denoted as a core point. If it is less than $MinPts$, then it is referred to as a border point [KR16]. A point p is considered

noise if it does not qualify as either a core point or a border point [SC19]. An overview is given in Fig. 3.4. The DBSCAN algorithm uses the concept of density reachability and

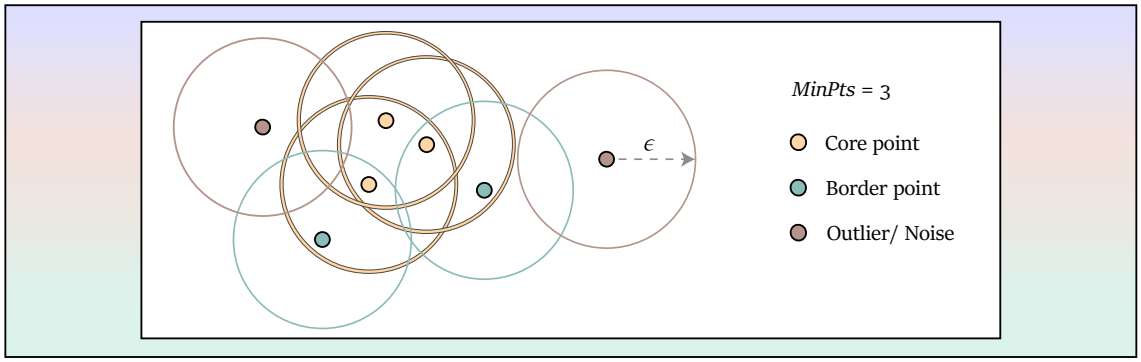


Figure 3.4.: Example of clustering by DBSCAN with the *MinPts* parameter set to three. The orange points represent core points that form a cluster. Two border points, which are density-reachable from the core points, are included in the same cluster. The rose-colored points, however, are not reachable and are therefore considered noise points.

density connectivity [BF23; SC19]:

(1) directly density-reachable: when q is a core point and p is in the ϵ-neighborhood of q, then p is directly density-reachable from q (with ϵ and $MinPts$).

(2) density-reachable: when there is a chain of points $p_1, p_2, \ldots, p_n$, with $p_1 = q$, $p_n = p$, such that p_{i+1} is directly density-reachable from p_i (with ϵ and $MinPts$) for all $1 \leq i \leq n$.

(3) density-connected: two points p and q are density-connected (with ϵ and $MinPts$) if there is a point o such that both p and q are density-reachable from o.

The algorithm starts by selecting a random unvisited point p from the data set and examining its ϵ-neighborhood [KR16]. If the ϵ-neighborhood of point p contains at least $MinPts$ points, then p is considered a core point, and a new cluster is formed. Points that are directly density-reachable from p are then iteratively included in the same cluster. This process can lead to the expansion of the cluster and may result in the merging with other clusters. The algorithm continues to form and expand clusters in this manner, and it terminates when all points have been visited and no further points can be added to any cluster.

As mentioned above, in DBSCAN, ϵ is determined by the user, and there is no definite method for selecting it when working with real-world data. Additionally, if the densities of individual clusters in a data set vary, ϵ might mistakenly classify a legitimate group as noise [PBC21].

Hierarchical Density-Based Spatial Clustering of Applications with Noise: HDBSCAN builds upon the concepts of a modified version of DBSCAN known as DBSCAN* [CMS13], which treats border points as noise. Unlike DBSCAN*, which relies on a fixed ϵ parameter to define clusters, HDBSCAN improves upon this by constructing a hierarchical structure that encompasses all possible ϵ values.

There are two important input parameters for this algorithm. The first, $MinPts$, is used to determine core distances and is key for calculating density, similarly to its function in DBSCAN. The second, referred to as the minimum cluster size, sets the lower limit on the number of data points necessary for a group of points to be considered as a valid cluster [SA22].

An overview of the HDBSCAN algorithm is given in Fig. 3.5. The first step in HDBSCAN involves transforming the feature space using a metric called mutual reachability distance, which incorporates density into the distance calculation between points. For any two data points p and q, the mutual reachability distance is defined as:

$$dist_{mreach}(p, q) = \max\{dist_{core}(p), dist_{core}(q), dist(p, q)\} \tag{3.11}$$

where $dist(p, q)$ is the distance (e.g. Euclidean distance) between points p and q, while $dist_{core}$ is the distance from p to its $MinPts$-nearest neighbor (including p) [MB20]. These mutual reachability distances are used to construct a weighted graph of the data set. In this graph, data points are vertices connected by edges weighted according to the mutual reachability distances [MB20]. This graph is then used to construct a minimum spanning tree whose edges' weight sum is minimal while still connecting all vertices without forming cycles. The minimum spanning tree is extended by adding a self-edge to each vertex weighted by its core distance [SA22]. Sorting the edges of this tree by mutual reachability distance in ascending order leads to the formation of a hierarchical tree known as a dendrogram, which represents clusters at different values of ϵ. At the root of the dendrogram lies a cluster that includes all data points [Wan+21]. In DBSCAN*, clusters can be selected using a fixed ϵ value with respect to $MinPts$, as a horizontal cut point. HDBSCAN, however, identifies clusters of varying density without relying on a single ϵ value.

The first step in cluster extraction by HDBSCAN is to condense the hierarchy into a smaller tree, known as the condensed cluster tree. Beginning at the root, HDBSCAN regards a cluster split as true only if both resulting child clusters have at least the minimum cluster size. If neither child cluster meets this requirement, the cluster is considered to have vanished at that density level [MB20]. In cases where only one child cluster has fewer points than the minimum cluster size, this is interpreted as the parent cluster losing

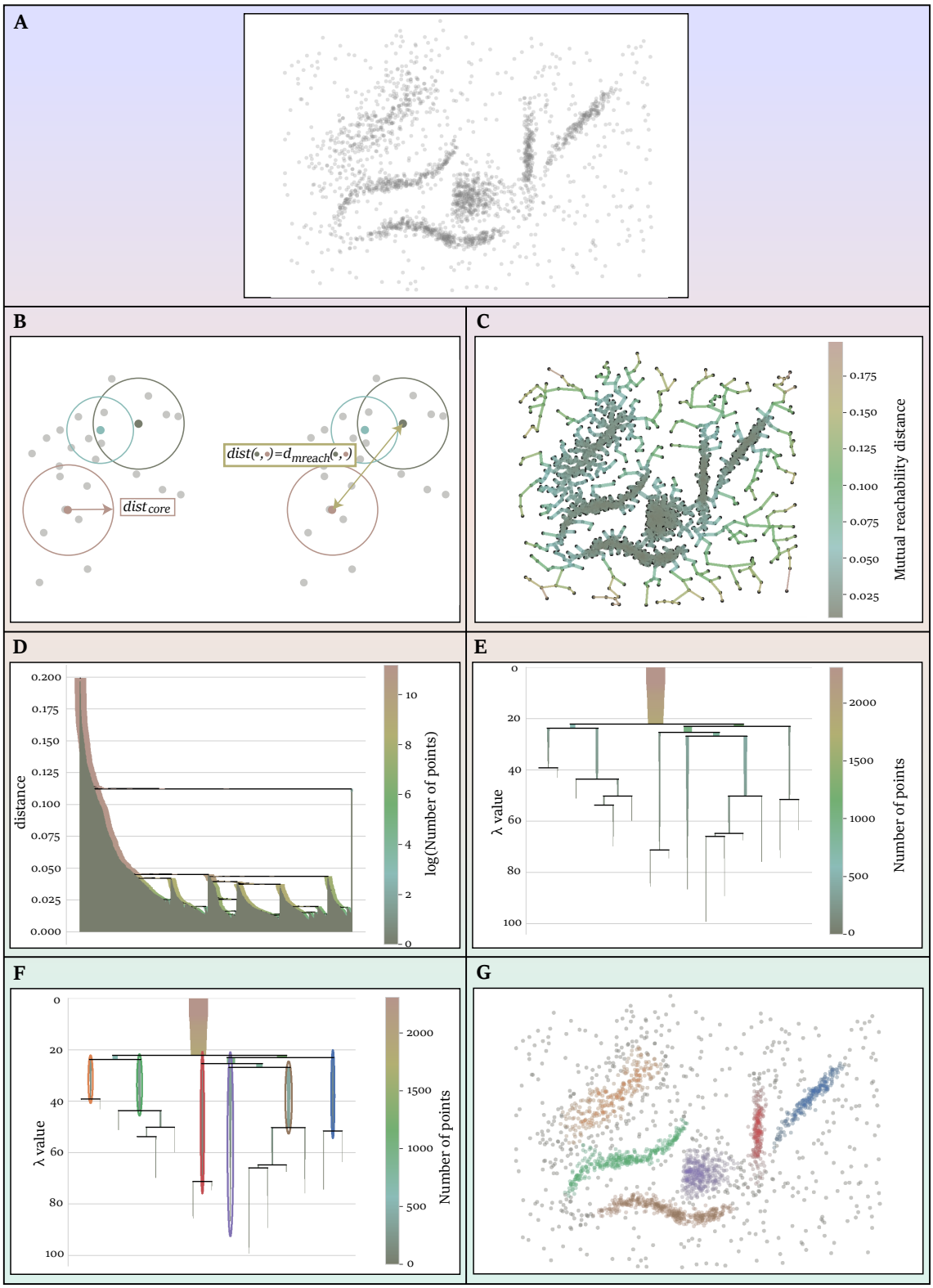


Figure 3.5.: Overview of the HDBSCAN clustering algorithm. (A) Data set for clustering from [McI]. (B) Visual illustration of mutual reachability distance. For simplicity, only a subset of the data set is shown. The mutual reachability distance between the rose-colored and grey points is equal to the distance from the rose-colored point to the grey point, as this distance is greater than either core distance. (C) Construction of the minimum spanning tree of the distance-weighted graph. (D) Building of the cluster hierarchy. (E) Condensation of the cluster tree. (F) Extraction of clusters. (G) Display of the resulting clusters with minimum cluster size set to 15, as in [MHA23]. The figures were created based on the code from [MHA23].

a few points that are regarded as noise. This process leads to a hierarchy of potential clusters across different density levels [MB20].

From the condensed tree, the algorithm offers two methods for selecting clusters: the excess of mass (EoM) method and the leaf approach. Each serves a distinct purpose: the EoM method (see F in Fig. 3.5) assigns a stability score to each cluster, calculated using the following equation:

$$\sum_{p \in \text{cluster}} (\lambda_p - \lambda_{\text{birth}}) \tag{3.12}$$

where λ_{birth} represents the threshold value of $\lambda = \frac{1}{\text{edge weight}}$ at which a new cluster is created from a cluster split [SA22]. For each point p within a cluster, λ_p is the threshold value at which the point falls out of the cluster. This value, λ_p, lies between λ_{birth} and λ_{death} because a point either falls out of the cluster during its existence or leaves when the cluster divides into two smaller sub-clusters [MHA23]. A high stability score indicates that a cluster exists over a wide range of ϵ values [PBC21], and clusters with strong stability scores are then retained. The leaf method, on the other hand, can reveal more fine-grained clusters [HR21] than EoM, as it selects all leaf nodes (end points or the tree structure), representing clusters with the lowest ϵ values in the hierarchy that cannot be further divided [MB20]. This approach was later chosen in the thesis to prevent the formation of very large clusters and to ensure that smaller, yet meaningful clusters are detected, as detailed in Section 4.3.2.

3.2. Deep learning

The principles of supervised and unsupervised learning in ML are also foundational to DL. DL leverages multilayered (deep) networks, which are used to map relationships between input and outcome variables [MMC22]. Originially inspired by the brain's neural architecture, DL is concerned with algorithms that process information through hierarchical layers to learn representations and features from data at increasing levels of complexity. Consequently, the term deep refers to the number of layers in the model [MMC22], and DL is highly effective at extracting features and learning patterns from complex data. To further comprehend how a DL model learns and is evaluated, the following sections will delve into this topic in more detail.

3.2.1. Feed-forward neural network

The feed-forward neural network, or multilayer perceptron (MLP), is considered the fundamental building block of DL. At its core, a perceptron functions as an artificial neuron [SB19]. As shown in Fig. 3.6A, each input is multiplied by a corresponding weight, reflecting the input's influence on the output [SB19]. The weighted inputs are summed, a bias term is added, and this total is passed through an activation function, resulting in the perceptron's output.

When one or more hidden layers are added between the input layer and output layer, the network becomes a MLP, as visualized in Fig. 3.6B. Each layer is composed of units (neurons), with each neuron (fully) connected to the neurons in both the preceding layer and following layer. The neurons in the input layer represent the input features. The information, or signals, flow sequentially in one direction through each layer, from input to output, without forming any loops (feed-forward) [Pop+09]. Once a neuron receives its inputs from the preceding layer, the same calculation described in Fig. 3.6A is performed. The results are then propagated to the next layer, where the process is repeated until the results reach the output layer.

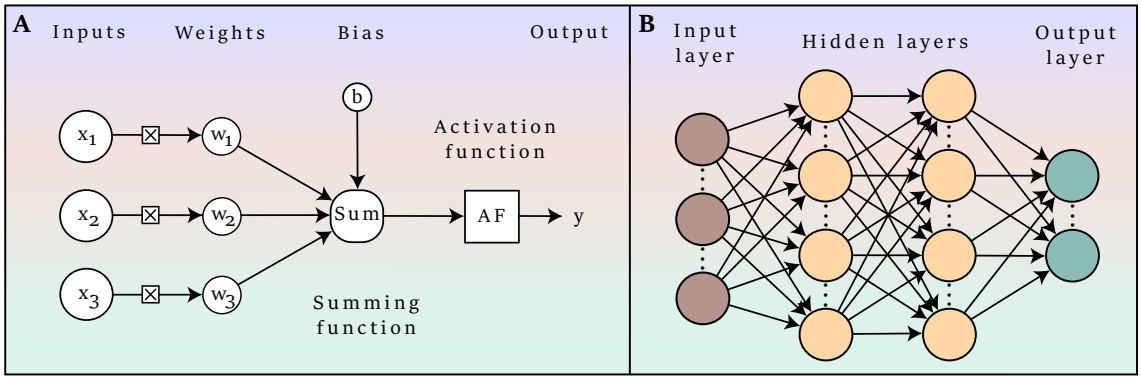


Figure 3.6.: Schematic visualization of a feed-forward neural network. (A) Example of a perceptron. Each input to a neuron is associated with a weight. The inputs are multiplied by their respective weights, a bias is added, and the resulting weighted sum is passed through a non-linear activation function. (B) Example of a MLP. The leftmost layer is the input layer consisting of a set of neurons representing the input features. This is followed by two hidden layers and an output layer. Each neuron is connected to all neurons of the previous layer and the next layer.

The activation function is a crucial component that introduces non-linearity into the model to learn more complex patterns. Among the most popular activation functions are the rectified linear unit (ReLU) and the sigmoid activation function [DQZ18]. The ReLU function is defined as:

$$f_{\mathrm{ReLU}}(x) = \max\{0, x\}. \tag{3.13}$$

This formula shows that if the input x is greater than zero, the output equals the input; otherwise, it outputs zero. As depicted in Fig. 3.7A, the ReLU activation function is flat for inputs less than zero and linear, with no upper bound, for inputs greater than zero. This function is frequently used due to its simplicity and computational efficiency, as it avoids the need to compute exponential functions during activation [DQZ18]. The definition of the sigmoid function, on the other hand, is as follows:

$$f_{\text{sigmoid}}(x) = \frac{1}{1 + \exp(-x)}. \tag{3.14}$$

The sigmoid function constrains output values to a range between 0 and 1, which can be useful when interpreting outputs as probabilities, particularly in classification problems [MH20]. However, when the input values are strongly positive or negative, the output values will be very close to 1 or 0, as shown in Fig. 3.7B. Both the ReLU and sigmoid activation functions are defined for a scalar input $x \in \mathbb{R}$.

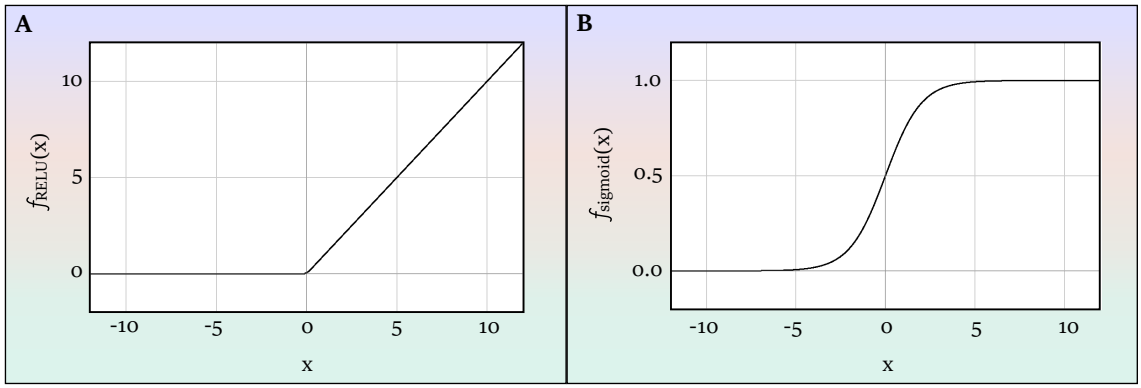


Figure 3.7.: Activation functions of DL. Activation functions transform the weighted sum of inputs that goes into the neurons. Among the most popular functions are ReLU and sigmoid. (A) Representation of the ReLU activation function. (B) Representation of the sigmoid function. Abbreviations: ReLU: Rectified linear unit.

To effectively use a neural network, whether deep or shallow, it must first be trained on data to adjust its learnable parameters θ_i - its weights and biases - in order to minimize the error between the predicted and desired outputs, as measured by a loss function. This optimization is typically achieved through backpropagation and gradient descent [KM21]. Backpropagation is a training algorithm and calculates the gradients of the loss with respect to the learnable parameters by propagating the error backward through the network. The parameters are then iteratively updated using gradient descent to minimize the loss to reach the minimum [KM21]. In the following, the most commonly used loss functions and gradient-based optimization methods for training neural networks are introduced and described.

3.2.2. Loss functions

The choice of the loss function typically depends on the model's architecture and the specific task. The mean squared error (MSE) loss function is one of the most widely used loss functions, particularly in regression tasks [Ter+25], and is calculated by taking the mean of the squared differences between predicted values and the target values:

$$\mathcal{L}_{\text{MSE}}(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2, \tag{3.15}$$

where N denotes the number of samples in the data set, y_i is the target value for the i-th sample, and $\hat{y}_i$ is the predicted value for the i-th sample. Because squared errors cannot be negative, the MSE produces a value ≥ 0. A value of zero indicates a perfect fit [Ter+25]. MSE disproportionately penalizes large errors by squaring them, which helps to reduce the influence of outlier predictions with large errors.

Another loss function commonly used in regression tasks is the mean absolute error (MAE). Although structurally similar to MSE, MAE computes the average absolute difference between predicted and target values. Compared to MSE, MAE is generally less sensitive to outliers [Elh+25] because it does not square the individual errors:

$$\mathcal{L}_{\text{MAE}}(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{3.16}$$

When it comes to classification problems, the most common loss function is cross-entropy (CE), which aims to minimize the difference between predicted probabilities and true probabilities (i.e., GT labels). This cost function is related to the Kullback-Leibler divergence [KL51] and to Shannon entropy [Sha48]. The Kullback-Leibler divergence measures the difference between two probability distributions [Cui+24] and is defined as:

$$KL(P, Q) = \sum_{x \in X} P(x) \cdot \log \frac{P(x)}{Q(x)}, \tag{3.17}$$

where $Q(x)$ is the estimated distribution and $P(x)$ is the true distribution of the data. The closer the Kullback-Leibler divergence gets to zero, the more similar the distributions are considered to be. The value is always non-negative, meaning that the divergence between these two probability distributions can never be less than zero. If it is zero, the

distributions match exactly [Shl14]. Shannon's information entropy is used to measure the uncertainty or randomness of a distribution:

$$H(x) = -\sum_{x \in X} P(x) \cdot \log P(x). \tag{3.18}$$

The lower the entropy, the more certainty is attributed to the outcome. Building upon the entropy of P and the Kullback-Leibler divergence between P and Q, CE is defined over the discrete variable x as follows:

$$\mathcal{L}_{\text{CE}}(P, Q) = -\sum_{x \in X} P(x) \cdot \log Q(x), \tag{3.19}$$

where P represents the true distribution of the targets across classes and Q represents the predicted likelihood for each class, commonly given by the softmax layer output [MS18]. The softmax activation function converts the raw output scores (logits) produced by the final layer into a normalized probability distribution over all possible classes [MLP24; Wan+18a]. Given a vector of logits z, where K is the total number of classes and z_i is the logit associated with class i, the softmax function for class i is defined as:

$$f_{\text{softmax}}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{K} \exp(z_j)}, \tag{3.20}$$

where j indexes over all possible classes from 1 to K. The output values of the softmax function are non-negative and sum to one [TD24], allowing them to be interpreted as a valid probability distribution.

The CE loss increases as the predicted probability differs from the correct label (GT). A special variation of the CE loss function is the binary CE (BCE) loss function, where there are two possible classes (1 for the positive class, 0 for the negative class) for each output. The BCE loss function is calculated as:

$$\mathcal{L}_{\text{BCE}}(P, Q) = -\sum_{x \in X} [P(x) \cdot \log Q(x) + (1 - P(x)) \cdot \log(1 - Q(x))] \tag{3.21}$$

where $P(x)$ represents the true probability of event x being the positive class, and $Q(x)$ represents the predicted probability of x being the positive class. Analogous to the formulations of the MSE and MAE loss functions (Eq. 3.15 and Eq. 3.16, respectively), the BCE loss for N training samples is:

$$\mathcal{L}_{\text{BCE}}(y, \hat{y}) = -\frac{1}{N} \sum_{i=1}^{N} [y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log(1 - \hat{y}_i)], \tag{3.22}$$

where $y_i \in \{0, 1\}$ denotes the target class label for the i-th sample and $\hat{y}_i \in [0, 1]$ is the predicted probability value for that sample [Yeu+22]. The BCE loss function penalizes the model more heavily when it assigns a low probability to the true class.

While the standard BCE loss expects the model to output probabilities $\hat{y}_i \in [0, 1]$, typically after a sigmoid activation, it is often preferable to combine both operations into a single step for improved numerical stability during training [KT21], particularly when dealing with exponential and logarithmic computations. By substituting the sigmoid function value (Eq. 3.14) directly in place of $\hat{y}_i$ in the BCE loss, the BCE with logits loss is obtained:

$$\mathcal{L}_{\text{BCE-logits}}(y, z) = -\frac{1}{N} \sum_{i=1}^{N} \left[y_i \cdot \log f_{\text{sigmoid}}(z_i) + (1 - y_i) \cdot \log(1 - f_{\text{sigmoid}}(z_i)) \right], \quad (3.23)$$

where z_i is the logit output (raw score) for the i-th sample, and $f_{\text{sigmoid}}(z_i)$ is the sigmoid-activated output.

3.2.3. Gradient descent optimization

Choosing the right loss function is crucial, as its minimization allows the optimization of a model's parameters $\boldsymbol{\theta}$. Gradient descent, an optimization algorithm, is used to find the optimal parameters by iteratively adjusting the weights and biases of the neural network [MMA20], as already mentioned in Section 3.2.1. To consider the impact each parameter θ_i has on the final prediction, the partial derivatives of the loss function with respect to each parameter θ_i of $\boldsymbol{\theta}$ are stored in the gradient. The parameters are adjusted in the direction opposite to the gradient to reduce the loss function. Backpropagation is an efficient approach to compute the gradients in a recursive manner - from the output layer backward through the network, which allows for the iterative updating of the model's parameters given some training data [Zha19].

In the following, three fundamental variants of gradient descent [TZZ23] are presented, which differ in the quantity of data used to compute the gradient of the objective function [MMA20]. These include batch gradient descent, stochastic gradient descent (SGD), and mini-batch gradient descent.

Batch gradient descent computes the gradient of the loss function with respect to the parameters $\boldsymbol{\theta}$ for the entire training data set, allowing for a precise gradient estimate. However, for large data sets, this variant requires significant computational and memory resources, since it can take a long time to converge to the optimal solution.

SGD, on the other hand, randomly selects one data point from the data set in each iteration to compute the gradient [WYZ21], thereby avoiding the need to calculate it over

the full data set. This allows for quicker iterations and introduces noise in the updates. The noise can help mitigate the risk of getting stuck in local minima. The update rule is defined as:

$$\theta = \theta - \eta \cdot \nabla_\theta \mathcal{L}\left(\theta; x^{(i)}, y^{(i)}\right), \tag{3.24}$$

where $x^{(i)}$ is the training example, $y^{(i)}$ the label, and η a hyperparameter called the learning rate [Rud16].

The mini-batch gradient descent algorithm splits the training data set into small batches that are used to calculate model error and update the model. This variant reduces the variance in the estimate of the gradient and combines the advantages of both batch and stochastic methods, but the batch size is a critical hyperparameter [Zha19].

The learning rate η is a key hyperparameter applicable to all variants of gradient descent. It is the size of the step taken in the direction of the negative gradient and affects the convergence of gradient descent algorithms [HA21]. A small learning rate results in slow convergence, whereas a high value can cause the loss function to fluctuate around the minimum [SS23]. However, in many cases, fixed learning rates do not perform well throughout the entire optimization process for gradient descent algorithms. At the initial stages, a larger learning rate is often beneficial for quickly approaching an optimal solution, either local or global. In later stages, adjusting the learning rate becomes necessary [HA21].

To mitigate fluctuations in the learning process associated with gradient descent algorithms, momentum was introduced to speed up convergence and reduce errors associated with the steepest descent [HKS16]. Instead of updating the parameters based on the current gradient directly, this optimizer employs the exponentially weighted (discounted) cumulative sum of the previous gradients [SA18], allowing it to advance in the same direction as previously and smoothing the path towards convergence. Momentum is often used with SGD and adaptive learning rate methods such as adaptive momentum estimation (Adam) among others [HA21].

3.2.4. Model evaluation

Finding the optimal set of model parameters is one aspect; another critical consideration is evaluating the performance of the trained neural network and assessing its ability to generalize to unknown or unseen data. In order to do so, splitting the data set is essential for reducing bias toward the training data and avoiding overfitting [Mur22]. Data splitting is the partitioning of data into subsets for model training and evaluation separately. Usually, a train-validation-test split is conducted, and various ratios may be

used. In general, the training set is used for model fitting, the validation set for model selection, and the test set for final model assessment [Mur22].

Data splitting

One of the most widely used data splitting techniques is cross-validation (CV) [XG18]. In this approach, the data set is divided into k distinct subsets, known as folds. During each iteration, one fold serves as the validation set, while the remaining k-1 folds are utilized to train the model. This process is repeated k times, ensuring that each fold serves as the validation set exactly once. The overall performance of the model is then calculated as the average of the evaluation metrics obtained from each fold during the CV process. When dealing with imbalanced data, stratified k-fold CV might be a preferable alternative. In this method, each fold preserves the proportion of the target classes found in the overall data set.

Another widely used data splitting technique, in addition to k fold CV, is Monte Carlo (MC) CV. In contrast to k fold CV, MCCV generates each split randomly without replacement [XG18]. For each split, the data set is divided into a predetermined training set and a validation set with the remaining samples. While k fold CV uses k non-overlapping partitions - ensuring each observation serves exactly once as validation - MCCV allows for a much larger number of possible random partitions and is typically repeated many more times. However, the computational cost of MCCV increases with the number of iterations [MSP05]. Additionally, its random sampling can result in some observations being underrepresented or never included in the training or validation sets, potentially introducing bias to the performance estimate [MSP05].

Evaluation metrics

To evaluate a model's performance and to obtain the optimal model, suitable evaluation metrics for the given task must be selected. As this thesis primarily addresses classification problems, the evaluation metrics commonly used in this context will be introduced.

In binary classification tasks, each data instance is predicted as belonging to either the positive or the negative class, for example, the presence or absence of a disease. With the following 2×2 confusion matrix in Table 3.1, each prediction can be categorized into one of four outcomes:

	Predicted class	
Actual class	Positive	Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

Table 3.1.: Confusion matrix for binary classification. The rows represent the actual (true) class, while the columns correspond to the predicted class.

A true positive (TP) refers to a positive case that is correctly identified as such, while a true negative (TN) is a negative case that is correctly identified. In contrast, a false positive (FP) occurs when a negative case is incorrectly classified as positive, and a false negative (FN) denotes a positive case that is mistakenly classified as negative. Based on these four outcomes, a variety of informative evaluation metrics can be calculated:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \quad \text{Accuracy} \in [0, 1] \tag{3.25}$$

$$\text{Sensitivity} = \text{Recall} = \frac{TP}{TP + FN}, \quad \text{Sensitivity} \in [0, 1] \tag{3.26}$$

$$\text{Specificity} = \frac{TN}{TN + FP}, \quad \text{Specificity} \in [0, 1] \tag{3.27}$$

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Precision} \in [0, 1] \tag{3.28}$$

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \quad \text{F1} \in [0, 1]. \tag{3.29}$$

Sensitivity (or recall) is also referred to as the true positive rate, specificity as the true negative rate, and precision as the positive predictive value. Except for accuracy, these metrics are typically considered in pairs, such as precision with recall or sensitivity with specificity [RTK24]. Sensitivity and specificity often provide a more nuanced understanding of model performance than accuracy alone, particularly in situations where the distribution of positive and negative cases is highly imbalanced. Another important metric is the F1 score that is the harmonic mean of precision and recall [DEA24] and is often used to assess diagnostic performance of prediction algorithms [DeV+21].

3.2.5. Convolutional neural networks

When it comes to classifying 2D images - where grid-structured inputs exhibit spatial dependencies among neighboring pixels (as adjacent pixels often share similar color values) - standard feed forward networks composed solely of fully connected layers are unable to capture spatial information [Pin+20]. CNNs, on the other hand, are specifically designed to process grid-structured inputs and learn hierarchical spatial features, from low- to high-dimensional level patterns [Yam+18].

The core component of a CNN is the convolutional layer, which gives the network its name and is a type of linear mathematical operation used for feature extraction. Each convolutional layer contains a set of filters (kernels), which are typically square-shaped, ranging from 3×3 up to 7×7 grids of discrete values [Pin+20]. These values are learnable parameters that are updated during the training process through backpropagation, so that relevant patterns or features in the input data are captured. The number of kernels used in a layer is another hyperparameter, often chosen as a power of 2, commonly between 32 and 512. These kernels are slid across the input array (tensor) from left to right and top to bottom with a specific stride size [Pin+20], computing how similar the kernel is to each corresponding region; higher resulting values indicate a stronger correlation between the kernel and that region of the input [Ket+21]. Mathematically, this operation is performed by calculating the element-wise dot product between the kernel's parameters and the sub-region of the input data, followed by summing these products to obtain a single number. This value is then stored in the corresponding position of the feature map. By convolving over all spatial locations in the input image, the feature map preserves the spatial relationships present in the original grid [Pin+20]. To prevent the feature map from shrinking during the convolution operation, the layer's input can be padded with zero-value pixels around the border [Pin+20].

The convolution procedure is usually repeated with multiple kernels to capture different characteristics and patterns of the image. The resulting feature maps are then stacked together to form the output of the convolutional layer [Pin+20]. The outputs of a convolution are then passed through a nonlinear activation function, as explained in Section 3.2.1.

The described localized operation sets CNNs apart from fully connected networks: whereas each neuron in a fully connected layer is connected to every neuron in the previous layer, each neuron in a convolutional layer is connected only to a limited subset. As a result, each position in the feature map is associated with a neuron with a receptive field [Pin+20]. When multiple convolutional layers are stacked, for example two consecutive layers, each with a 3×3 filter, the receptive field of a neuron in the

second layer is still 3×3. However, its effective receptive field with respect to the original input image expands to 5×5, allowing the network to learn more complex spatial dependencies [Pin+20]. The local connections reduce the number of parameters and can lead to faster convergence during training [Li+21]. Another property is weight sharing: each neuron within the same feature map shares the same weights, meaning that kernel weights are applied at multiple locations across the input, which further reduces memory requirements. Furthermore, if the input image is shifted, the resulting feature map also shifts by the same amount. The shared neuron's weights (parameters of each kernel) are usually randomly initialized (by sampling from a probability distribution) at the beginning and then learned during the training phase using a gradient descent method [Pin+20].

So far, the description above has assumed that the input image is represented as a single 2D matrix. However, fluorescence microscopy images, such as those used in this thesis, typically consist of multiple channels (in this case, three), with each channel (a 2D matrix) corresponding to a specific fluorescent marker. These channel images are stacked to form a 3D input tensor. When performing convolution on such multi-channel inputs, each kernel also spans all channels of the input, maintaining the same spatial dimensions as before.

Besides convolutional layers, typical CNN architectures include other building blocks, such as pooling layers and fully connected layers [Yam+18], as depicted in Fig. 3.8. Pooling (subsampling) [Sun+17] is typically applied after convolutional layers and serves

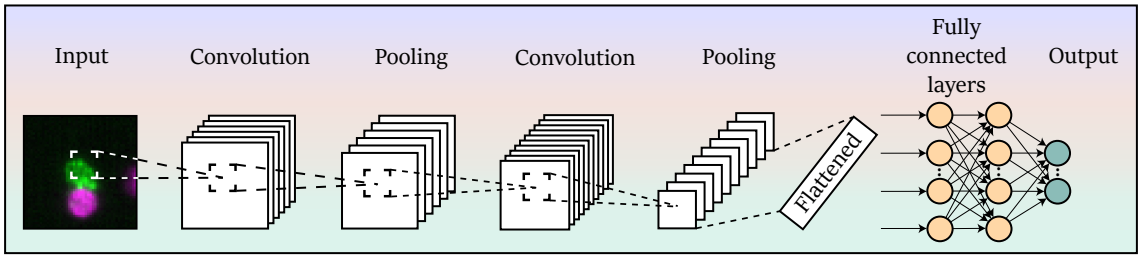


Figure 3.8.: Schematic overview of a CNN architecture. A CNN is composed of a stack of mutliple building blocks: convolutional layers, pooling layers, and fully connected layers.

to decrease the dimensionality of feature maps by dividing them into fixed-size regions [ZZ24] and retaining the most important information from each region [GK20]. This process reduces the spatial resolution of feature maps, lowers redundancy [ZZ24], and provides translational invariance [Nir+22], making the CNN less sensitive to the absolute position of features within the input. Among the most common pooling operations is max pooling, which slides a filter across its input and selects the maximum value from each region of the feature map covered by the filter, discarding all other values. Thus, the output is a feature map that retains only the most prominent features from

the previous layer, such as edges and textures. Similar to convolutional layers, the filter size (typically 2×2) and stride (usually 2) can be specified [ON15], the depth of the feature map is retained [Yam+18]. Another common form is average pooling, which computes the average of elements within the filter region, resulting in a more generalized representation of the input.

For classification tasks, the output feature maps from the last convolution or pooling layer are typically flattened into a 1D vector and passed to one or more fully connected (dense) layers [Alz+21]. In these layers, every input is connected to every output through a learnable weight [Yam+18]. The extracted features are mapped by these fully connected layers to the final outputs [Yam+18]. The choice of activation function in the output layer depends on the nature of the classification problem. For binary classification, a sigmoid activation function is typically used to generate a single probability value [Nwa+18]. For multiclass classification, a softmax activation function is applied to compute the probability distribution over the possible classes, with the sum of these probabilities equal to one [Nwa+18].

3.2.6. Techniques to improve performance

Compared to the initially achieved performance during training, neural networks often exhibit reduced accuracy when evaluated on unseen test data, but there are methods to improve generalizability, such as data augmentation and dropout.

Data augmentation

Training deep neural networks with limited data can lead to overfitting, causing an increase in generalization error. In the context of image data, the core idea of data augmentation is to increase the diversity of the training data by applying geometric and value-based transformations to the input data. Examples of geometric transformations include flipping, rotation, cropping, and zooming. Value-based transformations are techniques such as color jittering, adding noise, and adjusting image sharpness or blurriness, among others. The augmented data can be considered as samples drawn from a distribution that closely resembles the true data distribution [Yan+22].

Dropout

Networks with a large number of trainable parameters are more prone to overfitting [SK23a]. To address this, dropout is used as a regularization method by setting the activation values of randomly chosen neurons to zero at each iteration during training [SK19; WWL13], reducing the network's reliance on specific neurons and encouraging the

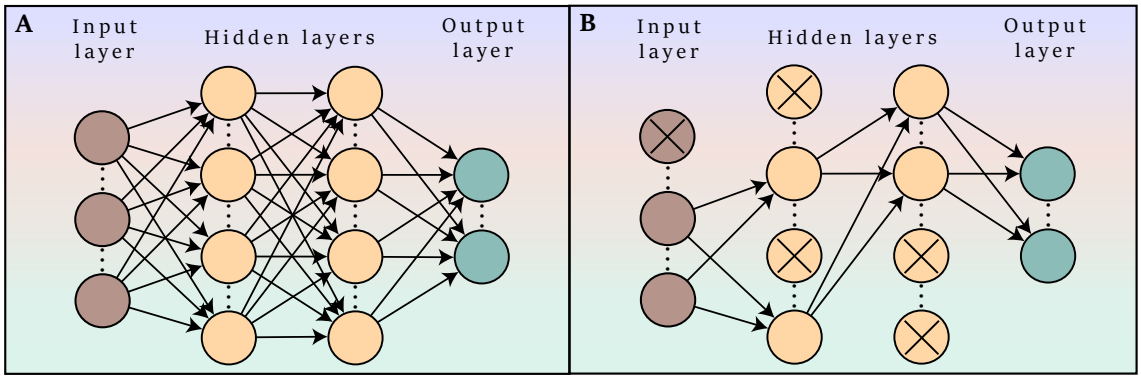


Figure 3.9.: Schematic visualization of a standard dropout. (A) The left network is fully connected. (B) The right network has neurons from the input layer and from the hidden layers dropped with a probability p. Neurons from the output layer are not dropped.

learning of more generalizable features. Dropout can be applied to both input and hidden layers, and the probability p of dropping a neuron can vary between layers [LSV19]. However, neurons in the output layer are not subjected to dropout, as shown in Fig. 3.9. When a neuron is dropped out, its connections are temporarily removed for the duration of that training iteration.

3.3. Image segmentation

Alongside classification, image segmentation is a central focus of this thesis and will be introduced in this section. Image segmentation refers to the division of an image into multiple regions (segments), each corresponding to a specific object, structure, or area of interest [Ray+24]. The goal is to assign a label to each pixel in the image such that pixels with similar characteristics or that belong to the same object are grouped together. This pixel-level classification enables a more detailed analysis than standard image classification, which assigns a single label to the entire input image. In practice, image segmentation is crucial in many applications, including the medical field. For example, in tumor detection on MRI scans, it is not sufficient to only determine whether a tumor is present; it is often crucial to precisely locate and segment cancerous areas within the tissue. Similarly, in fluorescence microscopy, it is important to segment individual cells in the images before proceeding with their classification. There are three primary types of image segmentation: semantic segmentation, which assigns a class label to each pixel; instance segmentation, which distinguishes and labels individual instances of objects belonging to the same class; and panoptic segmentation, which combines both approaches. With the rapid advancement of DL algorithms, image segmentation performance has seen improvement. In the biomedical domain, CNNs have been particularly influential for this task.

U-Net

The most widely used CNN architecture for the task of (bio)medical image segmentation is the U-Net architecture, introduced by Ronneberger et al. [RFB15]. Since its initial proposal for semantic segmentation, U-Net has been applied across a wide range of imaging modalities, including CT, MRI, and microscopy images [Sid+21]. The architecture is based on a fully convolutional network arranged in an encoder-decoder fashion [Aza+24]. It is named U-Net due to the characteristic U-shaped structure formed by the downsampling (encoder) and upsampling (decoder) paths within the network [Liu+19], as shown in Fig. 3.10. Unless otherwise noted, the following description refers to the base U-Net architecture as introduced by Ronneberger et al. [RFB15].

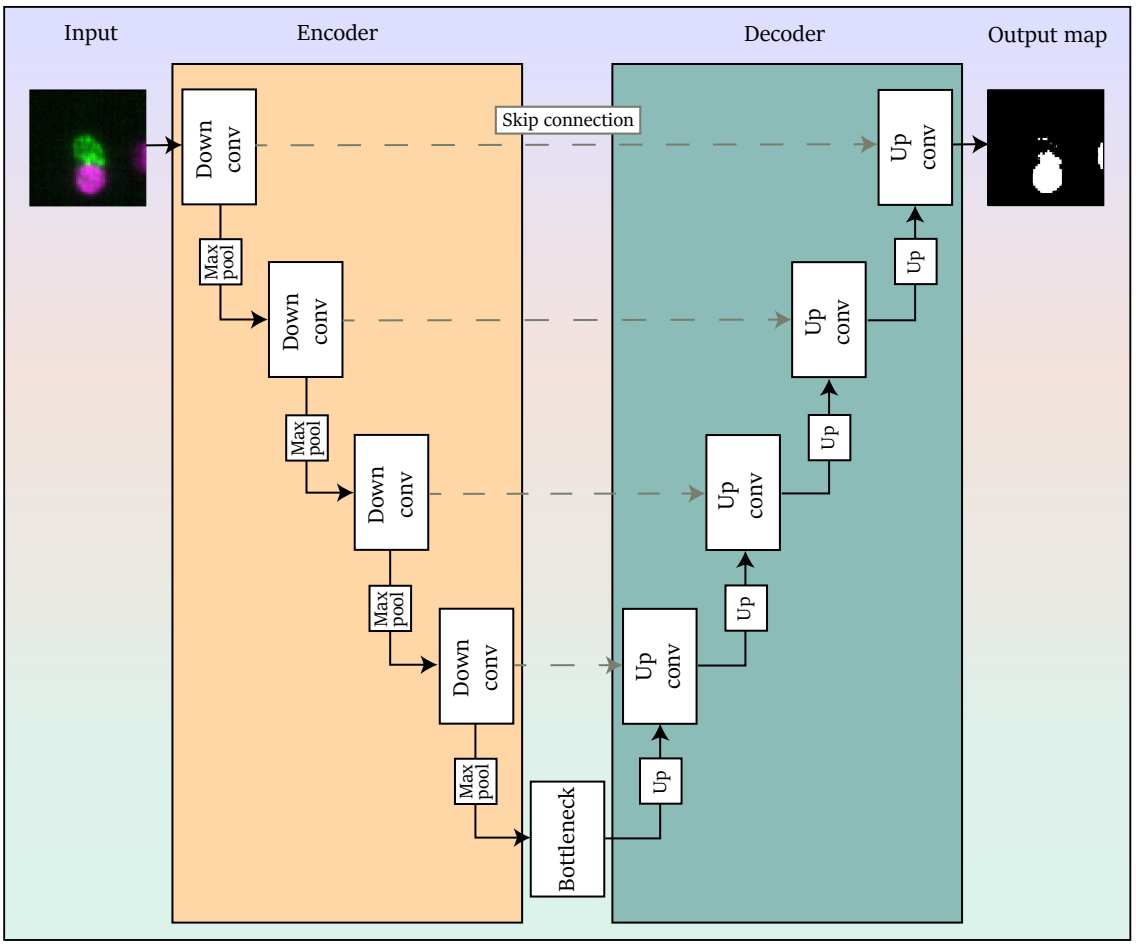


Figure 3.10.: Schematic overview of a U-Net architecture, based on [RFB15]. As illustrated, U-Net follows an encoder-decoder structure: the encoder consists of repeated blocks of convolutional layers and max pooling (Down conv), which progressively reduce spatial dimensions and extract features, leading to the bottleneck. The decoder path features upsampling and convolutional operations at each corresponding level (Up conv), eventually producing the final output segmentation map. At each level, skip connections are used to concatenate cropped feature maps from the encoder with those in the decoder. The number of encoder-decoder blocks can be adjusted to suit the input data and specific task. Abbreviations: Conv: Convolution.

When an image is fed into the the encoder, it passes through repeated blocks, each comprising two successive 3×3 convolutions followed by a ReLU activation and a max-pooling layer with a 2×2 kernel and a stride 2 for downsampling. At each downsampling step, the number of feature channels is doubled, while the spatial resolution is reduced. The encoding process gradually extracts features and captures contextual information [Aza+24], continuing until the input reaches the bottleneck (two successive 3×3 convolutions followed by a ReLU activation) of the network. At the bottleneck, the feature map achieves its lowest spatial resolution and typically has the highest number of channels. In the decoder path, the feature map is gradually upsampled at each level using 2×2 up-convolutions, which double the spatial dimensions and halve the number of feature channels [RFB15]. Each level in the encoder path is connected to the corresponding spatial level in the decoder by so-called skip connections. These connections help preserve high-level semantic information as well as finer spatial details [PSC23]. At each upsampling stage, the corresponding feature map from the encoder is cropped and concatenated with the upsampling feature map [RFB15; Sid+21]. The concatenated output then passes through two consecutive 3×3 convolutions followed by a ReLU activation. At the final layer, a 1x1 convolution is applied to map the feature map to the desired number of classes [Sid+21; RFB15], producing the segmented image.

3.4. Advanced AI models

Within the broader family of CNN architectures, beyond the U-Net, numerous other specialized models have been developed for various tasks. While these models share core principles, they differ in their architecture, operational mechanisms, and performance [Agg+23], as well as in their application domains. Many of these models, such as EfficientNet, have been employed successfully for (medical) image classification [Beh+24; Raz+23]. EfficientNet builds upon CNN principles but introduces an optimized scaling method, known as compound scaling, which efficiently scales the model up in three dimensions: depth (number of layers), width (number of filters per layer), and resolution of input images [TL19; Agg+23], achieving a good balance between performance and computational cost.

Despite these architectural advancements and their significant impact on the field of medical imaging and classification, CNNs predominantly rely on convolutional operations. These operations typically focus on local regions of the image through small receptive fields, which restrict the model's ability to capture long-range pixel relationships within the input image. This highlights the need for alternative architectures that

can better capture these dependencies, while maintaining optimal performance for a given computational cost or number of parameters.

3.4.1. Transformer

Transformers, introduced by Vaswani et al. [Vas+17], employ a self-attention mechanism to capture non-local relationships between all input sequence elements, referred to as tokens [Kha+22]. This approach has significantly advanced the field of natural language processing. In NLP, tokenization is the process of splitting a text into basic units [Dot+24] - such as words, subwords, or characters - which are then used as the model's tokens. Each token is mapped to an embedding - a numeric vector representation [Lin+17] - which encodes semantic or contextual information about the token. These embeddings form the input matrix $X \in \mathbb{R}^{n \times d}$ for the transformer model, where n is the sequence length and d is the embedding dimension of each token.

Transformers efficiently handle sequence-to-sequence tasks, such as transforming one type of sequence (for example, a text) into another (e.g., a translation). They are highly scalable and enable the training of models with hundreds of billions of parameters without performance saturation [MDB23]. The key mechanism in transfomers is self-attention, whose aim is to model the interactions among all n tokens by encoding each token in terms of global contextual information from the entire sequence [Kha+22]. This is achieved by mapping the input sequence X into three matrices - query Q, key K, and value V - using learnable weight matrices $W^Q \in \mathbb{R}^{d \times d_q}$, $W^K \in \mathbb{R}^{d \times d_k}$, and $W^V \in \mathbb{R}^{d \times d_v}$, where $d_q = d_k$ [Kha+22], so that:

$$Q = XW^Q, \quad K = XW^K, \quad V = XW^V. \tag{3.30}$$

The self-attention mechanism proceeds by computing the attention weights via the scaled dot-product attention [Vas+17], formulated as:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{3.31}$$

For each token in the sequence, self-attention first computes the dot product between its query and all keys, then scales the result by dividing by $\sqrt{d_k}$. The softmax is then applied to obtain the attention weights. These weights are used to calculate a weighted sum of the value vectors to control their contributions in an attention block [Lov+22]. Instead of relying on a single attention mechanism, transformers employ multi-head attention [Vas+17], in which the input is partitioned into multiple parallel self-attention heads,

allowing the simultaneous capture of multiple complex relationships between different tokens in the sequence.

The key distinction between self-attention and convolution operations is that, in self-attention, the filters are generated dynamically based on the input itself, whereas convolution uses fixed filters that remain unchanged regardless of the input [Kha+22].

The transformer architecture described in [Vas+17] is depicted in Fig. 3.11 and has an encoder-decoder structure. To enable the model to incorporate sequence order, positional encodings - which provide either relative or absolute position information for the tokens - are added to the input embeddings at the base of the encoder and decoder stacks [Kha+22]. The encoder consists of six identical blocks, each composed of two sub-layers: a multi-head self-attention mechanism responsible for contextual aggregation, and a position-wise fully connected feed-forward network (i.e., a feed-forward neural network applied separately and identically to each position in the sequence) for transformation and extraction of feature representations [Kha+22; Liu+23]. Additionally, residual connections and layer normalization are applied after each sub-layer. A residual connection allows information to flow through a shortcut (see Fig. 3.11) by adding the input of a layer to its output. This helps to improve performance [Han+22] and mitigates the vanishing gradient problem [Isl+24].

The decoder also consists of six identical blocks; however, each decoder block contains three sub-layers: a masked multi-head self-attention mechanism, a multi-head cross-attention mechanism, and a position-wise feed-forward network. Cross-attention acts as a bridge between the encoder and decoder by allowing the decoder's queries (which originate from the output of the masked multi-head attention sub-layer in the decoder) to attend to the keys and values derived from the encoder's output [GG20; Liu+23]. Thus, cross-attention differs from self-attention in that the queries, keys, and values do not originate from the same sequence. This mechanism allows the decoder to integrate contextual information from the source sequence [Liu+23].

During training, the decoder receives the target sequence shifted one position to the right, with a start-of-sentence token added at the beginning. This shifting prevents the model from simply copying the decoder input to the output [Kha+22]. The masked multi-head self-attention approach prevents access to future tokens during training by setting the attention scores of all future positions to zero [Kha+22]. As a result, when predicting a particular token, the model considers only the current and previous tokens, ensuring that information from subsequent tokens is not incorporated.

At the end of the decoder, the output vectors are linearly projected and passed through a softmax function to obtain a probability distribution, enabling the model to predict the next word [Liu+23].

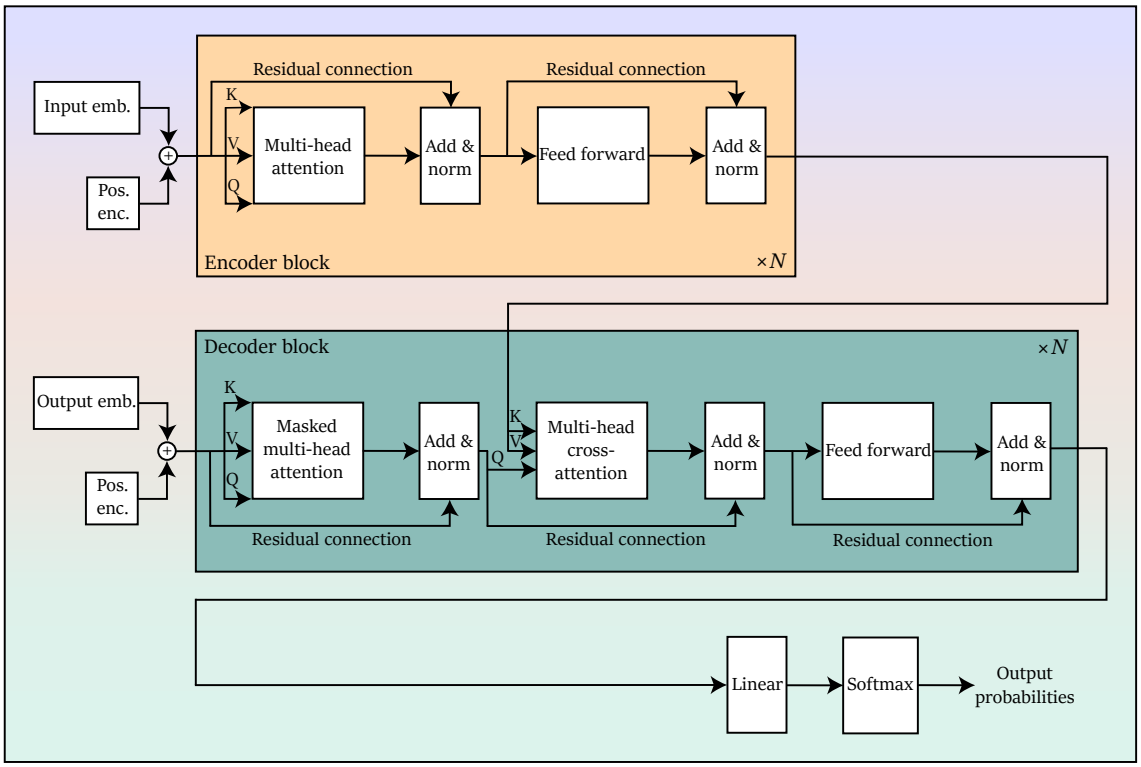


Figure 3.11.: Schematic illustration of the transformer architecture, based on [Vas+17]. The transformer follows an encoder-decoder design. For example, an input language sequence is first converted to embeddings, to which positional encodings are added; these are then fed into the encoder blocks. Both the encoder (top row) and decoder (bottom row) consist of N repeated blocks, each containing multi-head attention and feed-forward layers. Additionally, each block incorporates residual connections. For the decoder, the input sequence during training consists of the target (output) sequence, which is converted to embeddings with positional encodings. Decoder blocks additionally interface with the encoder's output via a cross-attention mechanism before passing through the feed-forward network. The output logits are transformed into a probability distribution over the target vocabulary through a final softmax operation. Abbreviations: Emb.: Embedding; Pos.: Positional; Enc: Encoding.

Vision Transformer

Given the potential of the self-attention module, Dosovitskiy et al. [Dos+20] proposed adapting the base Transfomer architecture - with some modifications - for image classification, resulting in Vision Transformers (ViTs). These models have achieved competitive performance on various image classification benchmarks [Dos+20; MDB23]. The framework of ViT is depicted in Fig. 3.12. Because the standard transformer model is designed for sequential inputs, ViT first divides an input image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ - where H and W denote the height and width C is the number of channels - into a sequence of N non-overlapping, fixed-size patches $\mathbf{x}_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$, where $N = \frac{HW}{P^2}$ and P is the patch size [Dos+20]. Each patch vector is treated as a (visual) token.

These patches are linearly projected into patch embeddings. This approach allows the ViT to learn the long-range dependencies between different patches. To retain spatial

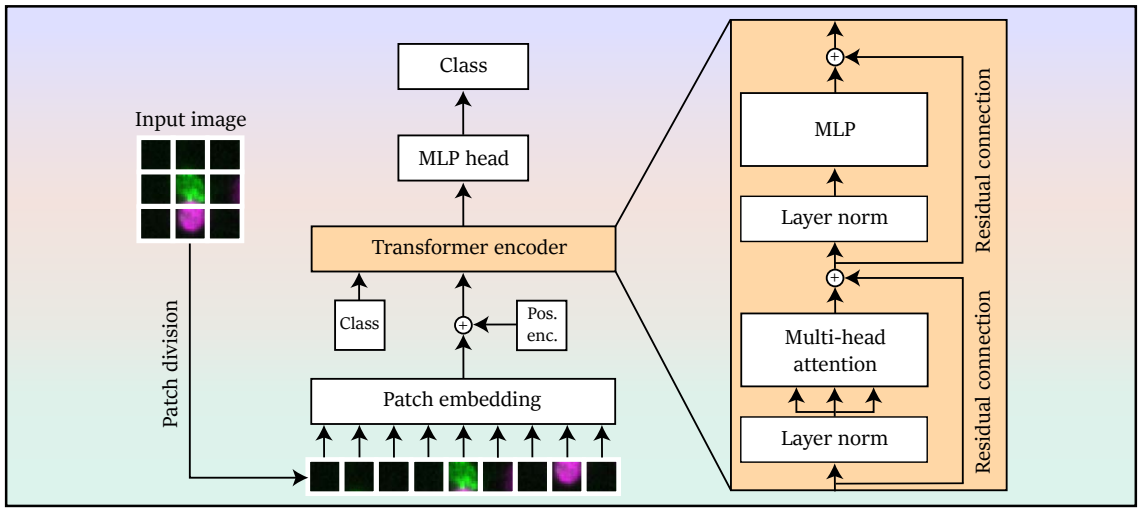


Figure 3.12.: Schematic illustration of the ViT framework, based on [Dos+20]. In ViT, the 2D input is reshaped into a sequence of flattened 2D patches. Both an additional classification token as well as the position embedding are added to the sequence. The classification token is used for classification. Abbreviations: Pos.: Positional; Enc: Encoding.

information, a 1D learnable positional encoding is added to each patch embedding, and an additional learnable class token is attached [Liu+23]. All together, these are fed into the encoder block of the transformer (see Fig. 3.12). The ViT encoder consists of multiple stacked layers, each comprising a multi-head self-attention mechanism, a feed-forward neural network, residual connections, and layer normalization - similar to the encoder from the base transformer architecture, except for the place for the layer normalization [Han+22]. These unified blocks are repeated several times to enable the model to capture complex representations of the input sequence [Kha+23]. After encoding, the output is passed to the MLP head to predict the input image class.

Building on the ViT paradigm, several ViT variants have been introduced to enhance performance on vision tasks [Was+22]. The main strategies involve strengthening locality, optimizing the self-attention mechanism, and improving the architectural design. In ViT, the self-attention mechanism enables global interaction between image patches; however, its quadratic time and memory complexity pose significant challenges for high-resolution images. This limitation has inspired many researchers to seek more efficient solutions. One notable example is the Cross-Covariance image Transformer (XCiT) [El-+21].

Cross-Covariance image Transformer: XCiT addresses computational complexity by replacing the conventional self-attention mechanism with a transposed attention variant, termed Cross-Covariance Attention (XCA). In contrast to the standard approach, where attention is calculated across tokens (i.e., each token attends to every other token), XCA computes attention along the feature dimensions for each position along the input

sequence [El-+21]. By transposing Q, K, and V, and inverting the order of the dot product in equation (3.31), XCA reduces the computational complexity with respect to the number of tokens from quadratic to linear [Ahn+23]. The resulting transposed feature dimension self-attention can be expressed as follows [El-+21; Ahn+23]:

$$\mathrm{XCA}(Q, K, V) = \mathrm{Softmax}\left(\frac{\widehat{K}^\top \widehat{Q}}{\tau}\right) V, \tag{3.32}$$

where τ is a learnable temperature parameter and $\widehat{Q}$ and $\widehat{K}$ are the L2-normalized Q and K vectors (denoted by the hat), meaning each vector is rescaled to have a Euclidean (L2) norm of one (i.e., the square root of the sum of squared vector elements equals one). L2-normalization has been observed to stabilize the training process [El-+21]. The scaling parameter τ adjusts the sharpness (i.e., how concentrated or spread out the attention weights are) of the attention weight distribution by dividing the normalized dot products by τ before applying softmax [El-+21].

Following the XCA layer, the architecture incorporates a Local Patch Interaction (LPI) layer and a feed-forward network. While the XCA layer facilitates information mixing across feature channels rather than sequence tokens, the LPI layer applies a 2D convolution to enable interactions among neighboring tokens, thereby enhancing local spatial information exchange [El-+21].

3.4.2. Self-supervised learning with DINO

Since the ViT architecture and its variants usually require large-scale data sets to obtain good performance, a two-stage training strategy is often adopted. First, the model undergoes supervised or self-supervised pre-training on a large data set or even a collection of data sets; then, the pre-trained model is fine-tuned on a smaller data set tailored for specific tasks such as classification [Ali+23]. Self-supervised learning (SSL) differs from unsupervised learning in that it aims to learn discriminative and generalizable feature representations from unlabeled data without the need for human-annotated labels [Gui+24; Ran+23b]. This enables downstream tasks such as image classification to be performed more efficiently and with reduced reliance on ground truth labels [JT20].

Various approaches to SSL exist in the literature, and many of them are based on contrastive learning. Contrastive learning is an approach in which a model learns to distinguish between similar and dissimilar data samples. The idea is to bring representations of similar (positive) pairs closer together in the feature space while pushing representations of dissimilar (negative) pairs farther apart [Ete+25]. Positive pairs generally consist of different augmented versions of the same sample, whereas negative pairs are

composed of views from distinct samples. Methods such as SimCLR (Simple Framework for Contrastive Learning of Visual Representations) [Che+20] and MoCo (Momentum Contrast) [He+20] have further enhanced performance by introducing improvements in data augmentation strategies, loss functions, and efficient sampling of negative samples. An important development in contrastive learning is the introduction of methods such as BYOL (Bootstrap Your Own Latent) [Gri+20], which achieve strong performance without relying on explicit negative samples [Ete+25].

Inspired by approaches like momentum encoder and BYOL, a more recent approach called self-distillation with no labels (DINO) was introduced by Caron et al. [Car+21] to learn visual representations without labels. As the name suggests, DINO is based on the principle of knowledge distillation. In traditional distillation, knowledge is distilled (transferred) from a larger teacher model to a smaller student model to improve the performance of the student model. However, in DINO, both the teacher and student share the same architecture, but have distinct parameters, denoted as θ_t for the teacher and θ_s for the student. Both networks consist of a backbone, utilizing either ViTs or CNNs such as ResNet50, along with a projection head (a 3-layer MLP) placed on top. The networks are trained on image patches of different sizes, as illustrated in Fig. 3.13. The DINO approach generates multiple image crops from two perspectives: global views

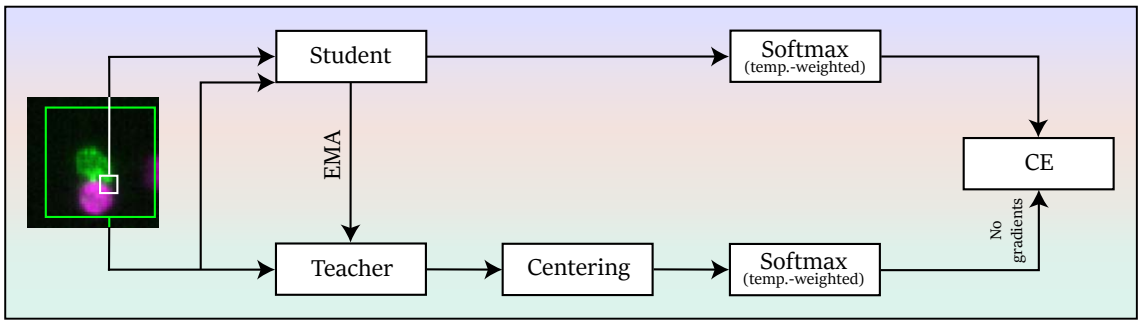


Figure 3.13.: Schematic overview of the DINO framework, based on [Car+21]. Two sets of augmented crops are generated using a multi-crop augmentation strategy: the teacher set includes two global views, while the student set contains these global views and five additional local crops. For simplicity, only one global and one local crop are illustrated. Both student and teacher networks share the same architecture (a backbone and an MLP head) but maintain separate weights. After forwarding the crops through their respective networks, only the teacher outputs are centered, while the student outputs remain uncentered. Next, a temperature-scaled softmax is applied, and the categorical CE loss is computed between the resulting output distributions. The student network is updated by backpropagation, whereas gradients are not computed for the teacher. Instead, the teacher's weights are updated as an EMA of the student's weights. Abbreviations: EMA: Exponential moving average; CE: Cross-entropy; Temp.: Temperature.

and local views. In its standard multi-cropping setup, each input image is processed to produce two large patches (global views) that cover more than 50% of the image area, as well as five smaller patches (local views) that cover less than 50% of the image area. All

these patches are then subjected to various augmentations. The teacher model processes only the global crops, while the student model receives both the global and local crops. The objective in DINO is to learn a consistent representation across different patches of the same input image by minimizing a temperature-scaled categorical CE loss between the output distributions of the student and teacher networks (the softmax is sharpened by temperature weighting). The student model parameters θ_s are updated using SGD, whereas the teacher model parameters θ_t are not given a priori and are dynamically built during training. In order to do so, the teacher network is frozen (i.e., it does not receive gradient updates through backpropagation) over an epoch and its parameters θ_t are updated as an exponential moving average (EMA), i.e., a momentum encoder, of the student's parameters θ_s:

$$\theta_t \leftarrow \lambda\theta_t + (1 - \lambda)\theta_s, \tag{3.33}$$

where λ is a momentum coefficient that follows a cosine schedule from 0.996 to 1 during training [Car+21]. To further stabilize training, centering is applied to the teacher outputs; that is, a running mean (center) is subtracted from the teacher's output.

DINO demonstrates particularly strong performance when used with ViT architectures. Evaluations of features pre-trained with DINO on various downstream tasks show that self-supervised pre-training enhances results on ImageNet, leading to improvements of 1-2% in accuracy [Car+21].

Chapter 4

Experiments and results

The chapters 2 (Biological principles) and 3 (Principles of artificial intelligence and image processing) introduced the fundamental concepts and methods that formed the basis of the analyses presented here. In total, 173 in-house CS cartridges were collected during the course of this thesis: 163 from patients with metastatic breast cancer (collected retrospectively) and 10 from healthy donors. Below is an overview of the sections presented in this chapter.

Section 4.1: StarDist segmentation of cells in liquid biopsy data
In this section, a state-of-the-art DL method is introduced, applied, and evaluated on fluorescence microscopy images acquired by the CS system for cell segmentation. Additionally, an approach for establishing correspondence between the DL-based cell segmentations and CS-derived cell detections is presented.

Section 4.2: Label efficient classification of circulating tumor cells by self-supervision
Bildverarbeitung für die Medizin, 2023

The limited availability of high-quality annotated image data sets poses a major challenge to ML-based image classification tasks, which SSL can help address by enabling robust representation learning without human annotation. In this section, after segmenting the cells, such an approach for label efficient, binary CTC classification in a metastatic breast cancer cohort (12 patients) is presented and evaluated against state-of-the-art supervised methods.

Section 4.3: Self-supervised learning and targeted human-in-the-loop strategy for improving circulating tumor cell classification
Patterns, 2025

The aforementioned setup is evaluated on an extended cohort (90 patients). To further improve CTC classification, especially in cases where the model is uncertain in its predictions, a cluster-based, targeted human-in-the-loop approach is proposed. This strategy is assessed on a separate hold-out test set and compared with the CS system regarding both the number

of suggested candidates/events and the actual CTCs. The corresponding source code is publicly available on GitHub [Hus].

Section 4.4: Analyzing CTC detection concordance and sensitivity: Human-in-the-loop model and CellSearch® in patients and healthy donors
This section presents an extended comparative analysis of CTC detection by the human-in-the-loop model from from the previous publication (*Patterns*) and the CS system in a more challenging cohort, including metastatic patients with low CTC counts as well as healthy donors. The concordance and sensitivity of both systems are systematically investigated, with particular focus on discrepant findings, ambiguous cells, and the impact on the number of detected CTCs when applying a model confidence threshold.

The experiments and analyses presented in this thesis were conducted using hardware resources provided by the University Medical Center Hamburg-Eppendorf (UKE). The specifications are listed below.

Laptop:	CPU: Intel Core i9-11900H (11th Gen) @ 2.50GHz (8 cores) RAM: 32 GB (2×16 GB) GPU(s): NVIDIA GeForce RTX 3080 Laptop GPU (4 GB), Intel UHD Graphics (1 GB)
Computer:	CPU: 2× Intel Xeon E5-2620 v4 @ 2.10GHz (16 cores) RAM: 62 GB GPU(s): 2× NVIDIA GeForce GTX 1080 Ti (11 GB each)
Server:	CPU: 2× AMD EPYC 7513 32-Core Processor (64 cores) RAM: 1 TB GPU(s): 5× NVIDIA A40 (46 GB each), 1× NVIDIA A100 (80 GB)

4.1. StarDist segmentation of cells in liquid biopsy data

Accurate and robust segmentation of cells in CS microscopy images is an essential prerequisite for the reliable detection of CTC candidates and forms the foundation for their subsequent CTC differentiation. U-Net, as introduced in Section 3.3, has established itself as a particularly effective and widely used architecture for (bio)medical image segmentation. Among the U-Net-based approaches is the StarDist method, which was used in this work.

4.1.1. StarDist

The StarDist method was introduced to localize cell nuclei in fluorescence microscopy images by representing their shapes with star-convex polygons [Sch+18]. In a star-convex shape, a point - referred to as the center - can be identified such that every boundary point of the shape is directly visible from this center. This implies that it is possible to connect the center to any point on the boundary with a straight line, allowing the shape to be reconstructed by the distances from the center to the boundary along various directions (Fig. 4.1A).

This property is also applicable to the channel signals observed in CS cartridge images, as these often exhibit a circular appearance (Fig. 4.1B). The authors of StarDist provide a pre-trained model, *2D_versatile_fluo*, intended for single-channel fluorescence microscopy images. This model was trained on approximately 600 2D images from the 2018 Data Science Bowl (DSB2018) data set, including about 20,000 annotated nuclei. Since the cell nuclei in these images show morphological similarities to the CK signals present in CK cartridge images, no new StarDist model was trained or further fine-tuned as part of this thesis. Instead the pre-trained model provided by the authors was directly applied to these images for segmentation.

Before describing the segmentation pipeline applied in this thesis (see Section 4.1.2), an overview of the StarDist method - inlcuding training data generation, network training, and model inference - is provided below, based entirely on the original work of the authors [Sch+18]. To support this overview, Figure 4.1 presents the training input and procedure using an example image from the DSB2018 data set, while Figure 4.2 depicts the inference and post-processing steps using a representative CK cartridge image.

Training data: The authors defined as input to the StarDist method pairs of raw microscopy images and annotated instance segmentation masks. In each mask, individual objects are assigned unique integer labels, while background pixels are set to zero.

The raw input images are normalized using a percentile-based approach. For each image, intensity values corresponding to a chosen lower percentile p_{min} are mapped to 0, while values at an upper percentile p_{max} are mapped to 1. Pixel intensities between these percentiles are linearly mapped to values in the interval $[0, 1]$, while intensities below p_{min} or above p_{max} are mapped outside this range.

From each GT segmenation mask, two types of GT targets are derived for network training: (1) an object probability map, and (2) a set of distance maps (Fig. 4.1C).

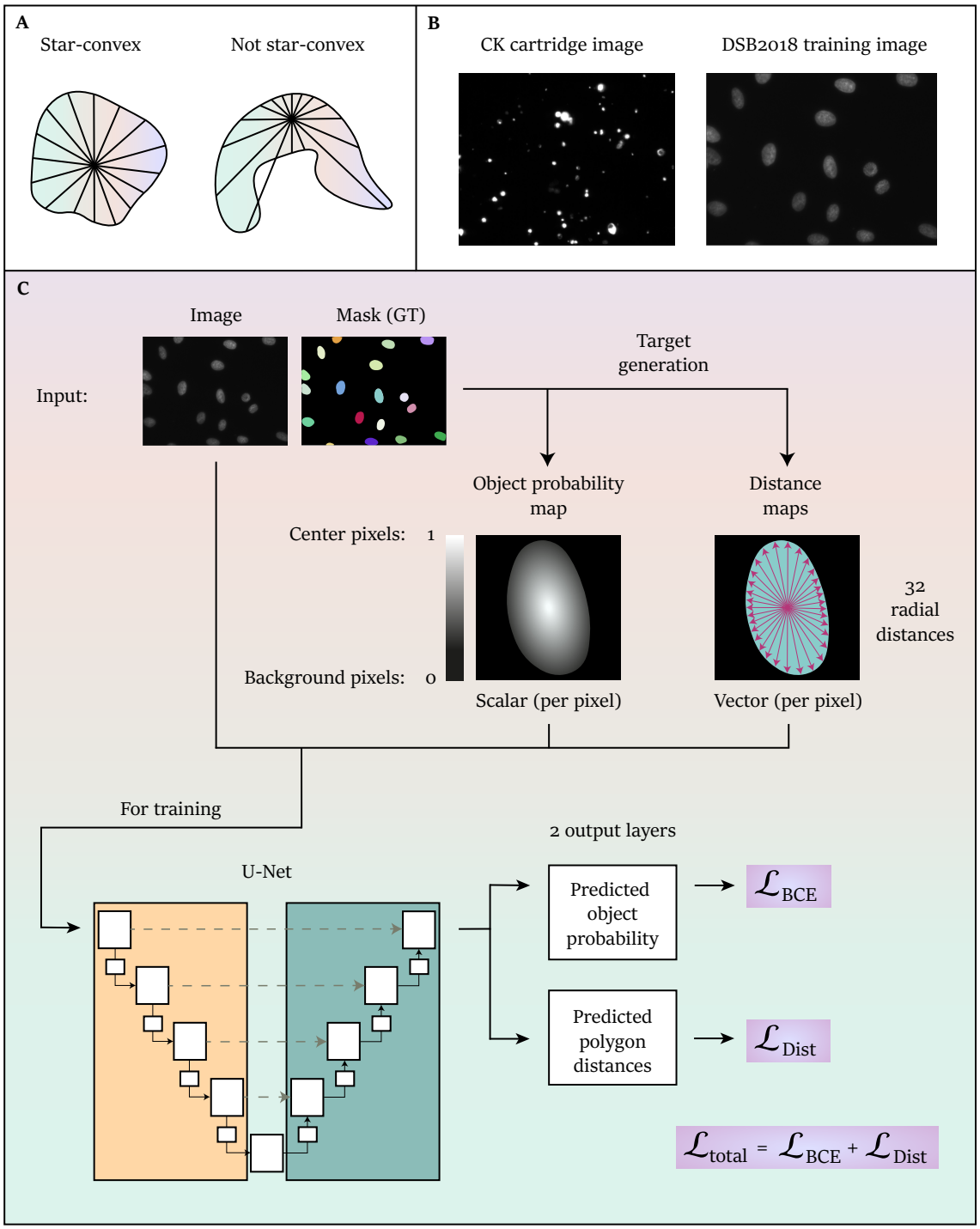


Figure 4.1.: Overview of the StarDist input and training method. (A) Example of a star-convex and a non-star-convex shape. (B) Example of a CK cartridge image and a representative training image from the DSB2018 data set. (C) Schematic of the StarDist training workflow. Each input consists of a normalized microscopy image and the corresponding segmentation mask. From each mask, an object probability map and several distance maps (one for each radial direction) are generated. For illustration, the radial distances for a single pixel are shown. During training, BCE loss is used for the object probabilities, and a distance loss is applied to the polygon distances; both losses are weighted and summed, but for simplicity, weighting is not depicted here. Abbreviations: DSB2018: Data Science Bowl 2018; GT: Ground truth; BCE: Binary cross-entropy; Dist: Distance.

(1) Object probability: rather than simply assigning each pixel as belonging to either an object or to the background, the object probability map encodes how likely a pixel is to be located at the center of an object. This map is created by applying the Euclidean Distance Transform (EDT) individually to each labeled object region in the mask image. For each object, a temporary binary mask is generated corresponding to that object only, and the EDT computes, for every pixel within the object, the shortest distance to the nearest background pixel (i.e., any zero-valued element) [Sch+18]. These distance values are then normalized by dividing by the maximum distance within that object, so that pixels at the object center have values close to one, whereas those near the object boundary approach zero. Pixels outside of any object remain at zero, ensuring that background regions are unchanged.

(2) Polygon distances: in contrast, the distance maps provide geometric information describing the shape of each object. Specifically, for every pixel within a labeled object, a fixed number of radial directions (so-called rays) are defined, evenly spaced in angle around the pixel. In the original StarDist approach, 32 rays are used [Sch+18]. Along each ray, the distance from the pixel to the boundary of the object is measured, resulting in a vector of 32 Euclidean distance values for each pixel. This distance vector describes the local shape of the object at each pixel. By leveraging these distance maps, the network can learn not only whether a pixel belongs to an object but also the spatial extent and boundaries of each individual instance.

Network training: The StarDist method employs a slightly modified U-Net architecture. After the final feature layer of the U-Net, an additional 3×3 convolutional layer with 128 channels is inserted [Sch+18]. This is followed by two output layers: (1) a single-channel layer with sigmoid activation that predicts the object probability for each pixel; (2) the second is a multi-channel output layer - with as many channels as there are rays - using a linear activation function to predict the distances from each pixel to the cell boundary along each ray [Sch+18].

(1) Object probability: during network training, a binary cross-entropy (BCE) loss is utilized for the predicted object probabilities. In its classical form (Eq. 3.22), BCE is defined between the predicted probability $\hat{y}_i \in [0, 1]$ and a binary GT label $y_i \in \{0, 1\}$. However, in StarDist, the GT is a continuous value in the interval $[0, 1]$, reflecting the degree to which each pixel is located at the center of an object.

(2) Polygon distances: for the polygon distances, a masked version of the mean absolute error (MAE; see Eq. 3.16 for the standard definition) is applied as described in [Sch+18]:

$$\mathcal{L}_{\text{Dist}}(y, \hat{y}, m) = \frac{\sum_{i=1}^{N} m_i \cdot \left(\frac{1}{n} \sum_{k=1}^{n} |y_{i,k} - \hat{y}_{i,k}|\right)}{\sum_{i=1}^{N} m_i + \varepsilon}, \tag{4.1}$$

where N is the total number of pixels, n is the number of predefined rays per pixel, $y_{i,k}$ and $\hat{y}_{i,k}$ are the GT and predicted Euclidean distances for the i-th pixel and k-th ray, $m_i \in [0, 1]$ is the object probability for pixel i, as derived from the normalized EDT, and ε is a small positive constant to avoid division by zero when no object pixels are present. In this formulation, the MAE between predicted and GT distances is first calculated across all rays for each pixel. This per-pixel error is then weighted by the object probability m_i, so that only pixels belonging to objects contribute to the loss, and pixels closer to the object center (with higher m_i) have greater influence. The sum of these weighted errors is then normalized by the total sum of mask weights, ensuring that the loss is invariant to the number of object pixels in a batch.

The objective of network training is to jointly optimize both outputs by minimizing a combined loss function. This combined loss is defined as a weighted sum of the BCE loss for the object probability map and the masked MAE loss for the distance maps:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{prob}} \cdot \mathcal{L}_{\text{BCE}} + \lambda_{\text{dist}} \cdot \mathcal{L}_{\text{Dist}}, \tag{4.2}$$

where λ_{prob} and λ_{dist} are weighting coefficients that control the relative contribution of the object probability and distance loss, respectively. For the original StarDist 2D model for microscopy images, values of $\lambda_{\text{prob}} = 1$ and $\lambda_{\text{dist}} = 0.2$ were used. This configuration places greater emphasis on the BCE loss for the object probability map, while still incorporating the distance loss to guide the polygon prediction.

Inference and post-processing: For model inference in StarDist, a normalized microscopy image is used as an input. When this image is passed through the pre-trained StarDist model, the network predicts per-pixel object probabilities and corresponding star-convex polygon distances. Then, StarDist applies a post-processing pipeline to generate the final set of segmented objects, as shown in Figure 4.2.

First, a user-defined probability threshold is applied to the object probability map to identify candidate pixels likely to serve as object centers; only pixels with a probability above this threshold are considered for further processing. For each candidate pixel, a

polygon is reconstructed using its predicted distances along the predefined rays, resulting in one candidate object (polygon) per center.

Because neighboring pixels within the same object often exceed the threshold, this step may yield multiple, highly overlapping candidate polygons per true object. To eliminate redundant polygons, StarDist employs non-maximum suppression (NMS). In this process, all candidate polygons are first sorted in descending order based on their object probability score (i.e., predicted object probability at their center pixel).

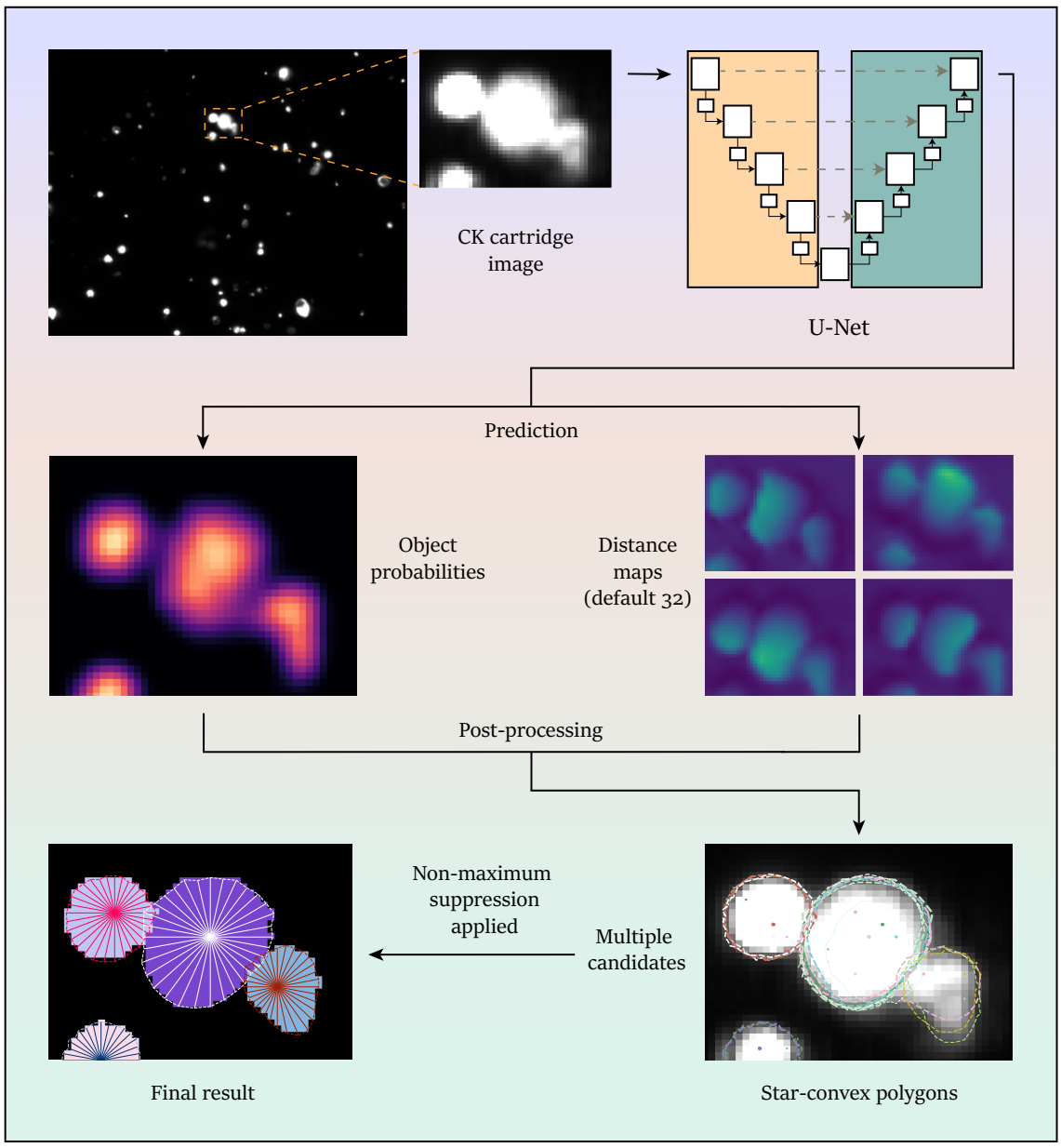


Figure 4.2.: StarDist model inference and post-processing. Given an input image (for example, a crop of a CK cartridge image), StarDist uses a U-Net backbone to predict both object probabilities and radial distances (by default, 32 distance maps). These predictions result in multiple star-convex polygons, which are subsequently processed using NMS to obtain the final set of object candidates. Abbreviations: CK: Cytokeratin.

The polygon with the highest score is selected first and added to the list of final objects. Next, the overlap between this selected polygon and each of the remaining candidates is evaluated using the Intersection over Union (IoU) measure, which is defined as follows:

$$\mathrm{IoU}(A, B) = \frac{|A \cap B|}{|A \cup B|}, \tag{4.3}$$

where A and B denote the two polygons under consideration, $|A \cap B|$ is the area of their intersection, and $|A \cup B|$ is the area of their union.

If the IoU between the selected polygon and any candidate exceeds a user-defined NMS threshold, that candidate is suppressed and removed from the pool of possible objects. In contrast, candidates whose IoU with the selected object is less than or equal to the NMS threshold remain in the candidate pool and may be selected in subsequent iterations. In each new iteration, the candidate with the highest remaining object probability is chosen from this pool, added to the final objects, and the suppression step is repeated against all other remaining candidates. This procedure continues iteratively until no candidates remain.

Generally, a higher NMS threshold allows more overlap between segmented objects, which may be useful in cases of touching objects. In contrast, a lower threshold enforces polygon suppression, reducing the likelihood of overlapping segmentations.

The proposed StarDist method was tested by the authors on two synthetic data sets as well as on the DSB2018 set [Cai+19] against baseline U-Nets and Mask R-CNN (state-of-the-art instance segmentation network) for several IoU thresholds. In particular, on the DSB2018 data set, which also contains densely packed nuclei, StarDist achieved superior segmentation performance compared to other methods, especially at moderate IoU thresholds (less than 0.75).

4.1.2. Segmentation pipeline

Following these overviews of the original StarDist work, the segmentation pipeline applied in this thesis will be presented. The pre-trained *2D_versatile_fluo* model is provided through the StarDist ImageJ/Fiji plugin, with default parameter values for NMS post-processing (probability threshold: 0.5, overlap threshold: 0.4), as well as through the StarDist Python module (probability threshold: 0.48, overlap threshold: 0.3). By default, the percentile-based normalization of the raw input images uses the following settings: ImageJ/Fiji: percentile low: 1.0, percentile high: 99.8; Python module: percentile low: 3.0, percentile high: 99.8. Throughout this thesis, both ImageJ and the Python library were used, and the parameters were chosen within the recommended ranges.

As the model is designed for single-channel fluorescence microscopy images, the four-channel cartridge images (DAPI, additional filter, CK, and CD45) were first split by channel. Only the DAPI, CK, and CD45 channels are relevant for CTC identification; therefore, the additional filter was excluded from further analysis. StarDist segmentation was performed exclusively on the CK channel for several reasons. First, CK serves as a reliable marker for tumor cells. Second, DAPI images tend to contain more crowded signal regions compared to CK images, which increases the likelihood that some regions are not segmented at all (Fig. 4.3) or, if segmented, are delineated imprecisely (Fig. 4.4).

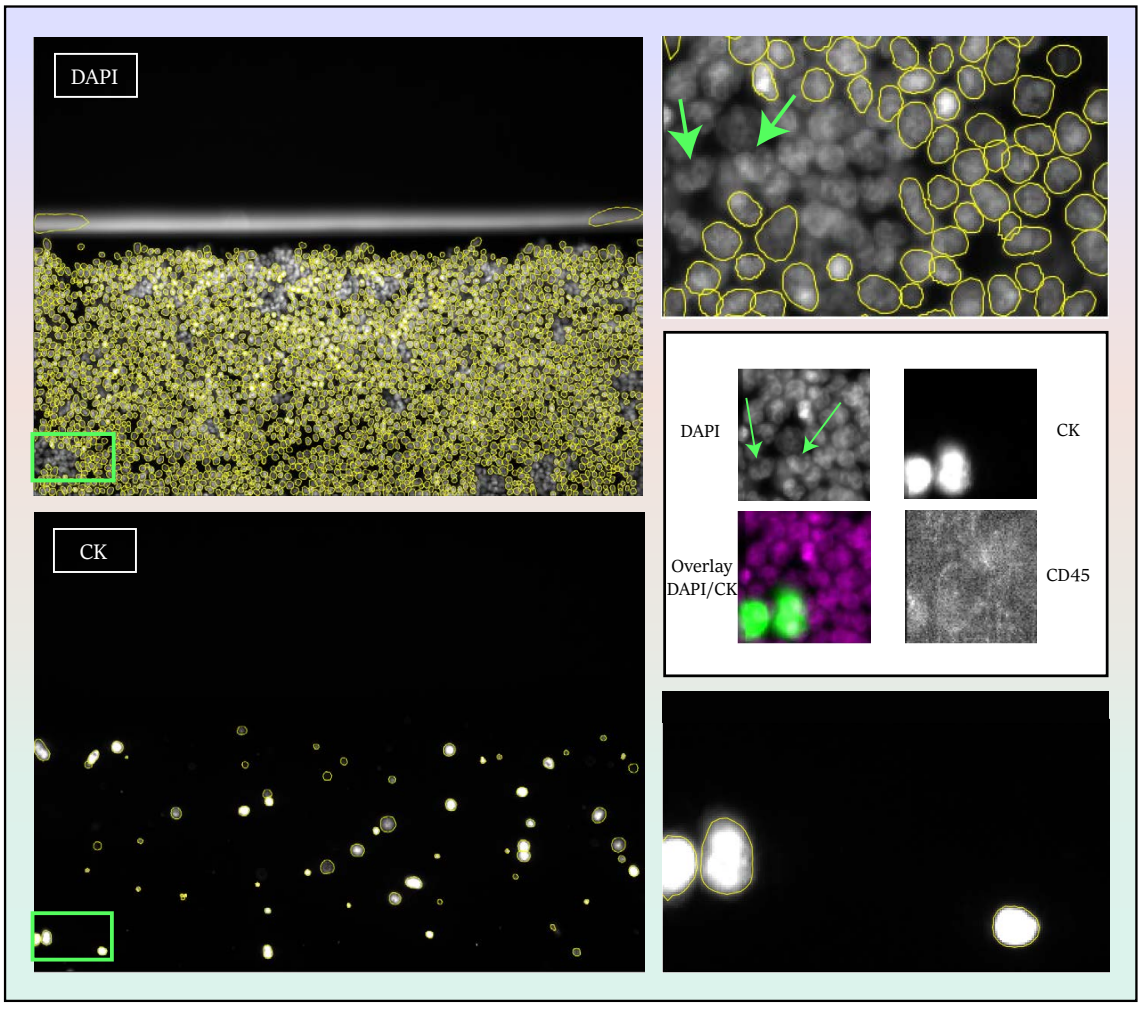


Figure 4.3.: Example segmentations generated by the pre-trained StarDist model for both DAPI and CK cartridge images. The shown DAPI channel image contains densely packed nuclear signals; the corresponding CK channel image has fewer signals. Segmented regions are outlined with yellow contours. Although most nuclear signals in the DAPI channel are segmented, some regions are missed; one such area is highlighted in the right crop. In the same region of the CK channel, the signals - corresponding to CTCs - are correctly segmented. This example illustrates that, in certain cases, CTCs would be missed if segmentation relied solely on the DAPI channel, whereas segmentation based on the CK channel could potentially capture these CTCs. Abbreviations and explanations: DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45-APC: Exclusion marker.

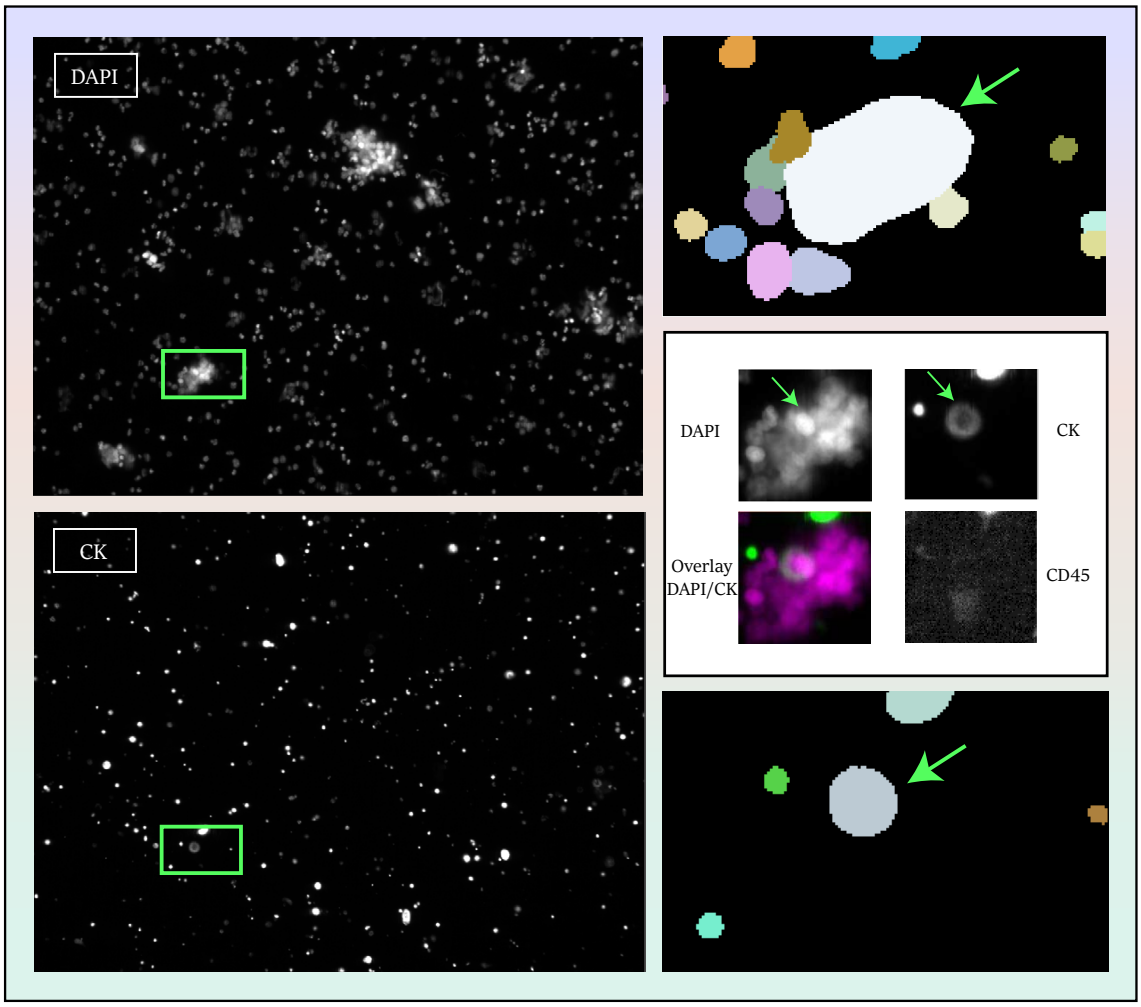


Figure 4.4.: Exemplary segmentation results show that DAPI images may contain crowded nuclear regions, leading to less precise segmentation masks (e.g., large blobs), whereas segmentation on the CK channel produces more accurate cell boundaries. The detected cell is a CTC. Abbreviations and explanations: DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45-APC: Exclusion marker.

In such crowded regions, CTCs may be present but are more difficult to detect and accurately segment in the DAPI channel compared to the CK channel, as shown in Fig. 4.3 and Fig. 4.4. Third, although low signal intensities can occur in both the DAPI and CK channels, relying solely on DAPI-based segmentation introduces a particular risk of missing CTCs that have weak DAPI signals but clear CK expression, as exemplified in Fig. 4.5.

When predicting on the data, StarDist produces instance labels and additional details such as the 32 coordinates (see yellow contours in Fig. 4.3) describing each segmented object in the current cartridge frame. Based on these coordinates, for each detected object, the smallest possible bounding box is first generated. This bounding box is then resized to obtain a square crop of 48×48 px, centered on the original bounding box center.

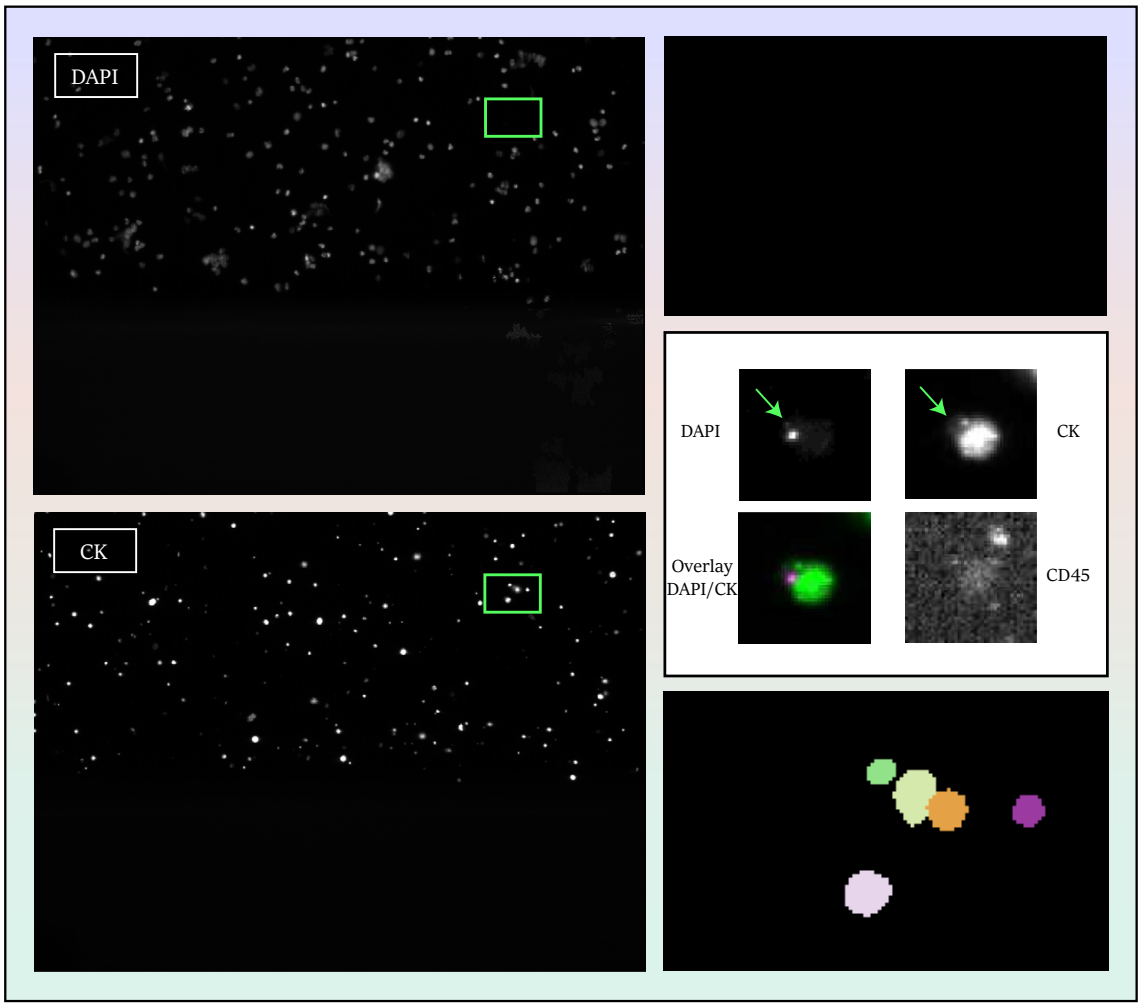


Figure 4.5.: An example segmentation result is presented where a cell - specifically, a CTC - is missed in the DAPI channel (no segmentation mask appears in the zoom-in), but is successfully captured on the CK channel. Increasing the contrast in the DAPI crop reveals that a weak DAPI signal is actually present in this region. Abbreviations and explanations: DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45-APC: Exclusion marker.

While StarDist generally performs robustly, it can occasionally produce jagged segmentation masks or divide regions into too many segments (oversegmentation), as shown in Fig. 4.6. Such segmentation masks may result in images where the CK signal is not positioned at the center of the crop. However, similar off-center signals are also observed in CS galleries and are therefore not uncommon for experts interpreting images for CTC selection.

After resizing and before cropping, it is further verified that this square bounding box lies entirely within the image boundaries (the current cartridge); if it extends beyond the edge of the image, the bounding box coordinates are adjusted accordingly to ensure that the resulting crop is valid and remains as close as possible to the intended size and centered position. The coordinates of the resized (and, if necessary, adjusted) bounding

box are also applied to the other two channel images (DAPI and CD45), so that for each detection, square-cropped images at the exact same position are obtained from all three channels. These channel image crops are saved as PNG files, together with a corresponding dataframe containing the bounding box coordinates and image file paths. For subsequent tasks, these channel images are later stacked together in the order DAPI, CK, and CD45.

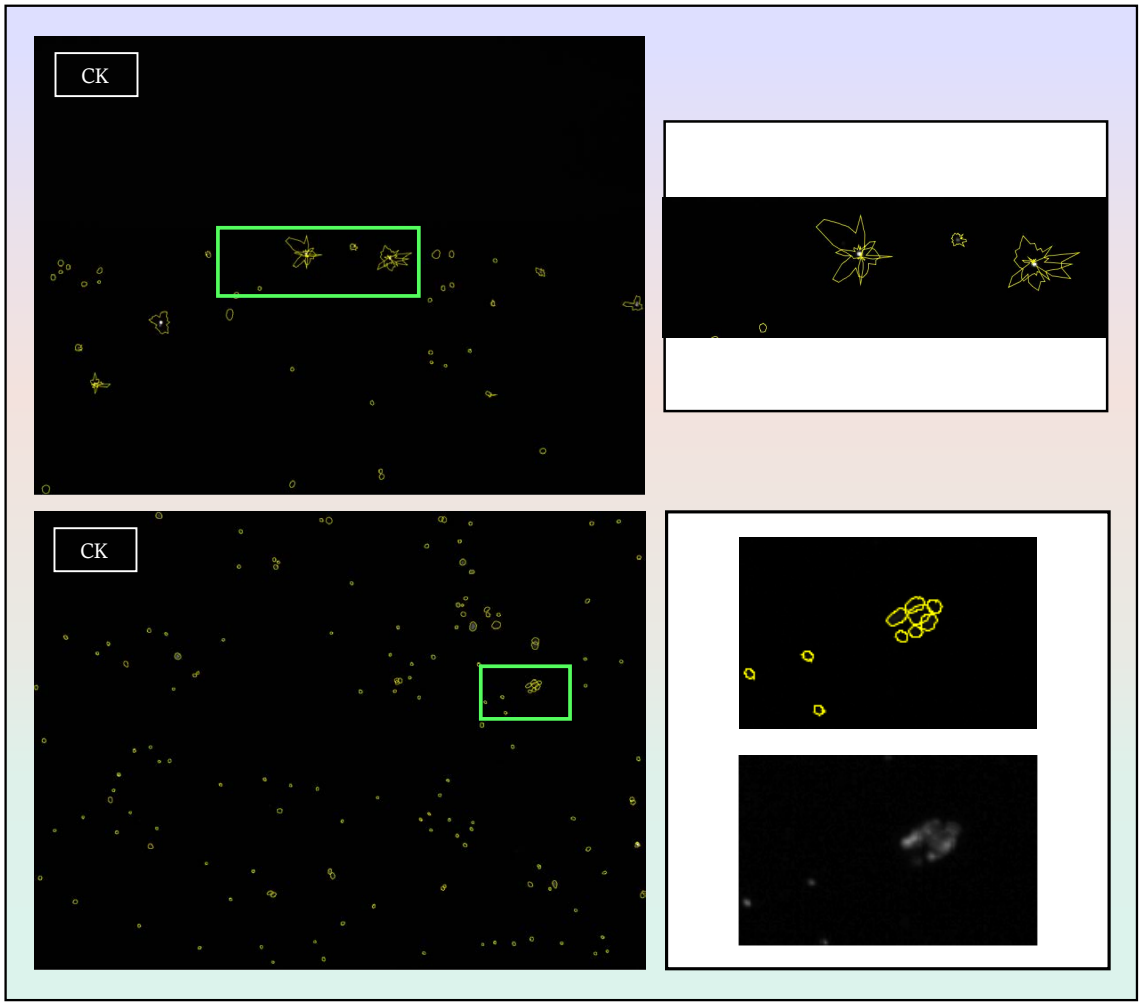


Figure 4.6.: Exemplary CK channel segmentations illustrating jagged masks in the top image and oversegmented signals in the lower image. For improved visualization, contrast and brightness were enhanced in the oversegmented area to better highlight this region. Abbreviations and explanations: DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45-APC: Exclusion marker.

4.1.3. Correspondence finding between StarDist segmentation and CellSearch® detection

Since the CS system is the gold standard in CTC enumeration for breast cancer, the detection and subsequent classification performance of the methods proposed in this

thesis are compared against the CS system, particularly with respect to the number of identified CTCs. Although the CS segmentation algorithm itself is not publicly available, the bounding box coordinates of the events in the CS image gallery, as well as the cartridge frame number they belong to can be accessed through an Extensible Markup Language (XML) file provided alongside the cartridge images. Additionally, this XML file contains information indicating which image gallery events were selected by the operator as CTCs and which were not. In Figure 4.7, the schematic scanning procedure from Figure 2.2 is extended to include the dimensions of the entire microscope slide, which measures 48,440 × 5,180 px, as the coordinates obtained from the XML file refer to the global coordinates.

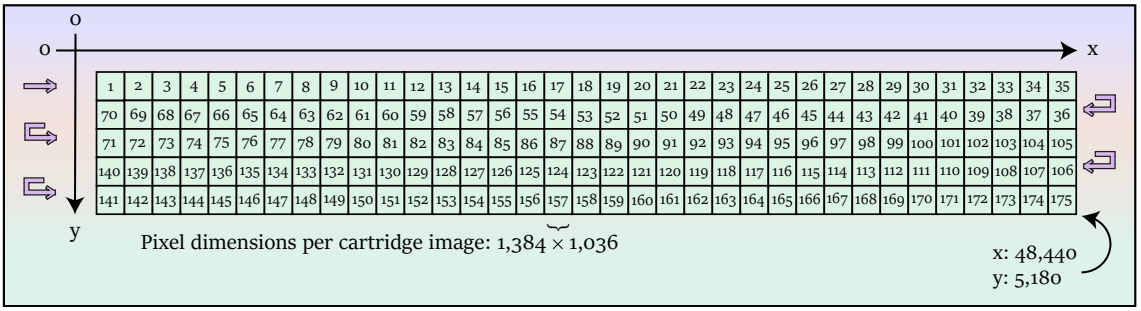


Figure 4.7.: Schematic depiction of the entire cartridge as subdivided into individual image patches (cartridge frames). The scanning process is illustrated by pink arrows, indicating the zig-zag pattern in which the frames are sequentially acquired. Each cartridge frame has pixel dimensions of 1,384×1,036 px, resulting in a total composite image size of 48,440×5,180 px. The coordinates provided in the XML file, which correspond to image gallery events, reference positions within the entire cartridge image.

Since StarDist is applied to each cartridge frame separately, the resulting coordinates are referenced within the pixel dimensions of a single cartridge image. To match the StarDist coordinates with the CS ones, the global coordinates from the XML file must be mapped to the local coordinate system of the corresponding cartridge image.

For each detected object, the mapping is performed in these main steps: first, based on the corresponding frame number, it is determined to which row the frame belongs. Next, the global x-coordinates of the bounding box are adjusted by subtracting the total width of all cartridge frames located to the left of the target frame within the same row. If the cartridge frame belongs to the first column (i.e., for frame numbers 1, 70, 71, 140, or 141), the global x-coordinates are identical to the local coordinates. As for the global y-coordinates, these are adjusted by subtracting the total height of all cartridge frame rows above the target frame. If the cartridge frame belongs to the first row (i.e., frame numbers 1 to 35), the global y-coordinates are identical to the local coordinates.

After obtaining the local bounding box coordinates of the CS events (both selected and non-selected), these can be visualized on their respective cartridge frames for comparison with the objects detected by the StarDist segmentation pipeline. Examples of such

overlaps are illustrated in Fig. 4.8. Across all four CK cartridge images, StarDist identifies more segmented objects per image than the number of CS events, as the CS system applies a preselection step - displaying only those segmented objects where CK and DAPI signals are apparent within a square (bounding box).

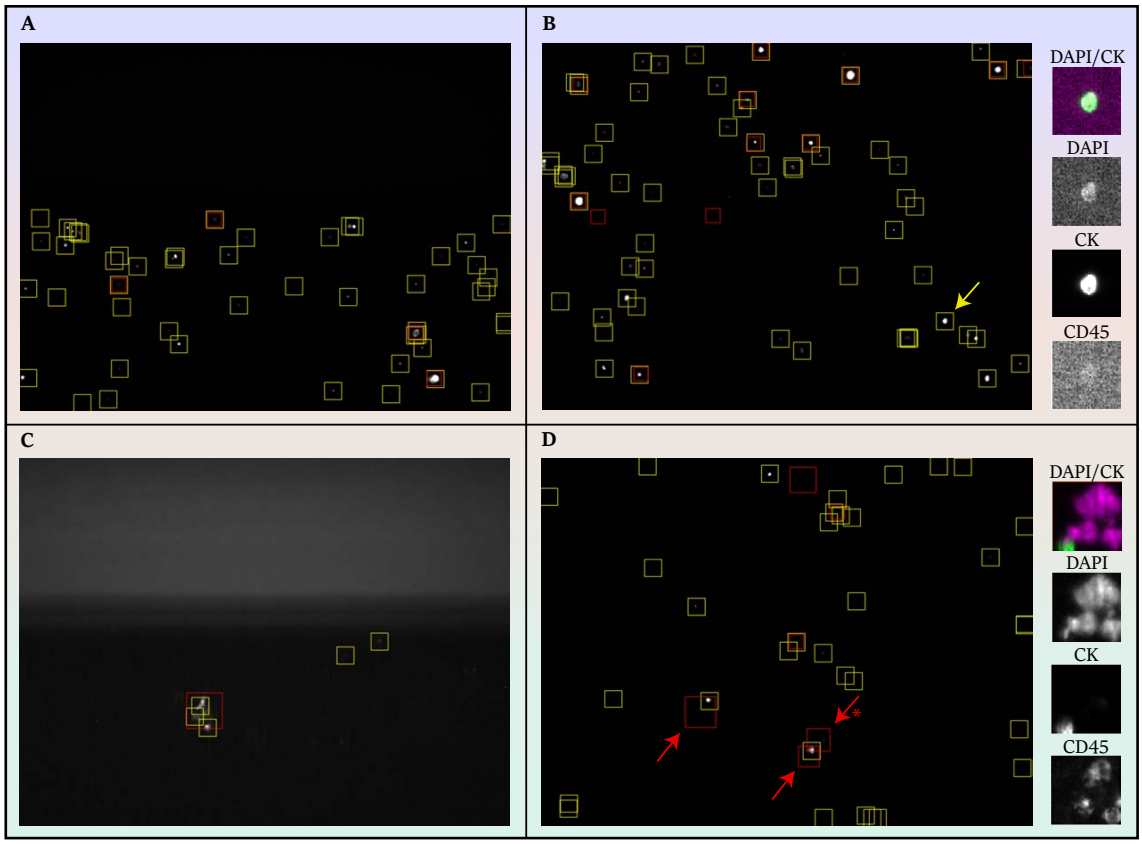


Figure 4.8.: Comparison of objects detected by StarDist (yellow boxes) and the CS system (red boxes, CS events), with bounding boxes visualized on CK cartridge frames. (A) StarDist bounding boxes overlap with all CS events. Furthermore, StarDist detects many other objects, with some bounding boxes showing multiple overlaps. (B) The CS system has identified some objects that StarDist did not detect, and vice versa. The yellow arrow indicates a CK-positive cell, which, upon closer inspection, is a CTC that was not listed as an event by CS in the gallery. (C) Three StarDist bounding boxes overlap with a single large CS bounding box. In one of the yellow boxes, the CK signal is positioned centrally, while in the corresponding red CS box, the signal is situated near the edge and is only partially captured. (D) Red arrows highlight large CS bounding boxes, each with a CK signal positioned at the edge rather than the center. The overlapping StarDist boxes center on the same CK signal. The red arrow marked with an asterisk indicates a CS event bounding box that was identified as a CTC by the operator. Abbreviations and explanations: DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45: Exclusion marker.

While one could manually inspect each cartridge frame to compare both systems in terms of CTC detection, this would be neither efficient nor practical, especially considering the presence of overlapping and redundant StarDist bounding boxes. Therefore, redundancy needs to be automatically reduced. To facilitate a systematic comparison,

the IoU between corresponding bounding boxes from both systems can be computed. However, the question remains as to which IoU threshold to consider. Here, a specific challenge arises from the difference in bounding box sizes: CS bounding boxes can vary considerably, while the StarDist segmentation pipeline generates fixed-size boxes (48×48 px). For example, in Fig. 4.8C, the CS event box measures 100×100 px. Even with perfect alignment, a StarDist box entirely within a CS box would yield a maximum IoU of only 0.23. When using the actual bounding box coordinates, the IoU is approximately 0.11, reflecting only partial overlap. Nevertheless, the biologically relevant region (the CK signal) is fully covered by the StarDist bounding box. Therefore, choosing a low IoU threshold (such as 0.1) could help ensure that relevant overlaps are not overlooked even when the overlap is only partial.

4.1.4. Discussion

StarDist, a DL-based approach, has been widely adopted for cell and nuclei segmentation tasks [Sch+18; WS22; Ste+22; Wol+23]. The method represents objects as star-convex polygons by predicting, at each pixel, the distances to the object boundary in a predefined set of directions. In the present study, it has been demonstrated that this shape representation is also applicable to signal shapes depicted in cartridge frames, particularly in the CK channel; however, in some cases, jagged segmentations may be produced, and certain areas with densely packed signals can still be challenging. Regarding the latter, Stevens et al. [Ste+22] conducted a comparison between the segmentation results produced by the StarDist model, the ACCEPT tool, and the CS system. They reported that both CS and ACCEPT tended to miss more cells in regions of high cell density than StarDist, or merge adjacent touching cells into a single segmented event. The authors argued that this limitation may be due to the fact that both CS and ACCEPT were developed for CTC samples, where the cell density is generally low enough that neighboring cells do not touch [Ste+22], whereas StarDist incorporates a NMS function.

While accurate and consistent segmentation is a prerequisite, the ultimate question remains whether a detected StarDist object (or cell) truly corresponds to a CTC. If an operator has already annotated all CTCs in the CS gallery, the IoU with corresponding StarDist detections can be calculated to evaluate agreement. However, in cases of metastatic disease where the image gallery is very large, it is not always guaranteed that every CTC has been marked manually. In addition, there may be CTCs detected by StarDist but missed by CS (see Fig. 4.8B), or vice versa. These considerations highlight the need for reliable classification of detected StarDist objects prior to comparison. Therefore,

in the following chapter, the automated classification of StarDist-segmented objects and cells for CTC identification will be addressed.

4.2. Label efficient binary classification of circulating tumor cells by self-supervision

The objects and cells segmented by StarDist are initially unlabeled, meaning that the segmentation output may contain not only cells but also artifacts or other objects, and it is not yet known which - if any - correspond to CTCs. Automated classification is therefore a crucial step for the identification and analysis of CTCs. While Zeune et al. [Zeu+20] have demonstrated that CNNs can achieve high accuracy in classifying CTCs in CS images (see Section 2.3.1), their supervised learning framework relied on a large GT data set for classification (binary classification: CTCs: 6,505; non-CTCs: 43,065). Annotating all - or even the majority - of StarDist-segmented objects for network training, however, is impractical. As current SSL approaches in computer vision [Car+21] have marked a shift in reliance from large annotated data sets to the usually much greater amount of available unlabeled data for model training - and allow downstream tasks to be performed with much smaller annotated data sets - the objectives of the experiments in section are as follows:

(1) To investigate the feasibility of applying an SSL approach, such as DINO [Car+21] (see Section 3.4.2), combined with a downstream ML classifier to LB data for binary CTC classification of StarDist-segmented objects, and to compare the general SSL performance against the performance of state-of-the-art DL models trained in a supervised manner.

(2) To compare the performance of both self-supervised and supervised approaches when only limited labeled training data is available.

(3) To analyze the latent space of the cell representations learned by both approaches.

The results are based on the following conference publication:

> [Hus+23] H. Husseini, M. Nielsen, K. Pantel, H. Wikman, S. Riethdorf, and R. Werner. "Label Efficient Classification in Liquid Biopsy Data by Self-supervision". In: *Bildverarbeitung für die Medizin 2023* (2023), p. 261

4.2.1. Data set

Cell segmentation and processing of cartridge images from 12 patients using the StarDist pipeline described in Section 4.1.2 resulted in a total of 69,297 single-cell images, each

with three channels (DAPI, CK, CD45). To establish a reliable GT composed of CTCs and non-CTCs, two experts collaboratively reviewed and annotated the cells according to CS criteria. By consensus, 1,101 cells were labeled as CTCs and 2,333 cells as non-CTCs across all patient samples. The remaining 65,863 cells in the data set were left unlabeled.

4.2.2. Methods

Model selection: For the supervised learning approach, two network architectures were used: EfficientNetB0 from the EfficientNet family as a representative of CNNs (see Section 3.4) and the XCiT model *xcit_nano_12_p8_224_dist* (see Section 3.4.1 for further details) as a representative of transformers. Both networks were pre-trained on ImageNet. For the SSL part, the DINO approach (see Section 3.4.2) was deployed with XCiT as the backbone.

Pre-processing: Prior to network training, the inputs were pre-processed. To match the input dimensions of the pre-trained models, all images were resized from 48×48 px to 224×224 px. Additionally, min-max normalization was performed by scaling the data values so that the minimum intensity was mapped to 0 and the maximum intensity to 1, resulting in all values lying within the interval. To ensure consistency with the normalization strategy described by Zeune et al. [Zeu+20] for the supervised approach, a percentile-based background correction was applied independently to each image channel. Specifically, for each channel, the 10th percentile of all pixel intensity values was calculated. This value - representing the intensity below which 10% of the pixel values lie - was subtracted from every pixel in the respective channel image. This approach suppresses background signal and minimizes the impact of background fluorescence or signal offset.

Data augmentation: Random rotations and horizontal as well as vertical flipping were applied to the images used for network training in a supervised manner. For the DINO approach, the multi-crop augmentation strategy was applied, and thus multiple augmentations were generated. The teacher network receives only the global crops, whereas the student receives both local and global crops. A local crop represents an area of the image smaller than 50%, and a global crop covers more than 50%.

Network training: For supervised training, the networks were trained for up to 600 epochs, with early stopping implemented to prevent overfitting on the training set. Training was performed using the Adam optimizer with a learning rate of 0.0001 and a weight decay of 0.001. Additionally, a learning rate scheduler was employed, in which

the learning rate was reduced by a factor of 0.5 if no improvement was observed for 25 epochs, with a cooldown period of 50 epochs. The minimum learning rate was set to 0.00001. BCE loss with logits (Eq. 3.23) was used. As for the SSL approach, parameters were chosen similar to Caron et al. [Car+21] and DINO was trained for a maximum of 300 epochs, but training was stopped once the loss had stabilized. The CE loss was calculated between all possible combinations of student and teacher outputs.

Evaluation The performance of the binary classification between CTCs and non-CTCs was assessed using the F1 score, calculated on a separate test data (4 patients, not used during training or validation) set comprising 410 samples (185 CTCs and 225 non-CTCs).

4.2.3. Experiments

In the following, the three main experiments will be described.

Assessment of general performance

The data from the 8 patients not used for testing was partitioned into training and validation sets. The labeled training set comprises 2,824 image samples (816 CTCs and 2,008 non-CTCs), while the validation set includes 200 samples, evenly split between CTCs and non-CTCs (100 each). For the supervised approach, the baseline models (EfficientNetB0, XCiT) were trained, and the classification was evaluated. Furthermore, four EfficientNets and four XCiTs were trained using a 4-fold cross-validation. For evaluation on the test data set, the models were ensembled by averaging the predictions of all eight models. This ensemble approach is referred to as Ensemble-DL. For DINO training, the data set consisted of all unlabeled objects - including ambiguous cells and artifacts - from patients outside the test set (n = 61,865). Validation was conducted using the same labeled validation set described before. After DINO training, the learned features were extracted and subsequently used to train a SVM (with a RBF kernel) for binary classification. The combination of DINO and the SVM will be denoted as DINO+SVM.

Labeled data efficiency

To assess whether CTC classification using a substantially reduced GT set - following SSL and feature extraction without labels - can achieve comparable classification performance to supervised learning when the same number of annotated samples is available, smaller and balanced data sets were constructed from the pool of labeled training samples. Specifically, data sets comprising 50 (CTCs: 25, non-CTCs: 25), 100, 200, 500, and 1,000 samples were created. Depending on the selected approach (supervised versus SSL),

multiple runs per limited data set were conducted to estimate the variability of the F1 scores. For each run, the samples were randomly drawn so that the data composition was not identical across runs, but the total number of training samples (and the number of samples per class) was kept constant. For the supervised XCiT, runs were repeated three times as end-to-end training is much more expensive in terms of computation and time, whereas the SVM is computationally inexpensive and thus was fitted 100 times in the DINO+SVM approach.

Latent space analysis

In addition to the two previously described experiments, an analysis of the latent space of the learned cell representations was carried out to investigate whether and how the cell classes form clusters. For this purpose, features extracted of the respective model (supervised XCiT, DINO) were used to fit a UMAP after undergoing PCA, on the whole labeled training data set, with the transformation subsequently applied to the labeled test data.

4.2.4. Results

The initial analysis, which focused on the general comparison of all approaches using the full labeled training set, revealed that the DINO+SVM approach (F1: 0.965) outperforms the supervised baseline models XCiT (F1: 0.943), EfficientNet (F1: 0.903), and the Ensemble-DL (F1: 0.944) with respect to the F1 score metric (see Fig. 4.9). When analyzing the latent space after feature extraction, both DINO and the supervised XCiT display distinct clustering of CTC and non-CTC cells, shown in orange and purple, respectively (see Fig. 4.9).

When evaluating the classification performance on smaller, balanced labeled data sets, DINO+SVM consistently achieved higher F1 scores than the supervised XCiT across all training set sizes examined. Most notably, with only 50 training samples, DINO+SVM already reached an F1 score of 0.865 ± 0.043, compared to 0.687 ± 0.025 for the supervised XCiT. This performance gap remains as the number of labeled samples increases: at 100 samples, DINO+SVM achieved 0.910 ± 0.033 (XCiT: 0.747 ± 0.021); at 200 samples, 0.925 ± 0.023 (XCiT: 0.837 ± 0.011); and at 500 samples, 0.959 ± 0.004 (XCiT: 0.854 ± 0.012), at which point DINO+SVM surpassed the Ensemble-DL (reference) line. With 1,000 training samples, DINO+SVM reached 0.963 ± 0.010 while XCiT remained at 0.871 ± 0.026.

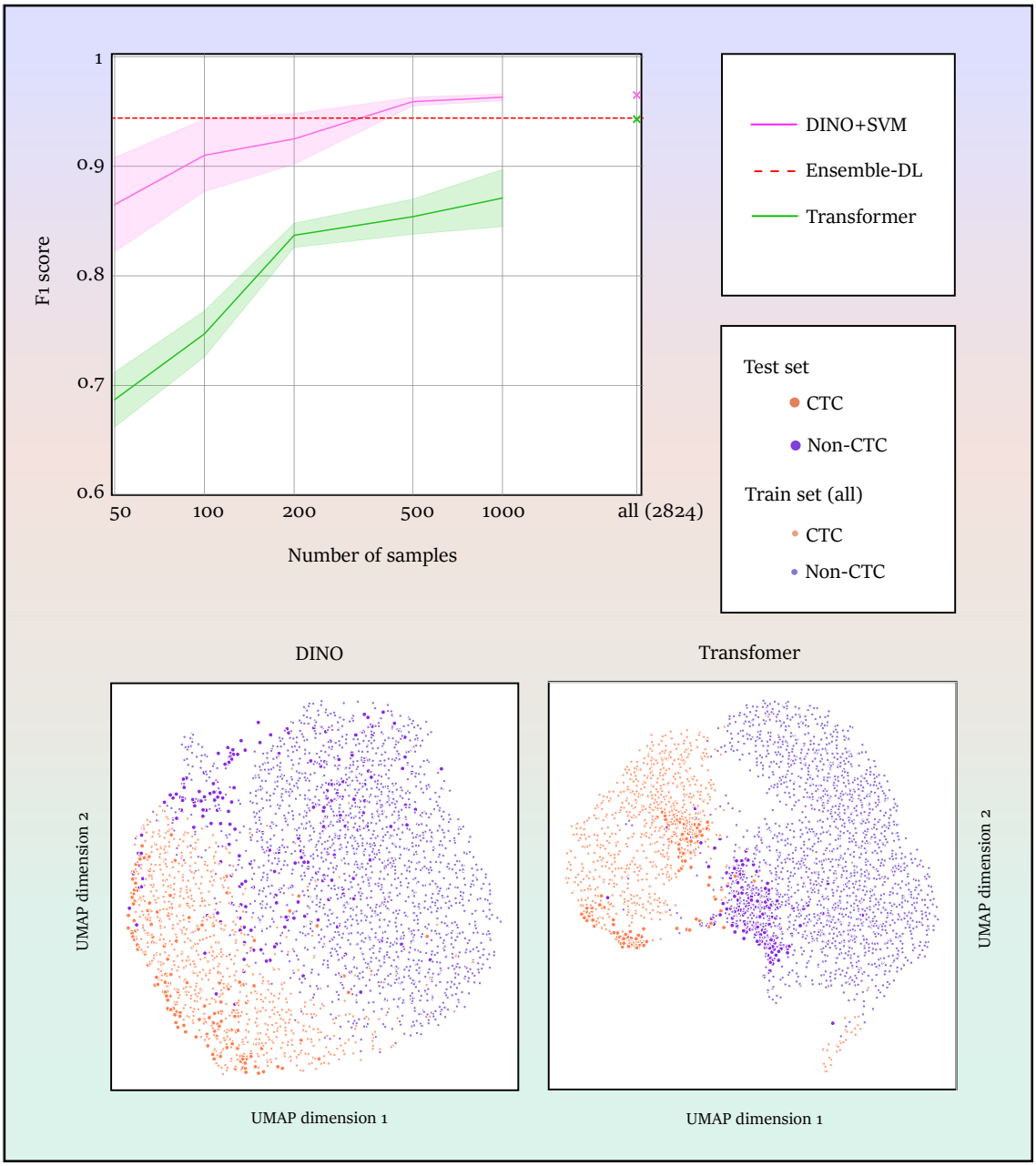


Figure 4.9.: Comparison of general and label efficient CTC classification performance between supervised and self-supervised approaches. Classification performance (F1 score) on the test set for the supervised XCiT and DINO+SVM models is shown for both the complete and limited numbers of labeled training samples, with means and standard deviations visualized. On the top: DINO+SVM results are shown in magenta, supervised XCiT (=transformer) in green and Ensemble-DL as a reference in red. For the Ensemble, all labeled training samples were used; therefore, its performance is represented by a horizontal dashed line. The cross indicates the general performance (training on the entire labeled training set). The lines refer to the balanced, limited training sets. There is no line between 1,000 samples and 2,824, since the latter is not a balanced class set. Below: Latent space analysis of the learned cell representations is shown. UMAP clustering of the CTC and non-CTC cell classes is displayed for both DINO and the transformer. CTCs are depicted in orange, non-CTCs in purple. Larger dots indicate samples from the GT test set, while smaller dots refer to GT samples from the entire available training set. Abbreviations and explanations: CTC: Circulating tumor cell; UMAP: Uniform manifold approximation and projection; DINO: Self-distillation with no labels. Figures adapted from [Hus+23].

The analysis then focused on how increasing the number of labeled training samples influences cluster formation in the latent space for the supervised XCiT model. As shown in Fig. 4.10, at 50 training samples, CTC clusters near the decision boundary appear highly heterogeneous. However, with the addition of more labeled data, the clusters become more homogeneous, indicating improved class separation with larger training set sizes (see Fig. 4.10 and Fig. 4.11). In contrast, DINO, despite not utilizing any labeled data, produces well-separated and homogeneous clusters for CTCs and non-CTCs near the decision boundary. Notably, the clustering behavior achieved by DINO resembles that of the supervised XCiT model only when the latter is trained on as many as 1,000 labeled training samples.

4.2.5. Discussion

Building on recent progress in both supervised and SSL, this study implemented and evaluated these approaches for binary CTC classification using fluorescence microscopy images, demonstrating their feasibility for LB analysis. The results showed that self-supervised feature extraction, when combined with an SVM classifier, can achieve higher classification performance (as measured by the F1 score) than state-of-the-art end-to-end trained supervised DL models, even when only minimal labeled data are available. This is particularly promising, given that data annotation is labor-intensive and that end-to-end trained DL models typically require resource-intensive retraining. In contrast, SSL-based image representations can be utilized for downstream tasks, such as CTC classification, by employing a standard ML classifier without the need for extensive additional training.

Furthermore, analysis of the latent space derived from the extracted features of both the SSL and supervised models revealed clearly separated clusters representing the two cell classes. Notably, the DINO approach achieved this cluster separation without incorporating any label information during training. The observed clustering patterns are also consistent with the quantitative improvements observed in classification performance.

Limitations of the study

This study focused on the feasibility of the DINO approach using LB data from 12 metastatic breast cancer patients. Since this represents a relatively small cohort, future work should incorporate data from a larger number of patients to enhance the robustness and performance of the self-supervised approach.

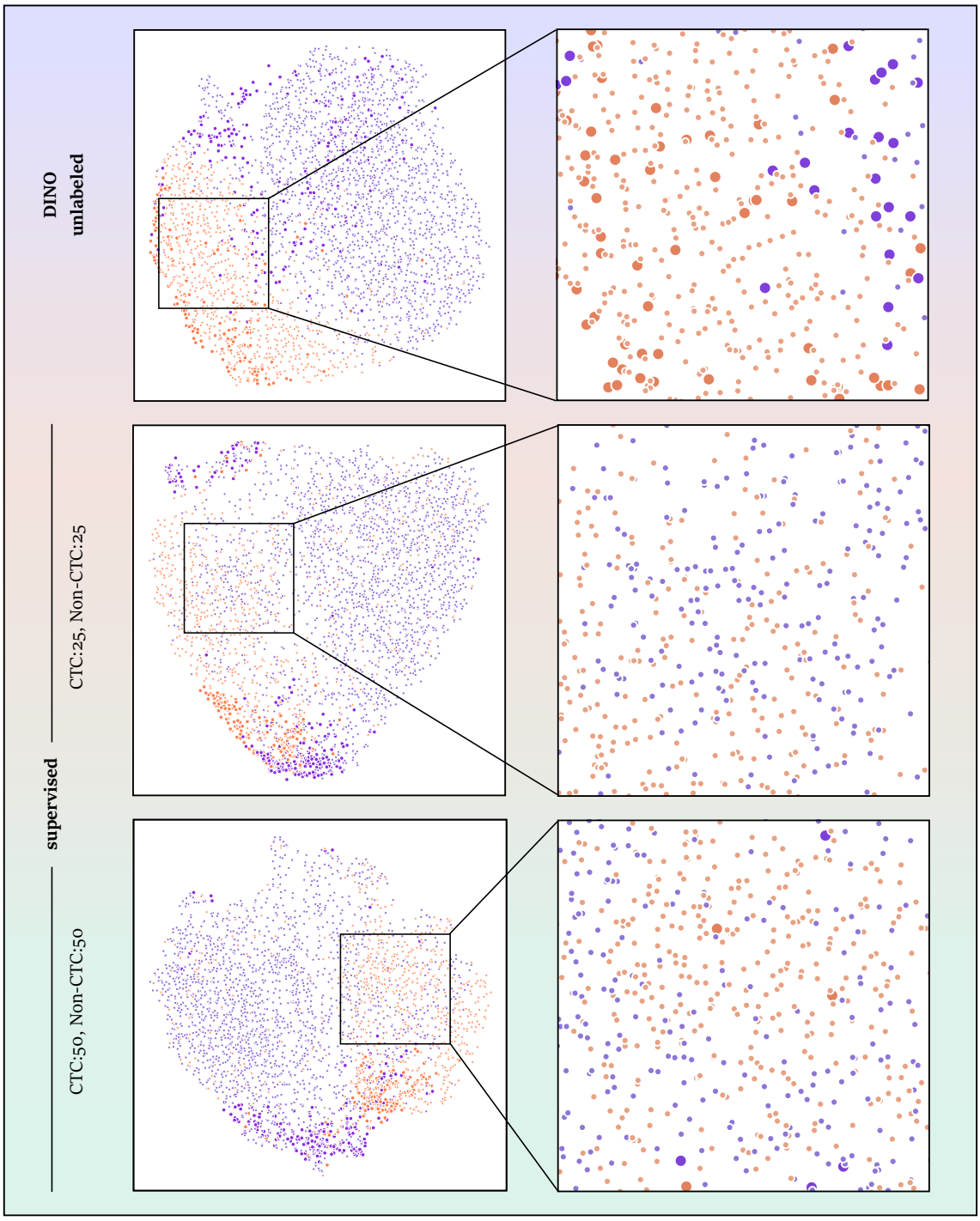


Figure 4.10.: Cropped UMAP visualizations highlight the area near the decision boundary in the latent space. For DINO, class distinction is apparent even near the boundary, while for the supervised transformer approach, cluster separation and homogeneity are poor when only 25 or 50 samples per class are available. CTCs are depicted in orange, non-CTCs in purple. Larger dots indicate samples from the GT test set, while smaller dots refer to GT samples from the entire available training set. Abbreviations and explanations: CTC: Circulating tumor cell; UMAP: Uniform manifold approximation and projection; DINO: Self-distillation with no labels. Figures adapted from [Hus+23].

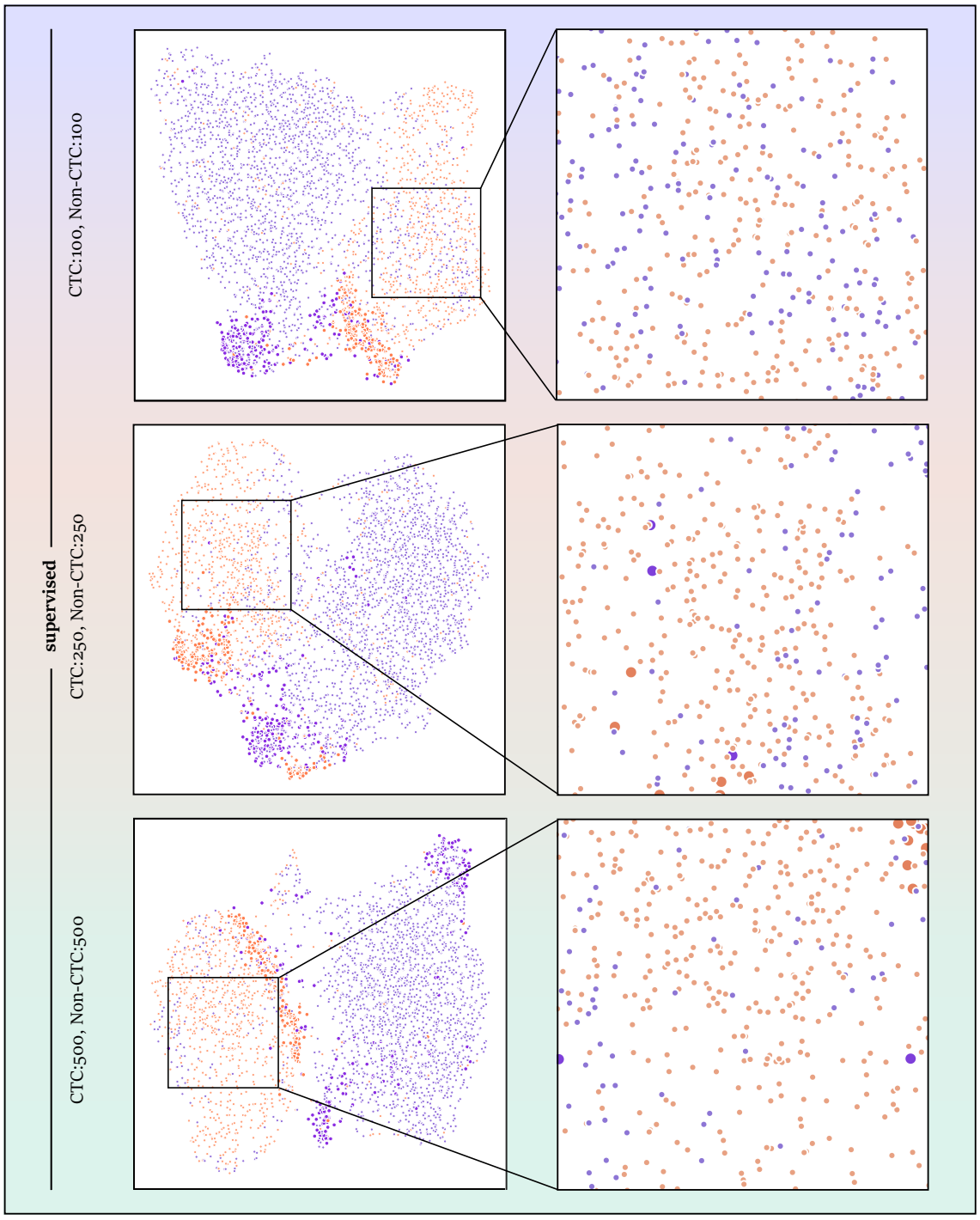


Figure 4.11.: Cluster separation and homogeneity improve for the transformer approach as the number of labeled samples increases from 100 to 500 per class. CTCs are depicted in orange, non-CTCs in purple. Larger dots indicate samples from the GT test set, while smaller dots refer to GT samples from the entire available training set. Abbreviations and explanations: CTC: Circulating tumor cell; UMAP: Uniform manifold approximation and projection. Figures adapted from [Hus+23].

4.3. Self-supervised learning and targeted human-in-the-loop strategy for improving circulating tumor cell classification

Building upon the potential of SSL for medical image data as reported by Nielsen et al. [Nie+23] but also for the liquid biopsy field as demonstrated in the previous study, the basis of the present study was the use of DINO in combination with an on top ML classifier for automated CTC classification of StarDist-extracted single-cell crops from cartridge images of an extended cohort of 90 metastatic breast cancer patients. In parallel, Nanou et al. [Nan+24] also addressed automated CTC classification using StarDist segmentation of cells from CS images and tested Zeune's DL network; however, they reported that relying solely on the DL classifier was too inaccurate to replace expert review. Furthermore, they observed many ambiguous cells in their data set and, to increase certainty in the labels of CTCs during training, they presented a semi-supervised approach in which newly labeled data from dense CTC and non-CTC latent space areas - identified by a KNN-based algorithm - were used to train another classifier (random forest).

However, by focusing on those areas, areas of high uncertainty, e.g. where most misclassifications happen, are neglected and remain uncertain. Targeting these uncertain regions for sampling and (re-)labeling may, in fact, be more effective in increasing a classifier's reliability, as these areas consist of image examples where the model lacks confidence in its predictions.

Consequently, in this work, efficient and targeted improvement of CTC and non-CTC image classification is pursued, with a particular focus on regions of elevated classifier uncertainty identified through latent space clustering. To achieve this, a human-in-the-loop (HiL) strategy is proposed, in which, as the name implies, human expertise is incorporated, given its value for further optimization of classifier predictions. Within the HiL framework, only a limited number of additional yet meaningful training samples from these uncertain regions - characterized by frequent misclassifications - are selected and provided to experts for (re-)labeling. By restricting the number of samples, this process reduces the annotation time required from experts, while the newly labeled data are subsequently incorporated into the classifier, enabling fast adaptation and improved performance in subsequent training loops. To demonstrate the feasibility of the proposed HiL strategy, the aims of this section are as follows:

(1) To provide a detailed analysis of the learned cell representations in the latent space and to investigate whether distinct regions correspond to differences in classification accuracy, which could reflect varying levels of model certainty.

(2) To assess the impact of different HiL sampling strategies - targeted and non-targeted - on classification performance when incorporating additional training images.

(3) To compare the CTC detection outcomes achieved by the targeted HiL approach with those obtained using the CS system.

The results are based on the following journal publication:

[Hus+25] H. Husseini-Wüsthoff, S. Riethdorf, A. Schneeweiss, A. Trumpp, K. Pantel, H. Wikman, M. Nielsen, and R. Werner. "Cluster-based human-in-the-loop strategy for improving machine learning-based circulating tumor cell detection in liquid biopsy". In: *Patterns* 6.6 (2025), p. 101285

The following section provides an overview of the full framework, incorporating the HiL principle. The description of the data set, the applied methods, and the individual experiments will be explained according to the overview shown in Fig. 4.12.

4.3.1. Data set

StarDist segmentation of cartridge images from 90 metastatic breast cancer patients resulted in 1,332,751 cell images (three channels: DAPI, CK, CD45). The data were then split into 60 patients (999,285 images) for SSL and 30 patients (323,466 images) for downstream CTC classification with the on top ML classifier. These 30 patients were further randomly divided into 10 for training (101,547 images) and 10 for testing (173,795 images) for the purpose of the HiL experiments (see Fig. 4.12A), while the remaining 10 patients (48,124 images) served as a hold-out test set for the final comparison with the CS system in terms of CTC detection. To establish a GT across training and testing sets (except for the hold-out test set), two experts labeled 1,509 CTCs and 1,129 non-CTCs for the training set, and 1,214 CTCs and 1,559 non-CTCs for the testing set. The remaining unlabeled images from the training set served as a pool for selecting and labeling additional images during the HiL experiments. To ensure the quality of the unlabeled training data set, an SVM was trained to identify and reduce the amount of noisy images - specifically, those with channel signals that were barely interpretable by human experts. For this purpose, a small subset of 800 unlabeled images was used for training the SVM, consisting of 400 images classified as noisy and 400 as non-noisy by expert assessment. The trained SVM was then applied to the entire unlabeled training data set, and all identified noisy images were removed. As a result, the refined and final unlabeled training data set comprised 48,312 samples.

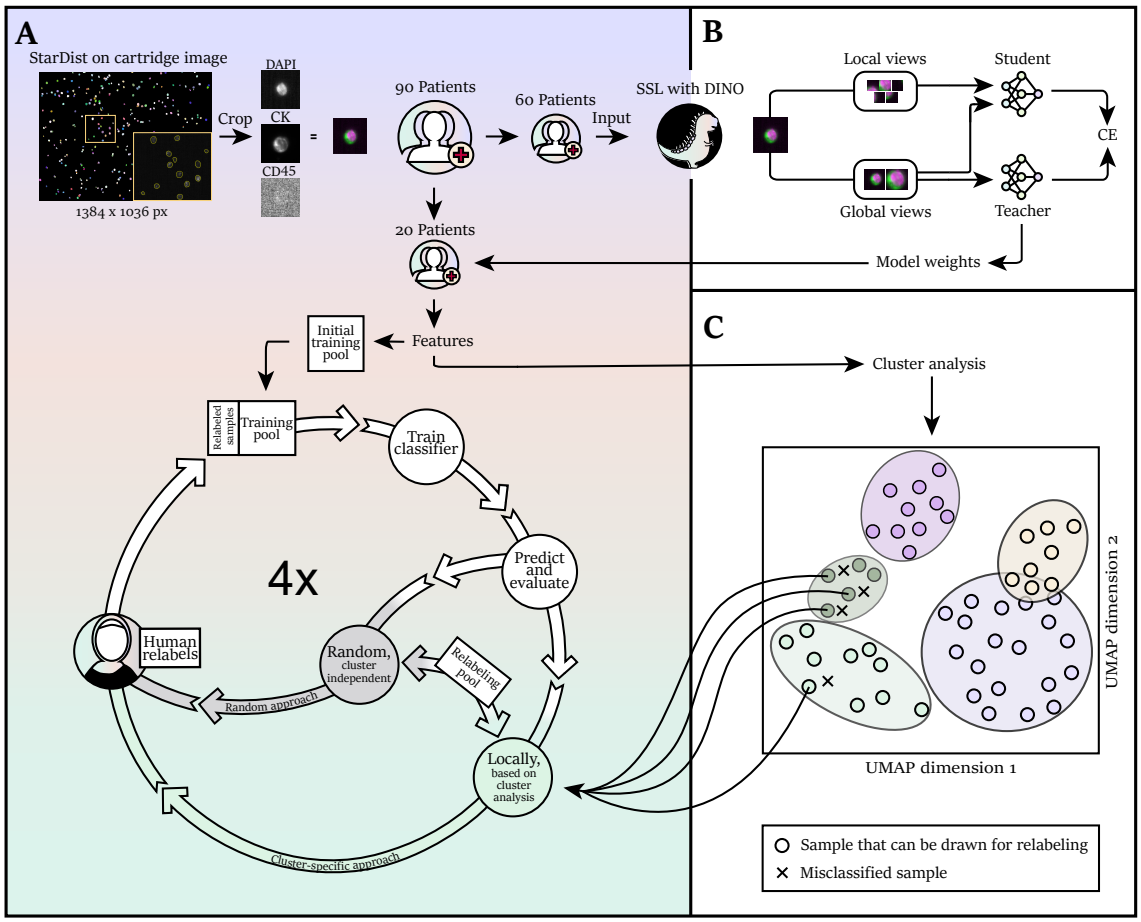


Figure 4.12.: Overview of the full framework for CTC detection and classification with the proposed HiL strategy. (A) The workflow begins with cartridge images from 90 patients acquired by the CS system. The StarDist pipeline is then used to segment cells and create image crops for each channel. The resulting single-channel images are merged to form a three-channel composite before being fed into the respective models. The 90 patients are split into groups of 60 and 20, with the latter group reserved for use within the HiL framework. The remaining 10 patients (out of the 90) are not shown and used for final evaluation and comparison against the CS system. (B) The unlabeled data from the 60 patients is used to train the DINO network. The model weights of the teacher network are utilized to extract features from the 20 patients described in (A), which are then used for binary CTC classification in the HiL experiments and for cluster analysis (C). (B) Within the HiL framework, an initial training pool is established before training and evaluating the ML classifier. To improve classifier performance, additional training images are sampled from a predefined relabeling pool. Sampling can be performed either randomly or based on a cluster analysis, where clusters with lower F1 scores are identified and images from these clusters are selected. Sampled images are then presented to an expert for labeling, after which the newly labeled images are added to the training pool. The HiL loop was performed four times. Abbreviations and explanations: DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45: Exclusion marker; SSL: self-supervised learning; CE: Cross-entropy; CS: CellSearch®; DINO: self-distillation with no labels; UMAP: Uniform manifold approximation and projection. Figure taken from [Hus+25].

4.3.2. Methods

DINO training: DINO was trained for 300,000 iterations on the unlabeled image data from 60 patients (see Fig. 4.12) using the public sparsam implementation by Nielsen et al. [Nie+23] of the DINO framework by Caron et al. [Car+21]. The setup was similar to that of the previous study (see Section 4.2.2) and the protocol described by Nielsen et al. [Nie+23]. The general concept is visually summarized in Fig. 4.12B. XCiT was used as the backbone, and after training, the backbone of the teacher model was used to infer image representations for subsequent latent space analysis and CTC classification.

DINO feature-based ML-CTC classification: For downstream classification, an SVM was selected for several reasons: (1) prior experiments (see Section 4.2) demonstrated good CTC classification performance in combination with the DINO approach; (2) Nielsen et al. [Nie+23] reported that SVM offered a slight performance benefit over other ML methods such as logistic regression (LR) and KNN in medical classification tasks; and (3) as further evidenced by the findings in Table 4.1, SVM consistently achieved better performance on the complete labeled test set, regardless of whether hyperparameter optimization was applied. Thus, an SVM was employed for all subsequent experiments,

ML model	Approach	Mean total F1 test score
KNN	Baseline	0.878
KNN	Bayesian Optimization	0.886
KNN	Proposed by Nielsen et al. [Nie+23]	0.887
LR	Baseline	0.907
LR	Bayesian Optimization	0.909
LR	Proposed by Nielsen et al. [Nie+23]	0.909
SVM	Baseline	0.924
SVM	Bayesian Optimization	0.921
SVM	Proposed by Nielsen et al. [Nie+23]	0.921

Table 4.1.: Comparison of mean total F1 test scores for different ML models and approaches (with and without hyperparameter optimization) on the labeled test set. The results correspond to one of the HiL experiments (simulated HiL scenario 2: limited local data (Section 4.3.3)), consisting of four loops, each repeated five times. In the final loop of each run, either SVM, KNN, or LR was used, with three approaches for classification: baseline, bayesian optimization and hyperparameters proposed by Nielsen et al. [Nie+23]. Baseline models were set up with the standard scikit-learn parameter settings. Bayesian optimization of the hyperparameters was conducted separately for each run. For each approach and ML model, the mean F1 score across all runs was calculated on the complete labeled test set. Abbreviations and explanations: KNN: K-nearest neighbors; LR: Logistic regression; SVM: Support vector machine; ML: Machine learning.

with parameters selected according to the recommendations of Nielsen et al. [Nie+23]. The SVM received as input the feature representation of the single-cell images, extracted from the trained teacher backbone model. This representation was then reduced from 128 to 32 dimensions using PCA, with these 32 components capturing more than 90% of the variance present in the labeled training features. The SVM was subsequently fitted on this labeled training set.

Latent space cluster analysis: The approach for identifying clusters in the latent space (see Fig. 4.12C) was to apply UMAP to further reduce the 32D PCA features to two dimensions, and subsequently perform clustering using HDBSCAN (Section 3.1.2). A key advantage of HDBSCAN is its ability to automatically determine the number of clusters as well as accommodate varying cluster shapes and densities. This is particularly beneficial, as the underlying structure of the data, apart from the two known cell classes (CTC and non-CTC), is not known in advance.

HDBSCAN was applied on the UMAP features of the joint unlabeled training and test sets, and the default parameters for HDBSCAN were used except for: the minimum cluster size, minimum number of samples, selection of cluster method and the cluster selection ϵ value (see [Hus] for further details). That is because relying solely on the default parameters of HDBSCAN led to the formation of hundreds of small micro-clusters. Drawing inspiration from one of the application examples in the HDBSCAN documentation [MHA23], setting only the minimum cluster size to 15 while leaving the remaining parameters at their default values resulted in a dominant cluster that encompassed almost the entire data space, alongside a few much smaller clusters. To achieve more fine-grained and homogeneous clusters, the leaf cluster method (Section 3.1.2) was chosen, which favors the selection of smaller, denser sub-clusters rather than merging them into one or two larger clusters. Additionally, the minimum number of samples and cluster selection ϵ value were adjusted for greater control over cluster formation. Increasing the minimum number of samples makes the clustering more conservative, which also complements the use of the leaf cluster selection method. To avoid the creation of excessive micro-clusters, ϵ was set to a value higher than the default of 0 to help merge clusters in regions of micro-clusters. By choosing $\epsilon = 0.25$, DBSCAN* clusters corresponding to this threshold are extracted from the condensed cluster tree. At the same time, HDBSCAN clusters that form at distances greater than 0.25 will remain unchanged.

The reasons for conducting the analysis on the UMAP features of both the unlabeled training set and the test set are the following: (1) if the test set were excluded from clustering and the analysis were carried out solely on the (labeled and unlabeled) training data, clusters would still form; however, precisely the regions characterized by the highest

uncertainty, namely, areas with the highest concentration of misclassifications in the test set, would be overlooked and not adequately captured or represented as distinct clusters (see Fig. 4.13); and (2) to mitigate the issue of limited size and potential bias in the labeled portion of the test set, the substantially larger and potentially more diverse unlabeled training data is incorporated.

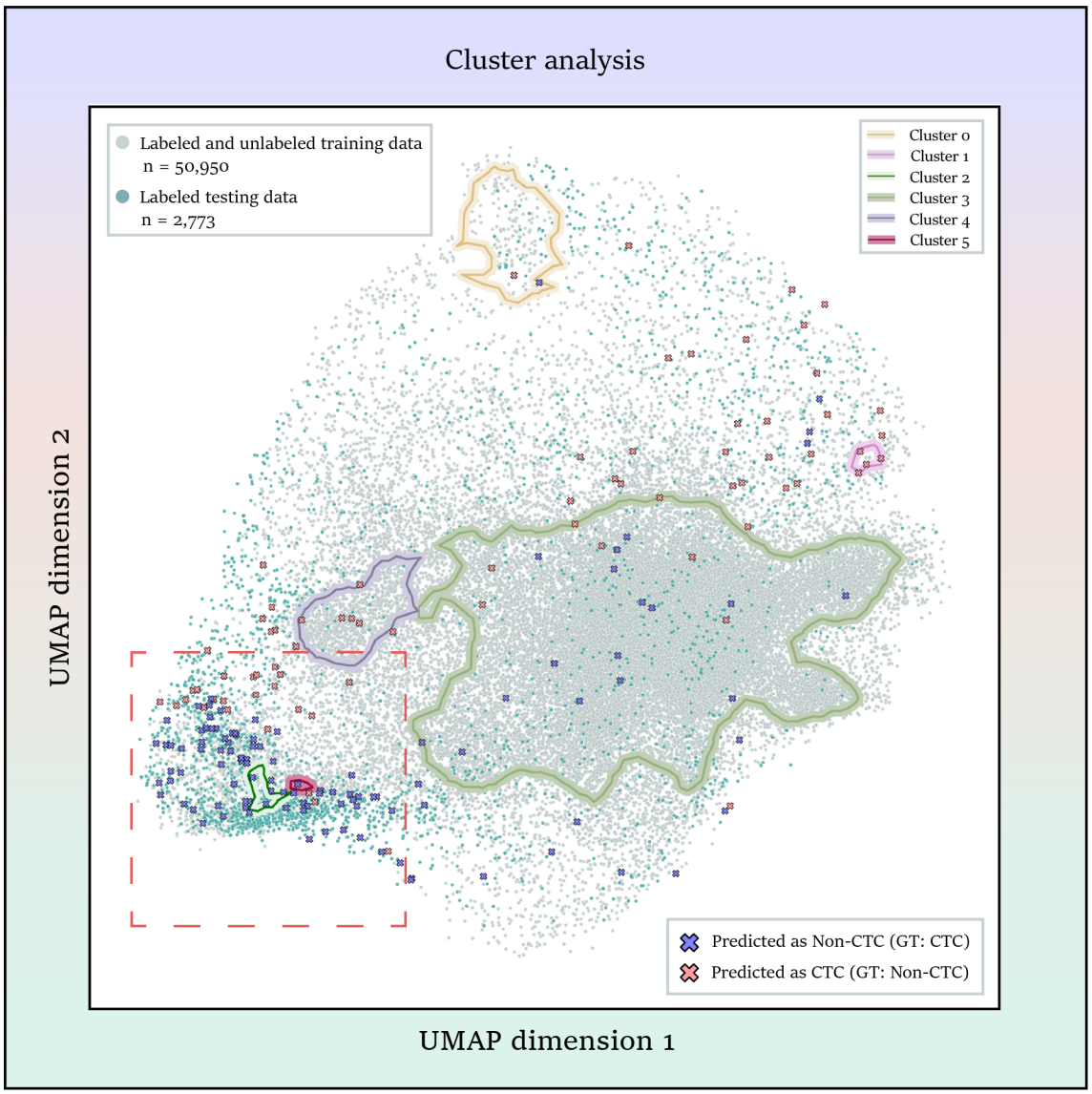


Figure 4.13.: Cluster distribution when performing HDBSCAN on the UMAP features of the combined labeled and unlabeled training data set. Turquoise-colored data points represent the labeled test data, while crosses indicate misclassifications. Grey data points correspond to the training set. The identified clusters are outlined with closed contours, while data points not enclosed by any cluster are classified as background. The red dashed box marks the region with the greatest concentration of test set misclassifications, an area largely not effectively captured by any cluster. Abbreviations: GT: Ground truth; UMAP: Uniform manifold approximation and projection. Figure taken from [Hus+25].

During evaluation, data samples that HDBSCAN did not assign to any cluster were labeled as background data. This analysis of latent space clusters provided the foundation for subsequent experiments using the proposed HiL strategy.

Proposed human-in-the-loop strategy: The proposed HiL strategy is an iterative process and has the following steps:

1. **Initialization**: Three sets of data are defined: an initial training pool, a relabeling pool (an unlabeled data pool for sampling and relabeling), and a test set for evaluation.

2. **Relabeling loop**: The following steps are iterated and repeated until either the classification performance meets the desired criteria or no additional data remain in the relabeling pool:

   (i) The labeled training samples are utilized to fit the ML classifier (SVM).

   (ii) The fitted classifier is used to perform classification of the labeled test set images.

   (iii) The classification performance is locally assessed and evaluated within the latent space clusters as well as on the labeled test set. The F1 metric is calculated for each cluster and for the complete test set.

   (iv) Cluster-specific sampling of new unlabeled data samples from the relabeling pool is carried out based on F1 scores.

   (v) The selected samples are reviewed by a human expert, who assigns class labels to them. These newly labeled images are then added to the training pool and simultaneously removed from the relabeling pool.

4.3.3. Human-in-the-loop experiments

A key element of the proposed HiL strategy is the cluster-specific sampling approach, which was proposed to enhance classification performance. The underlying hypothesis is that this targeted strategy can achieve improved results with fewer newly labeled samples compared to a naive, fully random sampling method. To evaluate this hypothesis, it was tested across multiple experimental scenarios: two simulation experiments (simulated HiL scenarios 1 and 2), which were conducted in an idealized environment, and, in addition, a real-world experiment in which a human expert labeled previously unseen and unlabeled data (real-world HiL experiment).

In the following, the respective HiL experiments will be presented. Each of these HiL experiments consisted of four relabeling loops, and each run was repeated five times using different random seeds to (to ensure variability in the sampling of the training and relabeling pools and to support reproducibility), so as to analyze the robustness of the findings. In line with the description of the proposed HiL strategy, the F1 score was used as the evaluation metric to assess CTC classification performance for each cluster and to report the overall F1 score.

Simulated human-in-the-loop scenario 1: Limited global data

This experimental setting simulates a scenario in which the quantity of labeled data is limited across the entire data set, as only a small fraction of the available labeled samples is used for initial model training, thereby restricting the information accessible to the classifier on a global scale. Classification improvements are therefore targeted through local training set adaptation. To achieve this setup, a random subset of 100 labeled samples was chosen from the available labeled training pool to initialize the SVM training data set. The remaining labeled data from the original training set was used as the relabeling pool for this experiment. In each relabeling loop, 100 additional samples were selected from the relabeling pool, so that after four loops (= one completed HiL run), the training data set was iteratively expanded to a total of 500 samples. Furthermore, in each loop, a 100-fold MCCV was performed on the current, limited labeled training pool by randomly splitting the data into 90% for MC training and 10% for MC validation at each step. This splitting was conducted without stratification in order to better simulate realistic data sampling conditions and to account for potential randomness in the selection process. Stratification would artificially enforce a specific class distribution, a level of information that may not be available in (real-world) sampling scenarios. All CV results were then aggregated, the F1 score for each cluster was assessed, and the classification performances for the individual clusters were used to guide the cluster-specific approach. New samples were selected from clusters in the relabeling pool with a relative frequency c_i inversely proportional to the corresponding MC validation F1 score s_i of cluster i, as shown in the following equation:

$$c_i = \frac{1 - s_i}{\sum_j 1 - s_j}. \tag{4.4}$$

The naive approach, on the other hand, was implemented independently of the cluster F1 scores. In each loop, 100 additional samples were randomly drawn from the entire available relabeling pool.

Simulated human-in-the-loop scenario 2: Limited local data

In contrast to the above-described limited global data setting, the limitation in the labeled training data set of simulated HiL scenario 2 is imposed only in a specific local region in the latent space, namely, a single main cluster. In order to do so, the initial training data set was constructed by reducing the number of labeled samples in the main cluster to only 20% of its original size, while the other clusters retained 80% of their original labeled training data. This composition of the training data leads to underrepresentation of the main cluster and results in reduced classification performance for this cluster compared to the others.

For the cluster-specific approach, the relabeling pool consisted of the remaining labeled training samples from the main cluster. During each relabeling loop, an additional 20% of the labeled samples of the main cluster were randomly drawn from this pool and added to the training set. The proportion of samples drawn in each loop was calculated based on the total number of labeled samples originally available in the main cluster. The sampling process was repeated until all labeled samples from the main cluster had been incorporated into the training set.

As for the random approach, the same initial labeled training set was used, but the relabeling pool consisted of 20% of the left-out labeled samples from each cluster, including the main cluster. Then, in each loop, samples were randomly selected from this relabeling pool, with the number of samples per loop consistent with that of the cluster-specific sampling strategy.

Real-world human-in-the-loop experiment

In addition to the simulated HiL experiments, a real-world HiL experiment was conducted in which labeling was performed by a human expert, rather than being simulated on artificially constrained labeled training data. In this experiment, the initial training set consisted of all available labeled training samples, and the relabeling pool comprised previously unseen and unlabeled training samples. Evaluation was conducted on the labeled test set, as described previously.

For the cluster-specific approach, the cluster with the lowest initial F1 score was identified, and the relabeling pool was constructed by exclusively sampling from this cluster. Specifically, all samples within the cluster that were predicted by the classifier as the class most frequently subject to erroneous predictions were included in the relabeling pool for further review. For example, if many actual CTCs were incorrectly predicted as non-CTCs, then all samples from the relevant cluster in the unlabeled training set that were classified as non-CTC formed the relabeling pool. The time given for human expert

review was set to 5 minutes to identify and relabel as many falsely predicted samples as possible from a pool of 1,000 samples. The newly labeled samples were subsequently incorporated into the training pool.

For the random sampling approach, samples predicted to belong to the same class as in the cluster-specific approach were selected from the entire unlabeled training set, without restricting them to any specific cluster. The number of falsely predicted samples relabeled per loop matched that of the cluster-specific approach.

Final model application

Lastly, the performance of the proposed HiL strategy was evaluated for CTC detection and classification and compared with the CS system. For this, the final model (after four HiL loops) of the cluster-specific real-world HiL experiment was used and tested on the hold-out test set (10 patients) mentioned in Section 4.3.1. As in the earlier experiments, the SVM's input comprised PCA-reduced representations of images generated by the trained teacher backbone model. For the identification of CTCs, the SVM decision function was set to a confidence threshold above 0.5; minor adjustments to 0.4 and 0.6 yielded similar positive predictive values. Furthermore, to ensure better enumeration of CTCs, duplicate cells were automatically identified and removed. After CTC classification for each patient, an expert reviewed the suggested candidates and assigned labels to the cells identified as CTCs.

For the CS system, gallery events were reviewed by the same expert, and CTCs were identified accordingly. To compare the CTCs detected by each method - i.e., to determine which cells were identified by both systems and which were exclusively detected by one system - cell coordinates were extracted from the XML files generated by the CS system and the IoU of the bounding boxes from both systems was then calculated, as detailed in Section 4.1.3. During evaluation, cases were taken into account in which a cell was classified as a CTC by one system and as a non-CTC by the other.

Subsequently, the positive predictive value was calculated, defined as the proportion of cells and events presented to the human observer that were confirmed to be actual CTCs.

4.3.4. Results

Detailed cluster analysis in the latent space

The cluster analysis revealed a total of five clusters, differing in size, shape, and F1 scores on the labeled test set, while unassigned points formed the background cluster, as illustrated in Fig. 4.14.

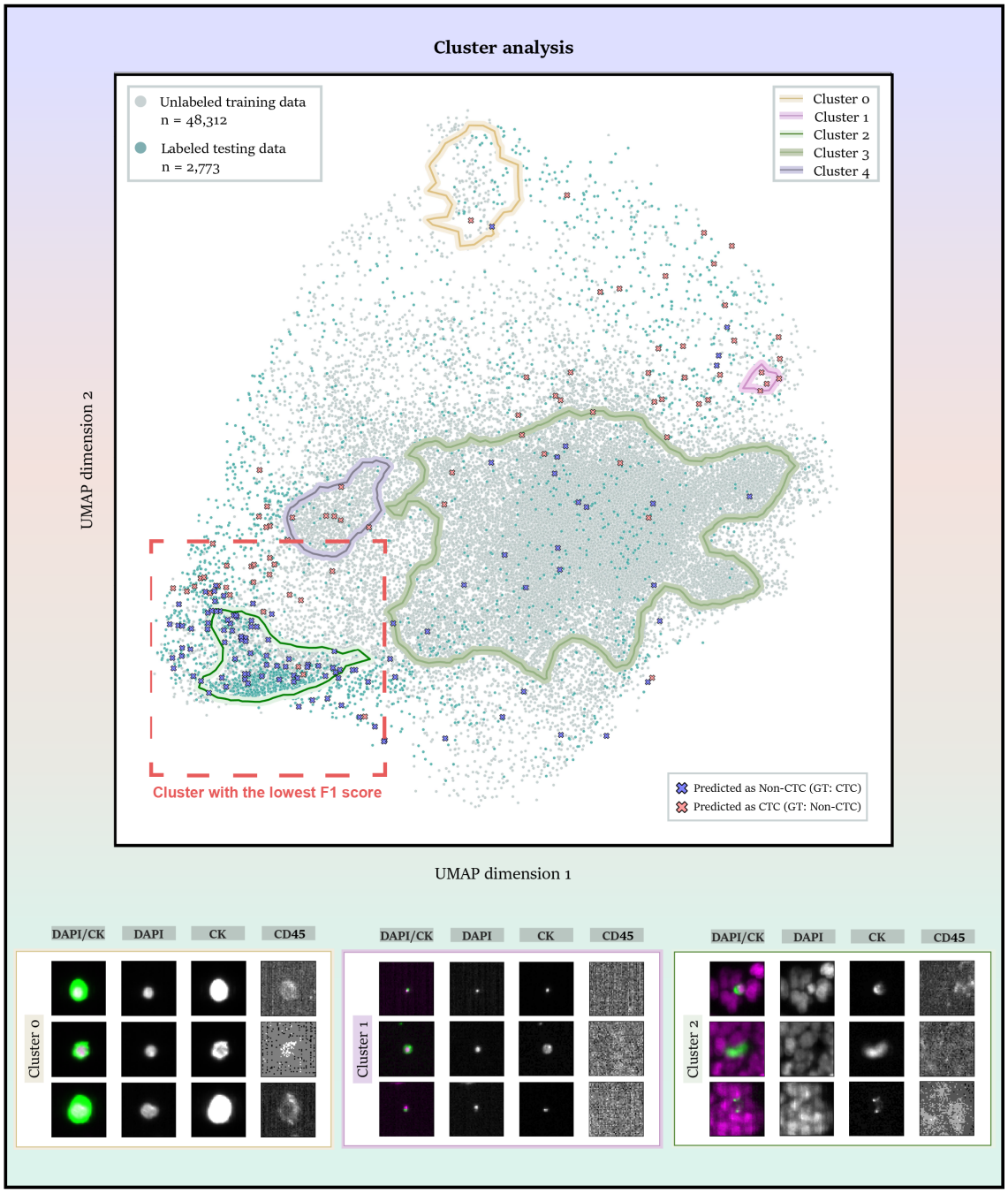


Figure 4.14.: Cluster analysis in the latent space reveals that the majority of the region with the highest concentration of test set misclassifications, as indicated by the red dashed box, is captured by a cluster, outlined with dark green contours. This cluster exhibits the lowest F1 score across all clusters. Data points outside of clusters are considered background. Results were obtained by applying HDBSCAN to the UMAP features of both the labeled test data and the unlabeled training data. Below the plot, exemplary cell images from clusters 0, 1, and 2 are shown: the first two rows display images from the labeled test data, and the third row presents images from the unlabeled training data. Abbreviations and explanations: DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45: Exclusion marker; GT: Ground truth. Figures adapted from [Hus+25].

The F1 scores for the five clusters were as follows: cluster 0 (yellow contours, see Fig. 4.14) with the highest score of 0.978; cluster 1 (pink contours) with 0.800; cluster 2 (dark green) with 0.542; cluster 3 (the largest cluster, light green) with 0.845; and cluster 4 (purple) with 0.929. Cluster 2 exhibited the lowest F1 score, accompanied by the most misclassifications both within the cluster and in its vicinity, as highlighted by the red dashed line box. A detailed examination of cluster 2 showed that the main misclassifications, both within and near the cluster, involved cells incorrectly predicted as non-CTCs, despite having been originally identified as CTCs by human experts.

In addition to analyzing the F1 scores and misclassification patterns, the biological relevance, that is, the meaningfulness, of each cluster was evaluated. Closer inspection of the individual clusters confirmed that cell images within the same cluster shared similar morphological characteristics. For instance, cluster 0 comprised images that displayed a shine-through phenomenon, caused by strong fluorescence signals in the CK channel that extended into the CD45 channel. Additionally, cluster 1 was characterized by small, point-like signals in both the DAPI and CK channels, whereas cluster 2 predominantly contained images with numerous background DAPI signals.

Impact of human-in-the-loop sampling on circulating tumor cell classification performance

After cluster identification and characterization, the HiL (simulated and real-world) experiments described in Section 4.3.3 were conducted to assess whether targeted sampling and labeling of cell images from regions of the latent space with lower F1 scores, based on the cluster analysis, can improve classification performance compared to random sampling.

Simulated human-in-the-loop scenario 1: Limited global data: The results of the first simulated HiL experiment, presented in Fig. 4.15A, show the average F1 score over five repetitions, based on evaluation on the test data set. Initially, cluster 2 exhibited the lowest F1 score at 0.107, whereas the other clusters began with higher scores, ranging from 0.684 for cluster 3 to 0.945 for cluster 0. The visualization of the latent space for cluster 2 and its surrounding areas reflects this poor F1 score by the numerous false predictions: many erroneous non-CTC predictions are concentrated in the northern region, while a higher occurrence of incorrect CTC predictions is observed in the southern region. Applying the cluster-specific method results in a substantially greater rise in the F1 score after four HiL loops, reaching 0.635, compared to just 0.260 for the random sampling approach.

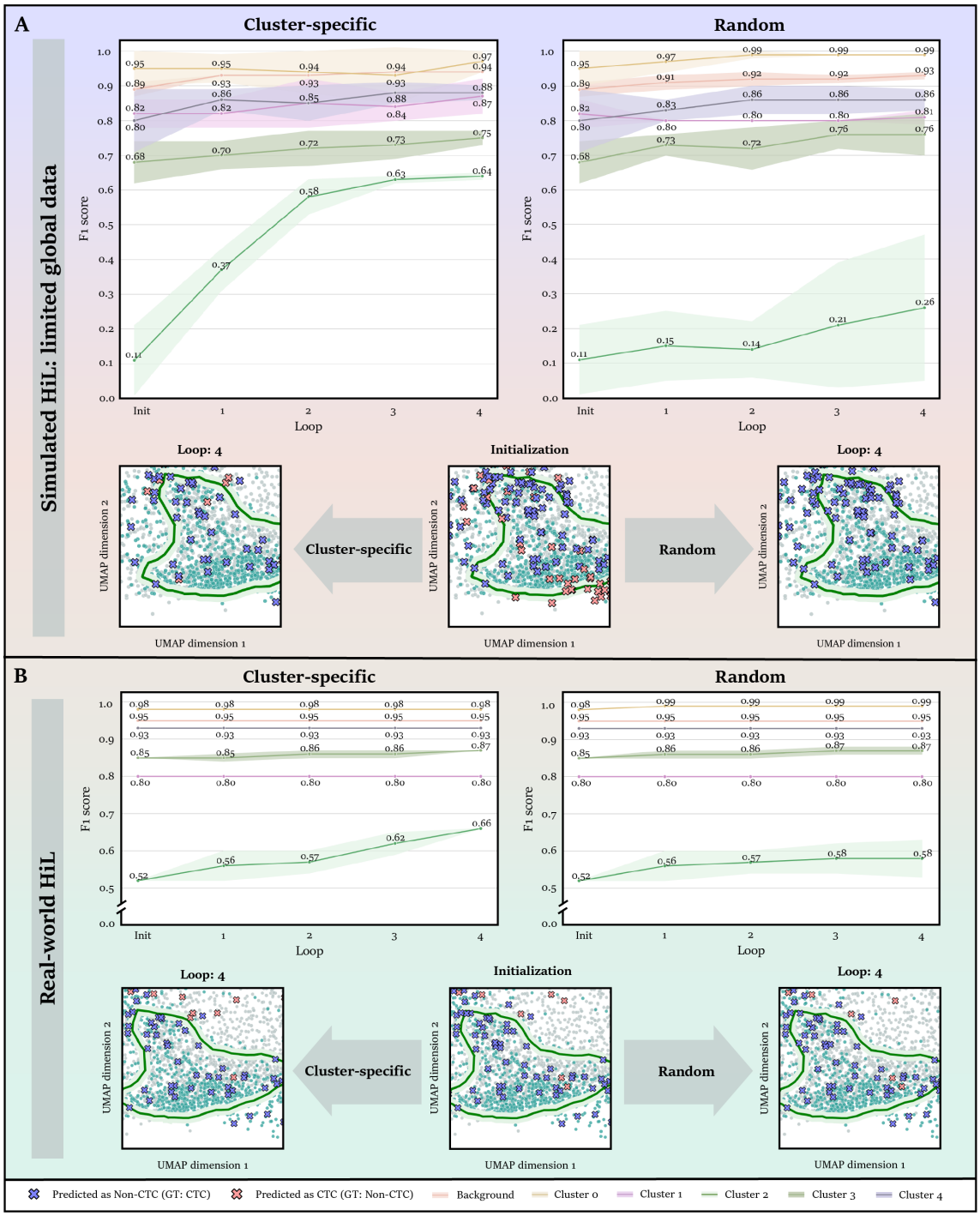


Figure 4.15.: Impact of HiL sampling, cluster-specific vs. random, on CTC classification performance. The line plots depict the mean F1 scores and standard deviations of the HiL loops over five repeated runs for each cluster, including the background cluster. Additionally, snapshots of the latent space are shown after initialization and after the final (fourth) loop, with a focus on cluster 2 (lowest F1 score; most misclassifications). (A) shows the results of the simulated HiL experiment with limited global data; (B) shows the real-world HiL experiment. Abbreviations and explanations: HiL, human-in-the-loop; Init, initialization; loops 1-4, sampling and relabeling loops. Figure taken from [Hus+25].

The effect of the targeted HiL sampling is also apparent in the corresponding snapshots (see Fig. 4.15A), confirming a reduction in false predictions, especially in predcited, erroneous non-CTCs, compared to the random approach. Regarding the classification improvements in the other clusters, in both approaches, most clusters showed increased F1 scores after four HiL loops, although the improvements were less pronounced than for cluster 2.

In the cluster-specific approach, new samples were drawn from clusters with a frequency inversely proportional to their F1 scores. Thus, as cluster 2's F1 score improves, the likelihood of selecting samples from other clusters for labeling increases. Starting from 0.849, the average F1 score increased to 0.911 with the cluster-specific approach after four HiL loops, whereas the random approach reached 0.896.

To reach an F1 score of 0.911 with random sampling, an average of five additional HiL loops (range: 2-6) was required, corresponding to an extra annotation effort of 500 samples (range: 200-600) per repeated run by an expert. An expert required about 5 minutes to annotate a set of 100 randomly chosen cells, providing a basis for estimating the potential time saved with the targeted approach. The additional annotation effort required for naive sampling was therefore estimated to be around 25 minutes, highlighting the time savings in manual CTC evaluation and annotation achievable with the cluster-specific approach.

Simulated human-in-the-loop scenario 2: Limited local data: The second simulation experiment addressed the challenge of limited training data within a specific region, i.e., a cluster, of the latent space. Such a scenario may arise when the images and cell representations of a new patient differ from the others used to train the classifier. The results, summarized in Table 4.2, focus on two clusters: cluster 2, which had the lowest F1 score, and cluster 3, which is the largest cluster.

When cluster 2 was selected as the main cluster, thereby initially hampering its classification performance, the cluster-specific approach achieved a higher F1 score for cluster 2 (0.492) compared to random sampling (0.456) after four HiL loops. This was also reflected in the overall test set evaluation, with F1 scores of 0.921 for cluster-specific HiL and 0.919 for random sampling.

Similarly, when cluster 3 served as the main cluster, the cluster-specific HiL strategy outperformed random sampling for both the main cluster (F1: 0.852 vs. 0.809) and the neighboring cluster 2 (F1: 0.485 vs. 0.423).

	Simulated HiL: limited local data			
	Main cluster: cluster 2		Main cluster: cluster 3	
F1 score	**Cluster-specific**	**Random**	**Cluster-specific**	**Random**
Total - init	0.919 ± 0.003	0.919 ± 0.003	0.909 ± 0.003	0.909 ± 0.003
Total - loop 4	0.921 ± 0.002	0.919 ± 0.003	0.921 ± 0.003	0.916 ± 0.003
Background - init	0.945 ± 0.001	0.945 ± 0.001	0.942 ± 0.002	0.942 ± 0.002
Background - loop 4	0.946 ± 0.001	0.945 ± 0.002	0.946 ± 0.002	0.946 ± 0.001
Cluster 0 - init	0.982 ± 0.006	0.982 ± 0.006	0.985 ± 0.006	0.985 ± 0.006
Cluster 0 - loop 4	0.982 ± 0.006	0.985 ± 0.006	0.980 ± 0.005	0.980 ± 0.005
Cluster 1 - init	0.800 ± 0.000	0.800 ± 0.000	0.818 ± 0.040	0.818 ± 0.040
Cluster 1 - loop 4	0.800 ± 0.000	0.800 ± 0.000	0.808 ± 0.019	0.800 ± 0.000
Cluster 2 - init	0.432 ± 0.087	0.432 ± 0.087	0.315 ± 0.082	0.315 ± 0.082
Cluster 2 - loop 4	0.492 ± 0.056	0.456 ± 0.109	0.485 ± 0.079	0.423 ± 0.029
Cluster 3 - init	0.854 ± 0.012	0.854 ± 0.012	0.799 ± 0.016	0.799 ± 0.016
Cluster 3 - loop 4	0.847 ± 0.018	0.847 ± 0.018	0.852 ± 0.017	0.809 ± 0.027
Cluster 4 - init	0.922 ± 0.015	0.922 ± 0.015	0.895 ± 0.024	0.895 ± 0.024
Cluster 4 - loop 4	0.919 ± 0.009	0.919 ± 0.009	0.919 ± 0.009	0.919 ± 0.009

Table 4.2.: Classification performance comparison: cluster-specific versus random sampling approaches in simulated HiL experiment 2 with limited local data. The values presented correspond to the average F1 scores and their standard deviations calculated for all clusters, with the background cluster included in the analysis. "Total" indicates the F1 score calculated for the entire labeled test set. Abbreviations: Init: Initialization; HiL: Human-in-the-loop. Table taken from [Hus+25].

Real-world human-in-the-loop experiment: In the real-world HiL experiment, the initial classifier was trained on the complete labeled training set, with further refinement achieved through expert labeling of unlabeled training data. The results are depicted in Fig. 4.14B. For the cluster-specific approach, sampling was restricted to the region of cluster 2, as it had the lowest F1 score, and only images predicted as non-CTCs within this cluster were selected for expert review, since most misclassifications in cluster 2 were false non-CTC predictions. Among these, only samples identified as CTCs by the expert were added to the training pool. In contrast, for the random approach, the same class of predictions (non-CTCs) was sampled across the entire data set without restriction to cluster 2. During a single run of the HiL process using the cluster-specific approach, a total of 32 new samples were labeled as CTCs across four loops (loop 1: 11; loop 2: 10;

loop 3: 7; loop 4: 4). As no additional CTCs were identified, the newly labeled samples remained the same for the remaining repeated runs but were shuffled, ensuring that each loop contained the same number of new samples as previously. Despite adding only a small number of new samples to the training set, the proposed HiL strategy led to an increase in the F1 score for cluster 2 from an initial value of 0.524 to 0.661 after four HiL loops. In comparison, applying a random sampling strategy resulted in an F1 score of 0.578 after four loops. The more modest increase in F1 observed in this case can be attributed to the fact that, on average, only about 4 new cluster 2 samples (range: 2-5) were added to the training data set per run. Additionally, many cells were also sampled from the background cluster. When correcting non-CTCs to CTCs, this may also influence performance in cluster 2, as many falsely predicted non-CTCs are located in the vicinity of cluster 2. Further, for both approaches, minor improvements in the F1 score were also observed in neighboring cluster 3. Additionally, with an initial overall F1 score of 0.923 on the fully labeled test set, the cluster-specific method reached a higher F1 score of 0.930, compared to 0.926 obtained with random sampling.

Comparison of detection and classification performance between proposed human-in-the-loop model and CellSearch®

To assess CTC detection performance, the results of the final model from the cluster-specific real-world HiL experiment and the CS system, evaluated on the hold-out test patients, are shown in Table 4.3.

Across 10 patients, for each detection system, the number of suggested CTC candidates or CS events, the number of confirmed CTCs, and the positive predictive value, defined as the fraction of proposed cells or events that are actual CTCs, are reported and compared. While both systems detected a similar total number of actual CTCs, the proposed pipeline consistently achieved a higher positive predictive value for all patients, resulting in fewer false-positive images that required review.

To further evaluate detection performance, an analysis was conducted on the actual CTCs identified by each system to assess the overlap in CTC detection and to determine whether unique CTCs were identified by either approach. In the case of several patients, certain CTCs identified by the CS system were missed by the HiL approach (see Fig. 4.16B; CTCs found exclusively by CS across patients: range 2-18, average 7), while others were detected only by the proposed pipeline (see Fig. 4.16A; range 0-26, average 6). Among the CTCs not identified by the CS system were those with relatively low signal intensity in the CK channel (see the first and fourth CTCs, Fig. 4.16A) or in the DAPI channel (see

the third row, Fig. 4.16A), as well as a CTC with a relatively small DAPI signal in the second row of Fig. 4.16A, where the DAPI signal overlaps with CK.

	Final HiL model			CS		
Patient	**Suggested CTC candidates**	**Actual CTCs**	**Positive predictive value**	**Events**	**Actual CTCs**	**Positive predictive value**
1	91	77	0.846	239	74	0.310
2	135	104	0.770	531	107	0.202
3	91	76	0.835	219	80	0.365
4	38	24	0.631	101	23	0.228
5	225	124	0.551	1,197	113	0.094
6	71	34	0.479	168	38	0.226
7	50	31	0.620	133	35	0.208
8	23	20	0.870	167	20	0.120
9	23	14	0.609	93	16	0.172
10	148	130	0.878	790	144	0.182

Table 4.3.: Overview of CTC detection results for the proposed HiL system ("Final HiL model") and the CS system. "Events" denotes the images shown in the CS gallery. Positive predictive value is defined as the proportion of cells and events presented to the human observer that are verified as actual CTCs. Abbreviations: CTC: Circulating tumor cell; CS: CellSearch®. Table taken from [Hus+25].

4.3.5. Discussion

Recent advances in CTC detection have focused on automating the process using ML techniques - including supervised, semi-supervised, and, most recently, self-supervised methods - largely to address the limitations of the FDA-approved CS system. While full automation remains a central goal, it can be argued that the integration of human expertise is still essential, particularly when the ML system encounters uncertainty or when there is a mismatch between the training data and specific patient samples. This underscores the continued importance of human involvement, such as in a HiL process.

Building on this perspective, a novel HiL strategy was introduced in the present study, in which DINO feature extraction was first combined with a conventional ML classifier for CTC classification. Based on the resulting latent space representations, challenging clusters were then identified, enabling expert labeling efforts to focus specifically on regions of high uncertainty (e.g., low F1 score), thereby improving classifier performance

and confidence more rapidly compared to random sampling, while reducing the time required from experts. The effects of targeted HiL sampling and labeling were most evident for the cluster with the initially lowest F1 score. Moreover, the overall classification accuracy achieved by the proposed targeted HiL framework exceeded that reported by Nanou et al. [Nan+24] for their metastatic breast cancer testing set, with higher precision (0.938 vs. 0.814) and recall (0.923 vs. 0.793) for CTC detection.

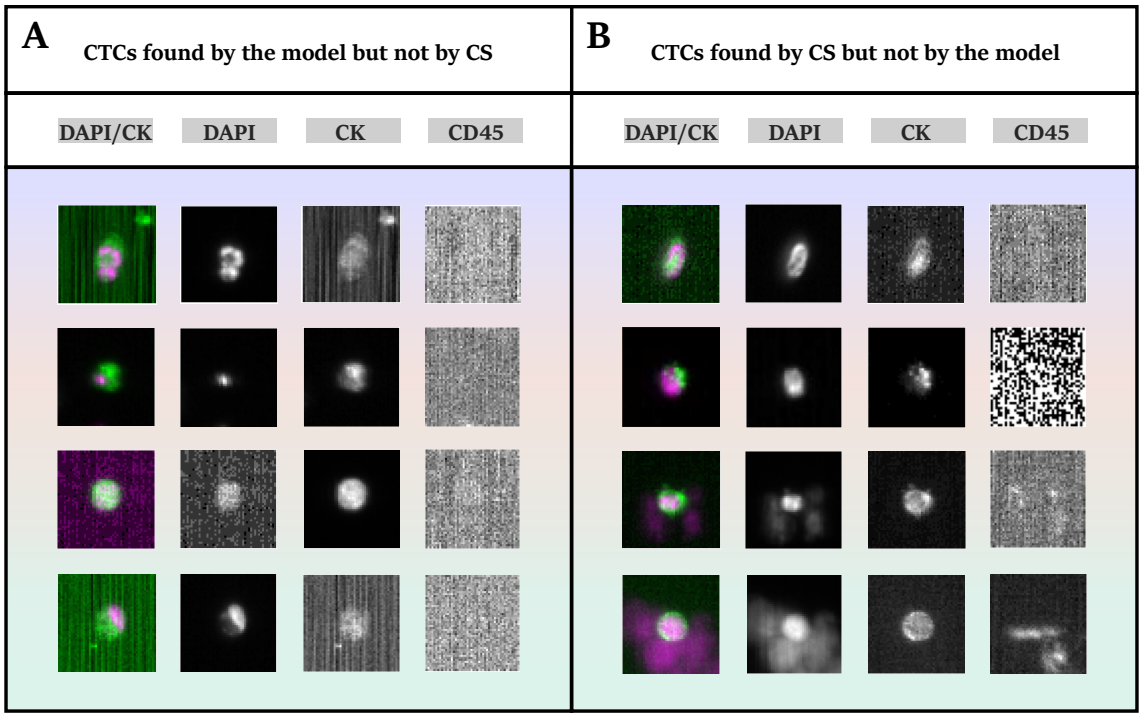


Figure 4.16.: CTC detection comparison between the proposed HiL model and the CS system. (A) CTCs identified by the model but missed by CS; (B) CTCs detected by CS but not by the HiL model. Abbreviations and explanations: CTC: Circulating tumor cell; DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45: Exclusion marker; CS: CellSearch®. Figure taken from [Hus+25].

The advantages of a targeted sampling strategy become even more apparent in clinical-like scenarios where an expert reviews CTC candidates from multiple patients. In the galleries generated by the CS system, a large number of events are presented - particularly in the metastatic setting - because the system is designed to display events with close DAPI and CK signals rather than specifically proposing CTC candidates, resulting in a lengthy and labor-intensive review process. This was also demonstrated in Table 4.3: the CS system required the review of 3,638 events across 10 new patients, whereas the final HiL model of the targeted (cluster-specific) approach reduced the number of cells needing expert review to 985, amounting to a reduction by a factor of approximately 3.7, while still identifying a similar number of actual CTCs as the CS system. Although the time required by an expert for CTC review can vary depending on factors such as

experience, the total number of images presented, and interpretation challenges stemming from signal intensities, background cells, and noise, increasing the image volume further elevates the possibility of expert fatigue, longer annotation times, and a higher risk of errors. Ultimately, for metastatic cancer cases, it would be ideal to eliminate the need for experts to review every image and to reduce evaluation time to a minimum, since the cut-off value for poor prognosis is already 5 CTCs per 7.5 ml blood sample - and prognosis declines further as the CTC count increases. To move toward this goal, in a clinical setting where the HiL approach is applied, users can assess, for each blood draw, whether the system's accuracy is likely to be high or poor for specific latent space clusters. If uncertainty is detected, the system can prioritize less certain CTC candidates for expert review. Feedback from these expert evaluations is then used to refine the classifier for those clusters. This targeted approach supports the development of highly adaptable classifiers and reduces review time when analyzing future samples with similar characteristics.

Limitations of the study

The current study focused on the feasibility of the proposed targeted HiL strategy and demonstrated comparable CTC detection performance to the CS system. However, the underlying reasons why each system identified certain CTCs that the other did not remain to be investigated. These differences in CTC detection may stem from the distinct thresholds employed by each method, such as segmentation and confidence thresholds in the HiL system (StarDist and SVM, respectively) and segmentation thresholds in the CS system. However, since the CS algorithm is not publicly accessible, its exact implementation and threshold settings remain unknown. To better understand these discrepancies, a visual comparison of both detected and missed CTCs on cartridge images from each system could help identify specific cases in which CTCs are overlooked.

Another relevant aspect, highlighted by Stevens et al. [Ste+22], is that the CS system often has segmentation difficulties in regions of high cell density and faces additional challenges when segmenting events located at the edges of its multiple adjacent image tiles. Therefore, it would be insightful to examine these challenges in both the CS system and the proposed approach in future studies.

Another important direction for future work includes the following: (1) incorporating data with lower reported CTC counts, given the current consensus threshold of 5 CTCs per 7.5 mL blood draw, and extending the analysis to include samples from healthy donors; (2) including additional tumor entities, such as metastatic prostate or colon cancer, for which the CS system is also approved; (3) expanding phenotypic characterization of CTCs

by incorporating additional markers - such as HER2 or estrogen receptor positivity - using the fourth channel of the CS system, which could further facilitate the distinction of different molecular subtypes.

4.4. Analyzing CTC detection concordance and sensitivity: Human-in-the-loop model and CellSearch® in patients and healthy donors

While the previous results section demonstrated that the final HiL model achieves CTC detection performance comparable to the CS system in metastatic patients, this comparison was conducted on samples from individuals with CTC counts above the prognostic cut-off of five. In fact, most of the 30 HiL patients (across training, testing, and hold-out test sets) exhibited elevated CTC numbers and were experiencing progressive disease.

In this section, the focus is shifted to a metastatic breast cancer cohort in which only a few patients are found to have progressive disease, while most have a CTC burden below the cut-off threshold. Some of these patients may be in earlier metastatic stages or display signs of disease stabilization - possibly responding more effectively to ongoing therapy. This raises a clinically relevant question: can the detection performance and concordance demonstrated by the final HiL model and CS in advanced disease be generalized to this new cohort?

For such a group, it is essential to determine whether all CTCs detected by CS are also identified by the final HiL model at the chosen confidence threshold, particularly since missing even a small number of CTCs below the cut-off carry greater clinical significance - unlike in advanced disease, where high CTC counts lessen the impact of a few missed cells. To comprehensively assess the sensitivity of the model in this context, both patients and healthy donors are included in the analysis. Accordingly, two main questions and experiment scenarios are addressed:

(1) How many CTCs are detected when a confidence threshold is applied?

(2) How does the number of detected CTCs change when the number of images for review is matched to the number of CS events?

For both of these questions, additional sub-questions - previously addressed in the comparative CTC detection analysis in Section 4.3.4 and of particular importance in this cohort - are taken into account:

(a) How many cells are detected by both systems and consistently confirmed as CTCs by the expert?

(b) How many cells are detected by only one system but not by the other and are confirmed as CTCs by the expert? What are the morphological characteristics of these cells? Of the CTCs detected by CS but not by the HiL model (in both main questions above), how many were segmented by StarDist?

(c) How many cells are regarded as unsure? How many are detected by both systems? Which are detected by only one system but not by the other?

In healthy donor samples, it is expected that only non-CTCs will be present. However, sub-question (c) is of particular interest: if any ambiguous or "unsure" cells are identified, it becomes relevant to examine whether the characteristics of these cells resemble those observed in the patient cohort, or if discrepancies between the groups can be identified.

4.4.1. Data set

Cartridge images from 73 patients and 10 healthy donors were included in the analysis. The clinical classification of patients into categories such as progressive or stable disease was not part of the analysis and therefore not available during the evaluation. StarDist segmentation was performed on all cartridge images, resulting in a total of 837,673 three-channel images (patients: 783,883; healthy donors: 53,790).

4.4.2. Methods and experiments

Pipeline: The image representations of the StarDist-segmented cells were extracted using the trained DINO teacher model, as detailed in Section 4.3.2. Duplicate cells were then removed, including those occurring at the edges of cartridge frames and thus also appearing in the adjacent frame. Then, the trained SVM described in Section 4.3.1 was applied on the PCA reduced features (see Section 4.3.2 for methodological details) to reduce the number of noisy images. For experiment scenario (1) - identifying CTC candidates based on a confidence threshold above 0.5 - the final HiL model described in Section 4.3.3 was used. For experiment scenario (2), aimed at increasing the number of cells for review to match the number of n CS events, the same model was applied and the n images with the highest confidence scores were selected. This approach always included all CTC candidates from experiment scenario (1), as the number of CS events exceeded the number of candidates from (1) across all patients and healthy donors. The model candidates in experiment scenario (2) will hereafter be referred to as model events.

Annotation: For the CS data, annotation was performed on the events directly within the CS system. All manually selected events were regarded as CTCs, while non-selected

events were not further differentiated; these could represent non-CTCs, overlooked CTCs, or ambiguous cells. Similarly, the selected CTCs may have included cells with ambiguous features, but further subclassification was not possible within the system. However, the Cell Select tool was available and could be used for reboxing across the different channels for each cell of interest.

For the proposed model-based approach, model events from experiment scenario (2) were annotated, as these already included the suggested cells from scenario (1). During this process, three categories were defined overall: CTC, non-CTC, and unsure cells. Unsure cells were those that could not be clearly classified as either CTC or non-CTC based on channel intensity signals or cell size. The likelihood of encountering unsure cells was higher in the model-based approach than in the CS system approach, in part because the Cell Select tool was not available for detailed review. Upon closer inspection of the unsure cells, they were further subdivided into two categories: unsure cells (in general) and particle-like, unsure cells. The particle-like cells are smaller in size, showing a strong CK signal but a relatively weak and small DAPI signal.

Correspondence to CS: Following the description in Section 4.1.3 and the comparative analysis in Section 4.3.3, the cell coordinates of the CS events, including the selected events, were extracted and converted to match the dimensions of a cartridge frame. Subsequently, the overlap of bounding boxes between cells identified by the CS system (both selected and non-selected events) and those from the model-based approach (StarDist, suggested candidates, and events) was calculated.

To compare which CTCs and unsure cells were detected by both systems or only by one and not the other, specific criteria were defined and applied in both experiment scenarios (1) and (2):

- **CTCs found by both systems**: Cells annotated as CTCs in the model-based gallery that also corresponded to selected CS events. If a CTC (in the model gallery) was also present among the non-selected CS events (presumably only missed during selection), it was included in this category and added to the total count of CS-identified CTCs. Likewise, if a cell was not annotated in the model gallery (possibly missed during labeling) but was a selected CS event, it was still regarded as detected by both systems.
- **CTCs found only by the model**: CTCs identified solely by the model-based approach and absent from all CS events (selected and non-selected).
- **CTCs found only by CS**: Selected CS events not detected by StarDist - i.e., lacking a segmentation mask - were considered CTCs identified only by the CS system.

- **Unsure cells (in general) detected by both systems**: Cells labeled as non-CTCs in the model-based gallery but were selected in the CS system, as well as any cell marked with a question mark in the model-based gallery and also appearing among the CS events, were considered ambiguous or unsure.
- **Unsure cells (in general) detected only by the model**: Cells marked with a question mark in the model-based gallery that did not overlap with any CS event were included here.
- **Particle-like, unsure cells detected by both systems**: Explicitly annotated particle-like cells in the model-based gallery, regardless of question mark status, identified by both systems.
- **Particle-like, unsure cells detected only by the model**: Particle-like cells recognized exclusively by the model-based approach and not present in any CS event.

It is important to note that both categories of unsure cells (unsure (in general) and particle-like cells) detected by both systems were not counted as actual CTCs, even if they were initially selected as CTCs in the CS system. Therefore, the actual number of CTCs by CS reflects the total number of CTCs attributable to the CS system after accounting for cross-system matches and excluding ambiguous cases. The calculation is as follows:

$$\text{Actual CTCs}_{\text{CS}} = (\text{CS-sel CTCs}) + (\text{CS non-sel CTCs}) - (\text{Unsure cells by both}). \quad (4.5)$$

Here, "CS-sel CTCs" refers to CTCs originally selected and annotated by the expert among the CS events. "CS non-sel CTCs" are CTCs that were not initially labeled among the CS events, but were subsequently labeled as CTCs in the model-based gallery. "Unsure cells by both" denotes cells identified as unsure by both systems, which are therefore excluded from the number of actual CTCs.

The equation for the actual number of CTCs by the model-based approach is analogous: it comprises the sum of CTCs found by both systems and CTCs found only by the model, with all unsure cells - regardless of category - being excluded.

Furthermore, there was no criterion defined for any unsure cell identified exclusively by the CS approach, as the expert could only annotate CTCs within the CS system.

4.4.3. Results

The following section reports the results of the extended CTC detection performance analysis comparing the final HiL model and the CS system in both patients and healthy donors. For both experiment scenarios, the detection of CTCs and unsure cells by each

system, as well as their overlap and exclusivity, are systematically evaluated. The results for scenario (1) are summarized in Tables 4.4 to 4.6, while those for scenario (2) are presented in Tables 4.7 to 4.9.

For each patient, the tables display the number of CS events and the corresponding number of actual CTCs. For the model-based approach, they report either the number of suggested CTC candidates (highlighted in pastel green) for scenario (1), or the number of model events (highlighted in pastel lilac) for scenario (2), as well as the number of actual CTCs. For each detected CTC (excluding unsure cells), the tables indicate whether it was identified by both systems, only by the model, or only by CS (i.e., without a corresponding StarDist segmentation mask). Finally, the two categories of unsure cells ("unsure cells (in general)" and "particle-like unsure cells") are initially grouped into a single category, indicating whether these were detected by both systems or only by the model. This combined group will be further subdivided and analyzed in detail in a later section.

Comparison of CS event numbers and suggested CTC candidates: Across all 73 patients, the number of CTC candidates suggested by the model in scenario (1) was significantly lower (range 0-839, average 19) than the number of CS events requiring expert image review (range 16-1,230, average 168).

Patients without CTC detection: In scenario (1), no clearly identifiable CTCs were found in either system for 45 patients. In these cases, the average number of suggested CTC candidates was approximately 5 (range 0-20), while the CS events averaged 131 (range 16-1,230). The highest number of CS events (1,230) was observed for patient 40 (see Table 4.5), for whom the model-based approach suggested only 6 CTC candidates. Notably, for these 45 patients, there were no unsure cells that were identified by both systems.

In scenario (2), the number of patients without any detected CTCs decreased slightly to 44. This discrepancy is due to patient 67 (see Table 4.9), for whom one additional image within the model events was labeled as a CTC. This cell was not among the suggested CTC candidates in scenario (1), but rather was part of the non-selected CS events and was likely overlooked by the expert in the CS system.

	CS		Model		CTCs found			Unsure cells	
Patient	Events	Actual CTCs	Suggested CTC candidates	Actual CTCs	Both	Only model	Only CS	Both	Only model
1	32	0	0	0	0	0	0	0	0
2	76	6	10	5	5	0	1	0	0
3	69	0	9	0	0	0	0	0	0
4	42	0	4	0	0	0	0	0	0
5	69	1	1	1	1	0	0	0	0
6	158	0	1	0	0	0	0	0	0
7	86	0	1	0	0	0	0	0	0
8	122	0	0	0	0	0	0	0	0
9	127	0	3	0	0	0	0	0	0
10	92	1	6	1	1	0	0	0	1
11	171	1	13	0	0	0	0	3	0
12	187	2	9	0	0	0	0	0	0
13	110	0	1	0	0	0	0	0	0
14	704	2	10	0	0	0	0	3	0
15	236	0	14	0	0	0	0	0	0
16	26	0	2	1	0	1	0	0	0
17	95	0	2	0	0	0	0	0	0
18	233	0	8	0	0	0	0	0	0
19	355	1	3	0	0	0	0	0	0
20	140	0	8	0	0	0	0	0	0
21	143	5	21	1	1	0	0	6	0
22	94	3	6	0	0	0	0	0	0
23	118	1	8	2	1	1	0	0	1
24	92	1	17	0	0	0	0	1	0
25	174	4	11	6	4	2	0	0	2
26	163	7	16	7	7	0	0	2	1
27	91	0	1	0	0	0	0	0	0
28	923	654	839	643	545	98	20	32	111
29	75	6	33	11	5	6	1	6	9
30	91	0	20	0	0	0	0	0	0

Table 4.4.: Summary of CTC detection and agreement between CS and the final HiL model in experiment scenario (1) for patients 1-30. “Events” are gallery images in CS. “Suggested CTC candidates” by the model are highlighted in pastel green. “Both” indicates detections by both systems; “Only model” are cells unique to the model; “Only CS” are cells found only by CS (lacking a StarDist segmentation mask). Abbreviations and explanations: CTC: Circulating tumor cell; CS: CellSearch®.

	CS		Model		CTCs found			Unsure cells	
Patient	Events	Actual CTCs	Suggested CTC candidates	Actual CTCs	Both	Only model	Only CS	Both	Only model
31	171	0	7	0	0	0	0	0	0
32	100	0	1	0	0	0	0	0	0
33	128	0	6	0	0	0	0	0	0
34	82	0	5	0	0	0	0	0	0
35	105	1	4	1	1	0	0	0	0
36	153	1	3	1	1	0	0	1	0
37	371	30	92	15	15	0	1	24	32
38	172	1	1	0	0	0	0	1	0
39	85	0	4	0	0	0	0	0	0
40	1230	0	6	0	0	0	0	0	1
41	452	4	10	4	4	0	0	0	1
42	198	0	13	0	0	0	0	0	1
43	282	2	2	2	2	0	0	0	0
44	114	0	11	0	0	0	0	0	0
45	161	4	7	3	3	0	0	0	0
46	105	0	2	0	0	0	0	0	0
47	111	0	15	0	0	0	0	0	0
48	94	0	7	0	0	0	0	0	0
49	347	1	3	1	1	0	0	0	0
50	169	2	6	0	0	0	0	1	1
51	25	0	17	0	0	0	0	0	0
52	58	1	1	0	0	0	0	0	0
53	91	0	5	0	0	0	0	0	0
54	158	0	3	0	0	0	0	0	0
55	41	0	1	0	0	0	0	0	0
56	99	0	2	0	0	0	0	0	0
57	100	0	3	0	0	0	0	0	0
58	177	0	4	0	0	0	0	0	0
59	66	0	5	0	0	0	0	0	0
60	180	0	9	0	0	0	0	0	0

Table 4.5.: Summary of CTC detection and agreement between CS and the final HiL model in experiment scenario (1) for patients 31-60. “Events” are gallery images in CS. “Suggested CTC candidates” by the model are highlighted in pastel green. “Both” indicates detections by both systems; “Only model” are cells unique to the model; “Only CS” are cells found only by CS (lacking a StarDist segmentation mask). Abbreviations and explanations: CTC: Circulating tumor cell; CS: CellSearch®.

	CS		Model		CTCs found			Unsure cells	
Patient	Events	Actual CTCs	Suggested CTC candidates	Actual CTCs	Both	Only model	Only CS	Both	Only model
61	100	0	0	0	0	0	0	0	0
62	57	0	1	0	0	0	0	0	0
63	62	0	2	0	0	0	0	0	0
64	145	2	5	0	0	0	1	1	0
65	494	30	24	10	6	4	0	1	0
66	78	0	1	0	0	0	0	0	0
67	65	0	4	0	0	0	0	0	0
68	69	0	0	0	0	0	0	0	0
69	193	0	14	0	0	0	0	0	2
70	60	0	8	0	0	0	0	0	2
71	16	0	0	0	0	0	0	0	0
72	30	0	0	0	0	0	0	0	0
73	179	0	4	0	0	0	0	0	0

Table 4.6.: Summary of CTC detection and agreement between CS and the final HiL model in scenario (1) for patients 61-73. "Events" are gallery images in CS. "Suggested CTC candidates" by the model are highlighted in pastel green. "Both" indicates detections by both systems; "Only model" are cells unique to the model; "Only CS" are cells found only by CS (lacking a StarDist segmentation mask). Abbreviations and explanations: CTC: Circulating tumor cell; CS: CellSearch®.

CTC agreement between systems: For the comparison of CTC agreement, it is informative to define groups of patients based on the number of detected CTCs. This allows assessment of whether patients remain within the same group in scenario (1) or are reclassified in scenario (2).

1 CTC agreement: In scenario (1), complete concordance was observed between the model-based approach and the CS system for five patients (patients 5, 10, 35, 36, and 49): in each case, exactly one CTC was identified by both systems and labeled as a CTC by the expert. For patient 23, one CTC was detected by both systems, although the model-based approach additionally identified another unique CTC not present among the CS events. In patient 21, one CTC was found by both systems, but four additional CTCs were detected by CS alone. In scenario (2), a CTC that had been missed by the model-based approach in patient 52 was now detected, resulting in agreement on one CTC for this patient as well. For patient 36, a second CTC was detected by both systems, leading to agreement on two CTCs. For patient 23, an additional unique CTC was captured solely by the model-based approach. For patient 21, although the previously missed CTCs were now present among the model events, these were classified as unsure and consequently excluded from the

actual number of CTCs in both systems, resulting in complete agreement on one CTC for this patient.

2 CTCs agreement: Patient 43 was the only patient in scenario (1) for whom complete agreement of both systems with respect to two CTCs was observed. In scenario (2), patient 36 was added to this group, as described previously.

Agreement on 2 < CTCs <5: In both scenarios, complete overlap of CTCs was observed for patient 41, with four CTCs identified by both systems. For patient 25, an agreement on four CTCs was also seen; however, in both scenarios, two additional CTCs were detected by the model-based approach compared to CS. Of particular interest is patient 45: in scenario (1), the same three CTCs were identified by both systems, but one more CTC was detected by CS. In scenario (2), the previously missed CTC in patient 45 was included among the model events, but was classified as an unsure cell, and thus was not counted as an actual CTC in either system.

Agreement ≥ 5 CTCs: Focusing on cases where five or more CTCs were consistently annotated as such by the expert and detected by both systems, agreement was observed in both scenarios for six patients (2, 26, 28, 29, 37, and 65). For patient 2, one additional CTC was detected by CS compared to the model-based approach; this discrepancy was attributable to the absence of a segmentation mask for the missing CTC, which prevented its detection by the model-based approach in both scenarios. Patients 29 and 37 represent additional cases in which one CTC detected by CS did not have a corresponding segmentation mask. In patients 28, 29, 37, and 65, it was observed that, for most cases, a larger number of CTCs was detected in scenario (2); at the same time, the number of unsure cells also increased.

Patients with higher CTC detection by the model-based approach: Instances in which the model-based approach identified more CTCs than the CS system were observed in patients 16, 23, 25, and 29 across both scenarios. Since some of these patients overlap with those mentioned in the section on CTC agreement between both systems, a similar grouping has been applied here and the comparison has been kept short.

Patients with only 1 CTC: For patient 16, the model-based method suggested two CTC candidates in scenario (1), of which one was confirmed as an actual CTC that was not detected by CS; this result remained unchanged in scenario (2).

Patients with 2 CTCs: In patient 23, two CTCs were detected by the model-based approach in scenario (1), with only one overlapping with CS. In scenario (2), an additional unique CTC not found by CS was identified by the model.

	CS		Model		CTCs found			Unsure cells	
Patient	Events	Actual CTCs	Events	Actual CTCs	Both	Only model	Only CS	Both	Only model
1	32	0	32	0	0	0	0	0	1
2	76	6	76	5	5	0	1	0	3
3	69	0	69	0	0	0	0	0	2
4	42	0	42	0	0	0	0	0	0
5	69	1	69	1	1	0	0	0	0
6	158	0	158	0	0	0	0	0	0
7	86	0	86	0	0	0	0	0	0
8	122	0	122	0	0	0	0	1	0
9	127	0	127	0	0	0	0	0	0
10	92	1	92	1	1	0	0	0	1
11	171	0	171	0	0	0	0	4	0
12	187	1	187	0	0	0	0	1	1
13	110	0	110	0	0	0	0	0	0
14	704	0	704	0	0	0	0	5	5
15	236	0	236	0	0	0	0	0	0
16	26	0	26	1	0	1	0	0	0
17	95	0	95	0	0	0	0	0	0
18	233	0	233	0	0	0	0	0	0
19	355	1	355	0	0	0	0	0	1
20	140	0	140	0	0	0	0	0	1
21	143	1	143	1	1	0	0	10	0
22	94	1	94	0	0	0	0	2	0
23	118	1	118	3	1	2	0	0	1
24	92	1	92	0	0	0	0	1	1
25	174	4	174	6	4	2	0	0	2
26	163	7	163	7	7	0	0	2	3
27	91	0	91	0	0	0	0	0	0
28	923	649	923	659	557	102	20	40	154
29	75	6	75	14	5	9	1	6	17
30	91	0	91	0	0	0	0	0	0

Table 4.7.: Comparison of CTC detection and concordance between CS and the HiL model for patients 1-30 in scenario (2). The column “Events” of the model is highlighted in pastel lilac. “Both” indicates CTCs or unsure cells identified by both systems; “Only model” refers to detections unique to the model (not present among CS events); “Only CS” refers to cells detected exclusively by CS (without available StarDist segmentation). Abbreviations and explanations: CTC: Circulating tumor cell; CS: CellSearch®.

	CS		Model		CTCs found			Unsure cells	
Patient	Events	Actual CTCs	Events	Actual CTCs	Both	Only model	Only CS	Both	Only model
31	171	0	171	0	0	0	0	0	0
32	100	0	100	0	0	0	0	0	0
33	128	0	128	0	0	0	0	0	0
34	82	0	82	0	0	0	0	0	0
35	105	1	105	1	1	0	0	0	0
36	153	2	153	2	2	0	0	1	0
37	371	21	371	15	15	0	1	33	114
38	172	1	172	0	0	0	0	1	1
39	85	0	85	0	0	0	0	0	0
40	1230	0	1230	0	0	0	0	0	2
41	452	4	452	4	4	0	0	0	2
42	198	0	198	0	0	0	0	0	1
43	282	2	282	2	2	0	0	0	0
44	114	0	114	0	0	0	0	0	0
45	161	3	161	3	3	0	0	1	1
46	105	0	105	0	0	0	0	0	0
47	111	0	111	0	0	0	0	0	0
48	94	0	94	0	0	0	0	0	0
49	347	1	347	1	1	0	0	0	0
50	169	1	169	0	0	0	0	2	2
51	25	0	25	0	0	0	0	0	0
52	58	1	58	1	1	0	0	0	0
53	91	0	91	0	0	0	0	0	0
54	158	0	158	0	0	0	0	0	0
55	41	0	41	0	0	0	0	0	0
56	99	0	99	0	0	0	0	0	0
57	100	0	100	0	0	0	0	0	0
58	177	0	177	0	0	0	0	0	2
59	66	0	66	0	0	0	0	0	0
60	180	0	180	0	0	0	0	0	0

Table 4.8.: Comparison of CTC detection and concordance between CS and the HiL model for patients 31-60 in scenario (2). The column “Events” of the model is highlighted in pastel lilac. “Both” indicates CTCs or unsure cells identified by both systems; “Only model” refers to detections unique to the model (not present among CS events); “Only CS” refers to cells detected exclusively by CS (without available StarDist segmentation). Abbreviations and explanations: CTC: Circulating tumor cell; CS: CellSearch®.

	CS		Model		CTCs found			Unsure cells	
Patient	Events	Actual CTCs	Events	Actual CTCs	Both	Only model	Only CS	Both	Only model
61	100	0	100	0	0	0	0	0	1
62	57	0	57	0	0	0	0	0	0
63	62	0	62	0	0	0	0	0	0
64	145	1	145	0	0	0	1	2	1
65	494	20	494	19	13	6	0	14	46
66	78	0	78	0	0	0	0	0	1
67	65	1	65	1	1	0	0	0	0
68	69	0	69	0	0	0	0	0	0
69	193	0	193	0	0	0	0	0	2
70	60	0	60	0	0	0	0	0	2
71	16	0	16	0	0	0	0	0	0
72	30	0	30	0	0	0	0	0	0
73	179	0	179	0	0	0	0	0	0

Table 4.9.: Comparison of CTC detection and concordance between CS and the HiL model for patients 61-73 in scenario (2). The column "Events" of the model is highlighted in pastel lilac. "Both" indicates CTCs or unsure cells identified by both systems; "Only model" refers to detections unique to the model (not present among CS events); "Only CS" refers to cells detected exclusively by CS (without available StarDist segmentation). Abbreviations and explanations: CTC: Circulating tumor cell; CS: CellSearch®.

Patients with ≥ 5 CTCs: In scenario (1), two patients (25 and 29) were found to have more than five CTCs identified by the model-based approach, while for patient 25, fewer than five CTCs were detected by CS. For patient 29, the number of additional CTCs uniquely identified by the model increased by three. In scneario (2), an additional patient (patient 28) had more actual CTCs detected by the model-based approach than by CS. This result is primarily attributable to an increased number of unsure cells detected by both systems. Notably, patient 28 exhibited the highest number of additional CTCs identified by the model-based approach that were not found by CS. Representative examples are shown in the first three rows of Fig. 4.17. These CTCs display a strong CK signal, which leads to a shine-through effect in the CD45 channel, while the DAPI signal is comparatively weaker. Such features were commonly observed in other patients where additional CTCs were detected by the model-based approach but not by CS.

Patients with higher CTC detection by CS: In scenario (1), CS detected a greater number of CTCs than the model-based approach in 16 patients, with differences ranging from 1 to 15 additional CTCs. In scenario (2), this number decreased to 10 patients.

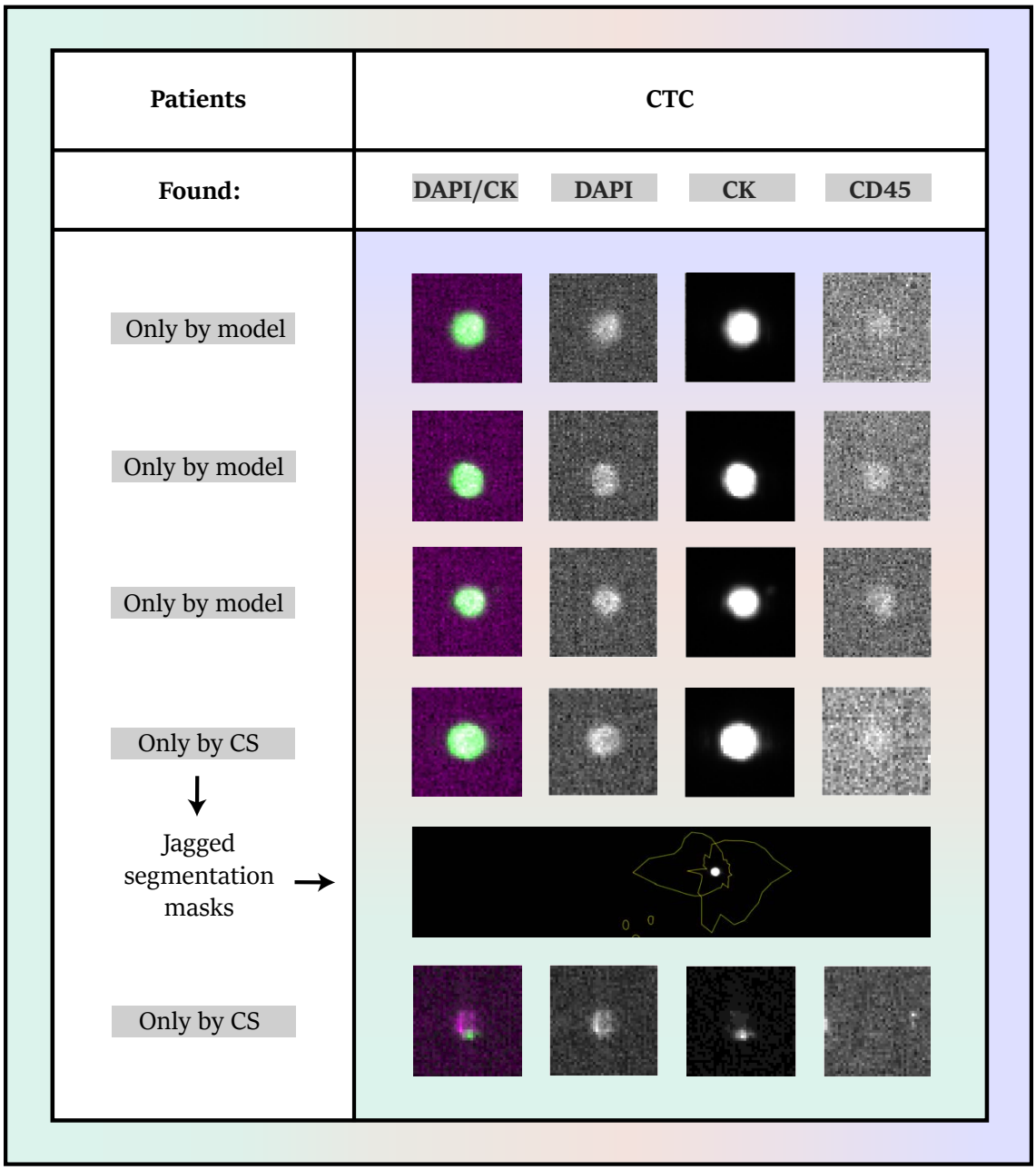


Figure 4.17.: CTCs uniquely detected in one system but not by the other. Three example CTCs are shown that were uniquely found by the model-based approach, and two CTCs are depicted that were only detected by the CS system. For one of the CTCs detected only by CS, jagged segmentation masks were generated by StarDist. Abbreviations and explanations: CTC: Circulating tumor cell; DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45: Exclusion marker; CS: CellSearch®.

This reduction is partly attributable to the identification of additional CTCs, and partly to an increased number of unsure cells detected by both systems. Among these 10 patients, there were still 7 cases (patients 12, 19, 22, 24, 38, 50, and 64) where the model-based approach failed to detect any actual CTCs, while CS identified at least one. Additionally,

there were instances where one or more StarDist segmentation masks for CTCs were missing.

Rarely did CS-unique CTCs display features similar to those seen in the first three cells shown in Fig. 4.17. One such example is the fourth CTC - corresponding to the missed CTC in patient 2 - in Fig. 4.17, where, despite StarDist segmentation, the resulting jagged masks led to offset bounding boxes and consequently prevented the cell from being captured by the model-based approach. In most cases, however, CTCs lacking a StarDist segmentation mask exhibited moderate to very weak CK signal intensities, as shown by the last CTC in Fig. 4.17.

Unsure cells: In scenario (1), an average of 6 unsure cells (range 1-32) detected by both systems were found in 13 patients. In a different set of 13 patients (not fully overlapping with the previous group), the model-based approach identified an average of 13 additional, uniquely detected unsure cells (range: 1-111).

As more images were required to be reviewed, the number of unsure cells increased in scenario (2): 17 patients had an average of 7 unsure cells (range 1-40) detected by both systems, while in 28 patients, the model-based approach identified an average of 13 additional unsure cells (range 1-154). This increase in unsure cells was also reflected in the actual CTC counts for both systems, as these cells were excluded from the reported totals. These findings motivated a more detailed analysis of unsure cells in scenario (2). Additionally, the two categories of unsure cells had previously been grouped together; in Table 4.10, they are distinguished as "Unsure cells (in general)" and "Particle-like, unsure cells."

For the first category, across all patients, 16 patients had unsure cells that were detected by both systems (range 1-10, average 3), while unsure cells uniquely found by the model-based approach appeared in 21 patients (range 1-5, average 1). Example cells of this category are shown in Fig. 4.18. The first two cells (detected by both systems) exhibit overlapping DAPI and CK signals, but have a very weak CD45 signal, which complicates interpretation. Additionally, the CK signal in these cells is not as strong as in the first four cells shown in Fig. 4.17, making it unlikely that the effect observed in CD45 is due to shine-through.

The last two cells in Fig. 4.18 represent unsure cells detected only by the model-based approach. The third cell demonstrates a clear DAPI signal and a CK signal that encircles the DAPI region without complete overlap. Since there is no CD45 signal, this cell could potentially be a CTC, but due to the uncertainty, it was classified as unsure. The last cell shows a CK signal that overlaps with DAPI but is quite weak, making interpretation difficult.

	Unsure cells (in general)		Particle-like, unsure cells			Unsure cells (in general)		Particle-like, unsure cells	
Patient	Both	Only model	Both	Only model	Patient	Both	Only model	Both	Only model
1	0	1	0	0	38	1	0	0	1
2	0	0	0	3	39	0	0	0	0
3	0	2	0	0	40	0	1	0	1
4	0	0	0	0	41	0	0	0	2
5	0	0	0	0	42	0	1	0	0
6	0	0	0	0	43	0	0	0	0
7	0	0	0	0	44	0	0	0	0
8	1	0	0	0	45	1	1	0	0
9	0	0	0	0	46	0	0	0	0
10	0	1	0	0	47	0	0	0	0
11	4	0	0	0	48	0	0	0	0
12	1	1	0	0	49	0	0	0	0
13	0	0	0	0	50	2	2	0	0
14	5	5	0	0	51	0	0	0	0
15	0	0	0	0	52	0	0	0	0
16	0	0	0	0	53	0	0	0	0
17	0	0	0	0	54	0	0	0	0
18	0	0	0	0	55	0	0	0	0
19	0	1	0	0	56	0	0	0	0
20	0	1	0	0	57	0	0	0	0
21	10	0	0	0	58	0	1	0	1
22	2	0	0	0	59	0	0	0	0
23	0	0	0	1	60	0	0	0	0
24	1	0	0	1	61	0	1	0	0
25	0	2	0	0	62	0	0	0	0
26	0	1	2	2	63	0	0	0	0
27	0	0	0	0	64	2	1	0	0
28	10	1	30	153	65	2	0	12	46
29	3	0	3	17	66	0	1	0	0
30	0	0	0	0	67	0	0	0	0
31	0	0	0	0	68	0	0	0	0
32	0	0	0	0	69	0	2	0	0
33	0	0	0	0	70	0	2	0	0
34	0	0	0	0	71	0	0	0	0
35	0	0	0	0	72	0	0	0	0
36	1	0	0	0	73	0	0	0	0
37	1	1	32	113					

Table 4.10.: Unsure cells and particle-like cells detected in both systems (“Both”) or only by the model (“Only model”) for all patients (1-73).

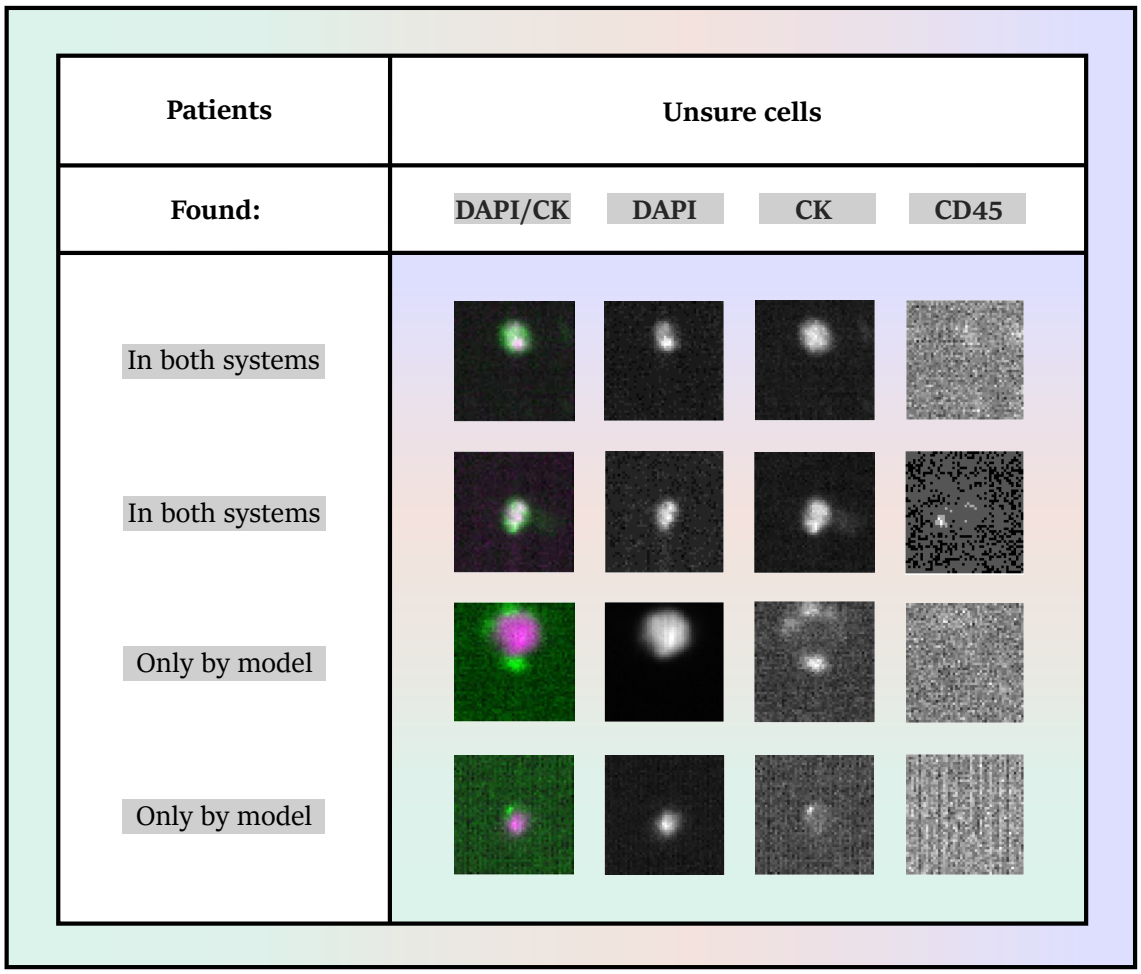


Figure 4.18.: Unsure cells (in general) in patient samples. The first two cells are unsure cells detected by both systems, while the last two were identified only by the model-based approach and missed by CS. Abbreviations and explanations: CTC: Circulating tumor cell; DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45: Exclusion marker; CS: CellSearch®.

In contrast to the generally unsure cells, particle-like unsure cells were detected much less frequently (in 5 patients by both systems and in 12 patients by the model only). However, when present within a patient, particle-like, unsure cells tended to occur more frequently: if detected by both systems, the range was 2-32 (average 16), and if detected only by the model, the range was 1-153 (average 28). Exemplary cells can be seen in Fig. 4.19.

Among both the cells detected by both systems (first two cells) and those identified only by the model-based approach (last two cells), the majority were small cells exhibiting a strong CK signal and a weak, even smaller DAPI signal. Notably, the last cell displayed an almost undetectable nucleus but a very strong CK signal.

Such particle-like, unsure cells were found only in patient samples and not in healthy donors. Although healthy donors did not display any clear CTCs - neither in CS nor in the model-based approach - some cells were marked as generally, unsure cells. In

experiment scenario (1), in three individuals, unsure cells detected by both systems were found (range 1-2); no unsure cells were uniquely detected by the model-based approach. In scenario (2), the number of unsure cells increased slightly.

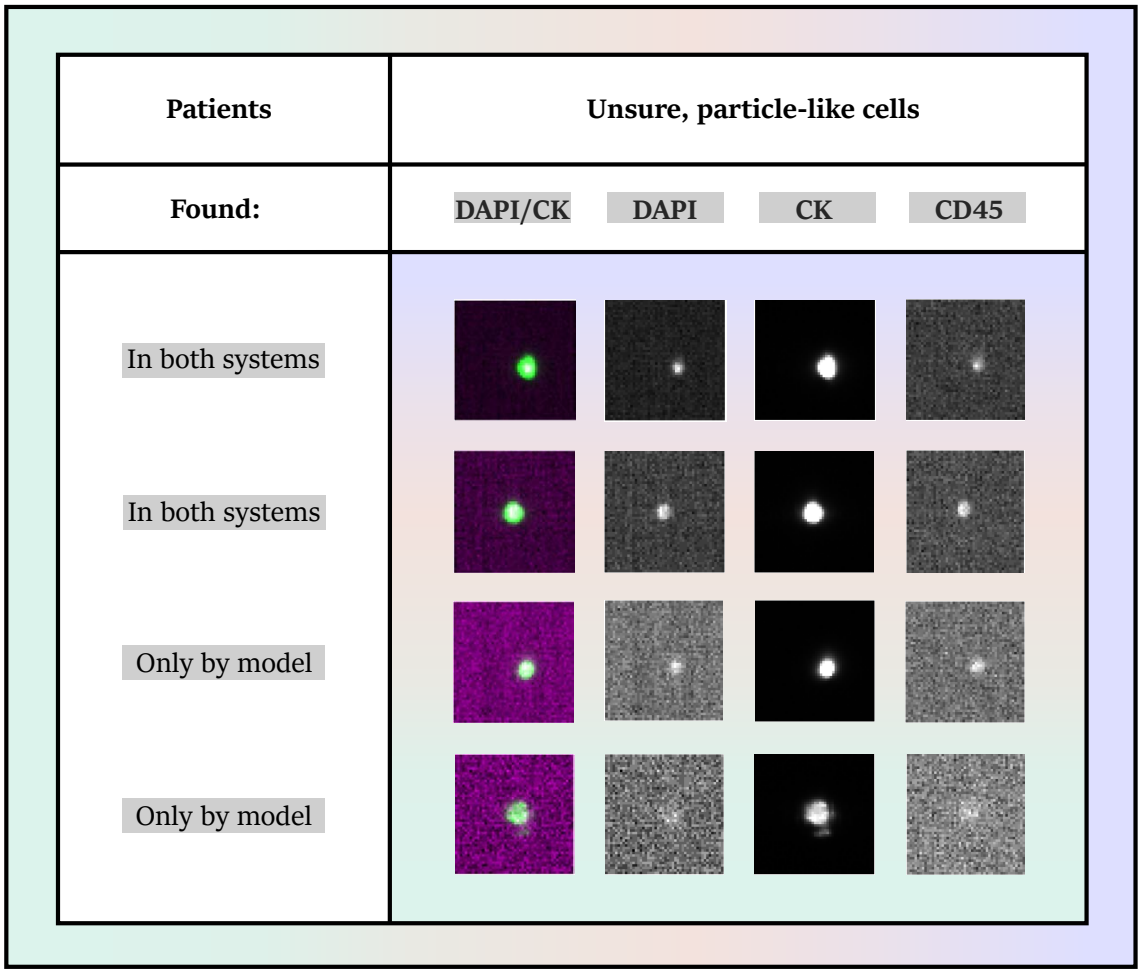


Figure 4.19.: Particle-like unsure cells in patient samples. The first two cells are unsure cells detected by both systems, while the last two were identified only by the model-based approach but missed by CS. Abbreviations and explanations: CTC: Circulating tumor cell; DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45: Exclusion marker; CS: CellSearch®.

Across four donors, 1-2 unsure cells detected by both systems were reported. An example is shown in the first row of Fig. 4.20, displaying overlapping DAPI and CK signals but lacking a CD45 signal.

Additionally, in two individuals, 1-2 unsure cells per donor were found uniquely by the model-based approach. These cells are represented by cells 2-4 in Fig. 4.20; all exhibit overlapping DAPI and CK signals and are CD45 negative, with cells 3 and 4 displaying a relatively weak CK signal.

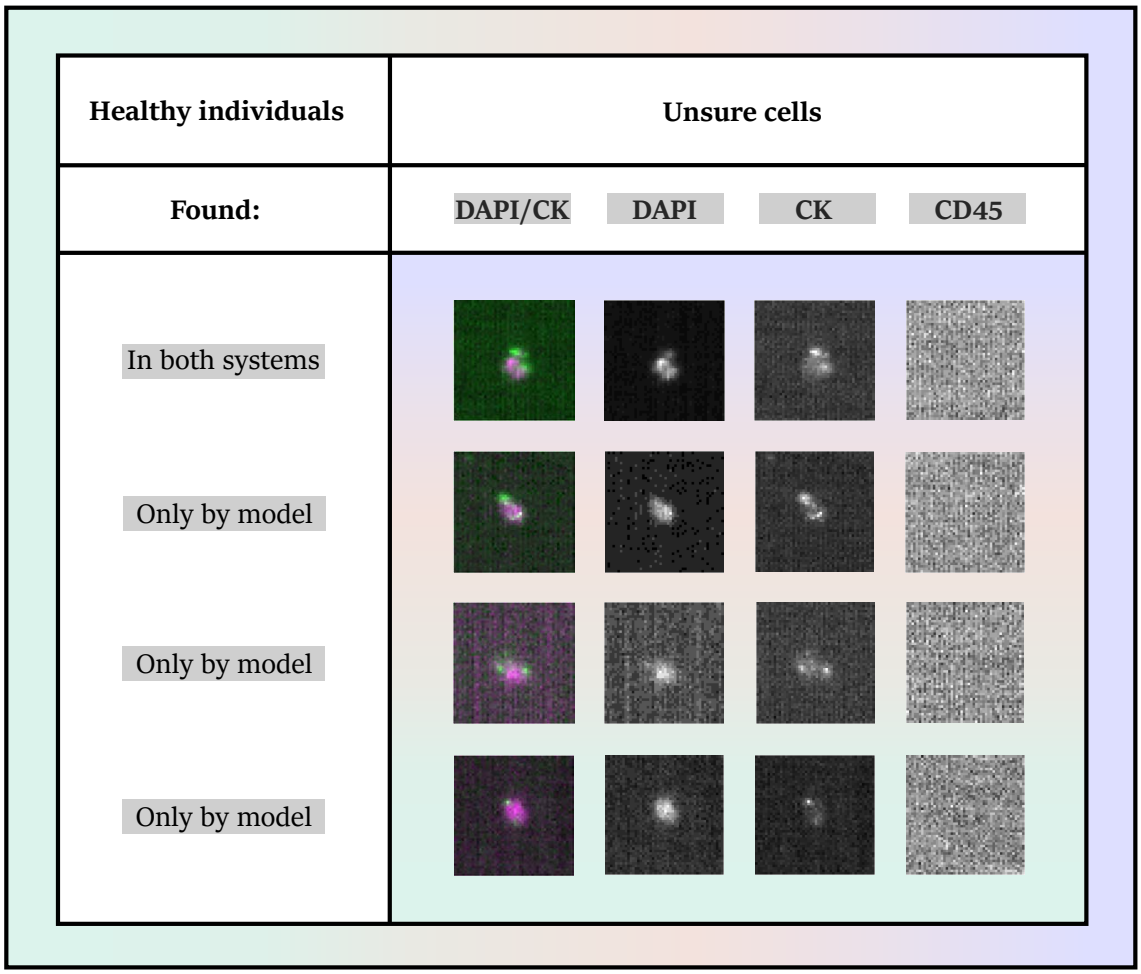


Figure 4.20.: Unsure cells (in general) in healthy individuals. The first cell is an unsure cell detected by both systems, while the remaining cells were identified only by the model-based approach and missed by CS. Abbreviations and explanations: CTC: Circulating tumor cell; DAPI: 4',6-diamidino-2-phenylindole, dihydrochloride; CK: Cytokeratin; CD45: Exclusion marker; CS: CellSearch®.

4.4.4. Discussion

This study evaluated whether the final HiL model - previously trained on metastatic patient data with high CTC counts - maintains performance comparable to CS in a more challenging clinical setting, where most patients exhibit CTC levels below the prognostic threshold. The analysis also extended to healthy donors, assessing model sensitivity in a setting not addressed in earlier work.

In scenario (1), the use of a fixed confidence threshold led the model to suggest substantially fewer cells for expert review than CS - confirming the reduced workload observed in the high-CTC cohort - but, consistent with previous findings, some CTCs detected by CS were missed by the model.

Therefore, a second scenario was defined in which the number of reviewed model events was increased to match the number of CS events per patient, enabling the potential recovery of previously missed CTCs and the discovery of additional ones. Scenario (2) yielded a moderate increase in detected CTCs (range 1-16, average 5) in seven patients, and in about five cases, the model identified more CTCs than CS. In one of these five patients, a CTC was found by the model while CS did not detect any. However, for ten patients where CS found at least one CTC and the model none in scenario (1), scenario (2) also failed to recover these cells - mainly due to the pre-processing pipeline (removal of noisy and duplicate images) rather than segmentation errors. This may be attributable to the removal of noisy images in which a CTC might have been mistakenly filtered out, or during the step where duplicates were removed because of overlapping bounding boxes.

Moreover, while no additional CTCs were identified for these ten patients in scenario (2), an increase in unsure cells - both those detected by both systems and those unique to the model - was observed. Some of these ambiguous cases, such as cells characterized by strong CK signal, CD45 shine-through, and variable DAPI expression, had also been noted in the previous HiL cohort, where they were often classified as CTCs due to a greater emphasis on sensitivity and the maximization of CTC detection. In the present study, however, the labeling strategy was adjusted to explicitly account for uncertain cells.

As uncertainty was frequently encountered by the expert during image review, it cannot be expected that the model will achieve confident classifications in these instances. This highlights the necessity for uncertainty to be explicitly incorporated into both labeling and model training. While the introduction of a dedicated unsure class - distinct from clear CTCs and non-CTCs - could be considered, this approach is not straightforward. The findings of this study demonstrated that the "unsure" category itself is heterogeneous, encompassing at least two subtypes: generally unsure, morphologically heterogeneous cells (for example, those with weak CD45 signal at areas of DAPI and CK overlap; various CK signal patterns; apoptotic or deformed cells), and particle-like cells (DAPI-negative or weak DAPI-positive, CK-positive, CD45-negative). Notably, cells of the latter category were not detected in any of the healthy donors; however, their definition was motivated by reports on tdEVs (DAPI-/CK+/CD45-) in certain cancer patients [Cie+25], which are known to be co-enriched during EpCAM-based CS enrichment. Notably, in the peripheral circulation of patients with metastatic colorectal cancer, CS-tdEVs have been found to be more abundant than CTCs, with their levels serving as an independent risk factor for shorter overall survival [Cie+25; Nan+20b; Nan+20a]. A similar trend was observed in several patients within the current cohort of metastatic breast cancer patients, where sometimes more particle-like unsure cells were identified than actual CTCs. However,

this finding should be interpreted with caution, as uncertainties in labeling remained for this category. Despite efforts to standardize annotation of unsure cells, complete consistency in the assignment of unsure labels was challenging.

Limitations of the study

A primary limitation of this study concerns the loss of CTCs during the pre-processing pipeline. While there were occasional cases where segmentation masks were not available, in most instances segmentation masks did exist for the missing CTCs (that were detected by CS). As these cells were not reviewed, it remains unclear whether these missing cells were located in regions of densely packed cells or at the edges of cartridge frames. Therefore, a systematic investigation is needed to clarify the precise reasons for their exclusion. Furthermore, it must be determined whether these lost cells represent clearly identifiable CTCs. Should this be the case, optimization of the pre-processing steps - such as noise and duplicate removal - will be essential to minimize the loss of relevant cells.

Another challenge relates to the assignment of cells located at frame boundaries. In some cases, a cell at the edge of a cartridge frame in CS was not registered to the same frame in the model-based approach, but instead to an adjacent frame. Since cell comparisons were made within the same cartridge frame, this may have led to instances where a CTC appeared to be “missing” or “additional,” when in fact it was simply assigned to a different frame in the two systems. This issue is particularly critical in samples with low CTC counts, where each individual cell carries greater significance. Therefore, more refined matching procedures - such as cross-frame analysis - need to be considered to ensure accurate correspondence of boundary cells between methods.

Finally, expert image review of cell morphology was limited by the CS scan’s magnification, restricting detailed evaluation. Definitive identification of CTCs requires molecular analysis to confirm genetic aberrations. If molecular information and corresponding fluorescent images are available for unsure cells, it would be possible to investigate whether specific image features are associated with CTC identity.

Chapter 5

Conclusion and outlook

The aim of this thesis was to develop, apply, and further evaluate image analysis, ML, and DL strategies for the automated and robust detection and classification of CTCs in fluorescence microscopy images obtained from liquid biopsy samples. CTCs, a clinically important yet rare biomarker in the blood of cancer patients, were analyzed in a metastatic breast cancer cohort. Since all image data were acquired with the CS system - the current gold standard for CTC detection, including metastatic breast cancer - a key focus of this work was the direct comparison of AI-based detection performance against CS. The analysis was based on three fluorescence channels: DAPI (nuclear marker), CK (tumor marker), and CD45 (exclusion marker).

For robust CTC detection, which is the basis for all further classification and analysis steps, the DL-based segmentation algorithm StarDist was thoroughly evaluated. StarDist offered particular advantages over other methods due to its star-convex polygon shape representation, which is well suited to the circular signal morphology in these microscopy images, and its ability, in most cases, to accurately segment individual cells even in densely packed or overlapping regions.

For automated CTC versus non-CTC differentiation of StarDist-segmented cells, this work demonstrated that SSL - where model training is based solely on unlabeled data - is also applicable to the LB field and that, when combined with a simple ML classifier for downstream tasks, enabled accurate CTC classification and outperformed supervised DL approaches, especially when labeled data were limited. Additionally, the SSL approach also allowed for the exploration of cell class clusters in the latent space.

Although full automatization is desirable, classifier uncertainties for certain cases may persist and had previously remained unaddressed. Therefore, in an extended study using a larger metastatic patient cohort, areas of model uncertainty became a focal point and were identified through cluster-based latent space analysis. To efficiently reduce these uncertainties, a targeted human-in-the-loop (HiL) method was introduced: additional training samples were automatically selected from uncertain regions and presented to experts for labeling, allowing this expert input to iteratively improve classifier performance.

When the model trained with the HiL method and a confidence threshold was tested on metastatic breast cancer patients with both high CTC counts (progressive disease,

all above the 5 CTC cut-off) and lower CTC burden (e.g., stable disease, responding to therapy), it suggested significantly fewer candidates than CS while showing higher positive predicitve values. However, in both cases, some cells were missed by either system. Increasing the number of reviewed model candidates to match CS led to a rise in detected CTCs for some patients, but also resulted in a notable increase in the number of unsure cells during expert image review in several patients. Even in healthy donors, a small number of uncertain cells were identified. Thus, in addition to model uncertainties that can be addressed with expert guidance in the HiL approach, expert uncertainties itself also remain. These, and potentially new forms of uncertainty, are likely to arise especially when extending the approach to additional patient cohorts or other tumor types. Future work should therefore focus on developing reliable strategies to communicate such labeling uncertainty to the model during training.

In summary, this thesis successfully developed, implemented, and evaluated AI-based methods for automated CTC detection and classification in liquid biopsy data. Strategies to address model uncertainties were presented, and additional categories of unsure cells among CTCs and non-CTCs were systematically investigated.

Bibliography

[Abd+21] T. S. Abdalla, J. Meiners, S. Riethdorf, et al. “Prognostic value of preoperative circulating tumor cells counts in patients with UICC stage I-IV colorectal cancer”. In: *PLOS One* 16.6 (2021), e0252897.

[Ace+14] N. Aceto, A. Bardia, D. T. Miyamoto, et al. “Circulating tumor cell clusters are oligoclonal precursors of breast cancer metastasis”. In: *Cell* 158.5 (2014), pp. 1110–1122.

[Agg+23] S. Aggarwal, A. K. Sahoo, C. Bansal, et al. “Image classification using deep learning: A comparative study of vgg-16, inceptionv3 and efficientnet b7 models”. In: *2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering*. IEEE. 2023, pp. 1728–1732.

[Ahn+23] J. Ahn, J. Hong, J. Ju, et al. “Redesigning Embedding Layers for Queries, Keys, and Values in Cross-Covariance Image Transformers”. In: *Mathematics* 11.8 (2023), p. 1933.

[ALL21] E. Alexander, K. W. Leong, and A. F. Laine. “Automated multi-process CTC detection using deep learning”. In: *arXiv preprint arXiv:2109.12709* (2021).

[Ali+23] A. M. Ali, B. Benjdira, A. Koubaa, et al. “Vision transformers in image restoration: A survey”. In: *Sensors* 23.5 (2023), p. 2385.

[AP25] C. Alix-Panabières and K. Pantel. “Advances in liquid biopsy: From exploration to practical application”. In: *Cancer Cell* 43.2 (2025), pp. 161–165.

[AKC20] M. Allaoui, M. L. Kherfi, and A. Cheriet. “Considerably improving clustering algorithms using UMAP dimensionality reduction technique: a comparative study”. In: *International Conference on Image and Signal Processing*. Springer. 2020, pp. 317–325.

[All+04] W. J. Allard, J. Matera, M. C. Miller, et al. “Tumor cells circulate in the peripheral blood of all major carcinomas but not in healthy subjects or patients with nonmalignant diseases”. In: *Clinical Cancer Research* 10.20 (2004), pp. 6897–6904.

[All+20] M. Alloghani, D. Al-Jumeily, J. Mustafina, et al. "A systematic review on supervised and unsupervised machine learning algorithms for data science". In: *Supervised and Unsupervised Learning for Data Science* (2020), pp. 3–21.

[Alz+21] L. Alzubaidi, J. Zhang, A. J. Humaidi, et al. "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions". In: *Journal of Big Data* 8.1 (2021), p. 53.

[AVT16] K. C. Andree, G. Van Dalum, and L. W. Terstappen. "Challenges in circulating tumor cell detection by the CellSearch system". In: *Molecular Oncology* 10.3 (2016), pp. 395–407.

[Aza+24] R. Azad, E. K. Aghdam, A. Rauland, et al. "Medical image segmentation review: The success of u-net". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).

[BF23] M. Bariklana and A. Fauzan. "Implementation of the dbscan method for cluster mapping of earthquake spread location". In: *Barekeng: Jurnal Ilmu Matematika dan Terapan* 17.2 (2023), pp. 0867–0878.

[BT13] A. M. Barradas and L. W. Terstappen. "Towards the biological understanding of CTC: capture technologies, definitions and potential to create metastasis". In: *Cancers* 5.4 (2013), pp. 1619–1642.

[Beh+24] M. Behzadpour, B. L. Ortiz, E. Azizi, et al. "Breast tumor classification using efficientnet deep learning model". In: *arXiv preprint arXiv:2411.17870* (2024).

[BM21] P. Bhattacharjee and P. Mitra. "A survey of density based clustering algorithms". In: *Frontiers of Computer Science* 15.1 (2021), p. 151308.

[Bid+18] F.-C. Bidard, S. Michiels, S. Riethdorf, et al. "Circulating tumor cells in breast cancer patients treated by neoadjuvant chemotherapy: a meta-analysis". In: *JNCI: Journal of the National Cancer Institute* 110.6 (2018), pp. 560–567.

[Boo+25] E. L. Boone, R. A. Ghanam, F. S. Alamri, et al. "Metabolite Screening for Heart Disease Using Support Vector Machine-Based AI". In: *International Journal of Advanced Computer Science & Applications* 16.6 (2025).

[Bra+24] F. Bray, M. Laversanne, H. Sung, et al. "Global cancer statistics 2022: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries". In: *CA: A Cancer Journal for Clinicians* 74.3 (2024), pp. 229–263.

[Cai+19] J. C. Caicedo, J. Roth, A. Goodman, et al. "Evaluation of deep learning strategies for nucleus segmentation in fluorescence images". In: *Cytometry Part A* 95.9 (2019), pp. 952–965.

[CMS13] R. J. Campello, D. Moulavi, and J. Sander. "Density-based clustering based on hierarchical density estimates". In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer. 2013, pp. 160–172.

[Car+21] M. Caron, H. Touvron, I. Misra, et al. "Emerging properties in self-supervised vision transformers". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 9650–9660.

[Cer+20] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, et al. "A comprehensive survey on support vector machine classification: Applications, challenges and trends". In: *Neurocomputing* 408 (2020), pp. 189–215.

[Che+16] C. L. Chen, A. Mahjoubfar, L.-C. Tai, et al. "Deep learning in label-free cell classification". In: *Scientific Reports* 6.1 (2016), p. 21471.

[Che+20] T. Chen, S. Kornblith, M. Norouzi, et al. "A simple framework for contrastive learning of visual representations". In: *International Conference on Machine Learning*. PmLR. 2020, pp. 1597–1607.

[Cie+25] S. A. Cieslik, A. G. Zafra, C. Driemel, et al. "Phenotypic diversity of CTCs and tdEVs in liquid biopsies of tumour-draining veins is linked to poor prognosis in colorectal cancer". In: *Journal of Experimental & Clinical Cancer Research* 44.1 (2025), p. 9.

[Cla+14] M. Claesen, F. De Smet, J. A. Suykens, et al. "Fast prediction with SVM models containing RBF kernels". In: *arXiv preprint arXiv:1403.0736* (2014).

[Coh+08] S. J. Cohen, C. J. Punt, N. Iannotti, et al. "Relationship of circulating tumor cells to tumor response, progression-free survival, and overall survival in patients with metastatic colorectal cancer". In: *Journal of Clinical Oncology* 26.19 (2008), pp. 3213–3221.

[Cri+04] M. Cristofanilli, G. T. Budd, M. J. Ellis, et al. "Circulating tumor cells, disease progression, and survival in metastatic breast cancer". In: *New England Journal of Medicine* 351.8 (2004), pp. 781–791.

[Cui+24] J. Cui, Z. Tian, Z. Zhong, et al. "Decoupled kullback-leibler divergence loss". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 74461–74486.

[Cui+20] S. Cui, H.-H. Tseng, J. Pakela, et al. "Introduction to machine and deep learning for medical physicists". In: *Medical Physics* 47.5 (2020), e127–e147.

[Cum+13] J. Cummings, K. Morris, C. Zhou, et al. "Method validation of circulating tumour cell enumeration at low cell counts". In: *BMC Cancer* 13.1 (2013), p. 415.

[CD21] P. Cunningham and S. J. Delany. "K-nearest neighbour classifiers-a tutorial". In: *ACM Computing Surveys* 54.6 (2021), pp. 1–25.

[De +08] J. S. De Bono, H. I. Scher, R. B. Montgomery, et al. "Circulating tumor cells predict survival benefit from treatment in metastatic castration-resistant prostate cancer". In: *Clinical Cancer Research* 14.19 (2008), pp. 6302–6309.

[DKB19] G. De Rubis, S. R. Krishnan, and M. Bebawy. "Liquid biopsies in cancer diagnosis, monitoring, and prognosis". In: *Trends in Pharmacological Sciences* 40.3 (2019), pp. 172–186.

[DeV+21] Z. DeVries, E. Locke, M. Hoda, et al. "Using a national surgical database to predict complications following posterior lumbar surgery and comparing the area under the curve and F1-score for the assessment of prognostic capability". In: *The Spine Journal* 21.7 (2021), pp. 1135–1142.

[DEA24] R. Diallo, C. Edalo, and O. O. Awe. "Machine Learning Evaluation of Imbalanced Health Data: A Comparative Analysis of Balanced Accuracy, MCC, and F1 Score". In: *Practical Statistical Learning and Data Science Methods: Case Studies from LISA 2020 Global Network, USA*. Springer, 2024, pp. 283–312.

[DQZ18] B. Ding, H. Qian, and J. Zhou. "Activation functions and their characteristics in deep neural networks". In: *2018 Chinese Control and Decision Conference*. IEEE. 2018, pp. 1836–1841.

[Dos+20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, et al. "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv:2010.11929* (2020).

[Dot+24] E. Dotan, G. Jaschek, T. Pupko, et al. "Effect of tokenization on transformers for biological sequences". In: *Bioinformatics* 40.4 (2024), btae196.

[Eff+18] K. E. Effenberger, C. Schroeder, A. Hanssen, et al. "Improved risk stratification by circulating tumor cell counts in pancreatic cancer". In: *Clinical Cancer Research* 24.12 (2018), pp. 2844–2850.

[Elh+25] O. Elharrouss, Y. Mahmood, Y. Bechqito, et al. "Loss Functions in Deep Learning: A Comprehensive Review". In: *arXiv preprint arXiv:2504.04242* (2025).

[Est+96] M. Ester, H.-P. Kriegel, J. Sander, et al. "A density-based algorithm for discovering clusters in large spatial databases with noise". In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. Vol. 96. 34. 1996, pp. 226–231.

[Ete+25] M. Etelvina, P. Ward, T. Kolton, et al. "Harnessing Unlabeled Data with Self-Supervised Learning". In: *Authorea Preprints* (2025).

[Ezu+22] A. E. Ezugwu, A. M. Ikotun, O. O. Oyelade, et al. "A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects". In: *Engineering Applications of Artificial Intelligence* 110 (2022), p. 104743.

[Fle09] T. Fletcher. "Support vector machines explained". In: *Tutorial paper* 1118 (2009), pp. 1–19.

[GG20] B. Ghojogh and A. Ghodsi. "Attention Mechanism, Transformers, BERT, and GPT: Tutorial and Survey". Working paper or preprint. 2020.

[GK20] H. Gholamalinezhad and H. Khosravi. "Pooling methods in deep neural networks, a review". In: *arXiv preprint arXiv:2009.07485* (2020).

[GS14] O. Gires and N. H. Stoecklein. "Dynamic EpCAM expression on circulating and disseminating tumor cells: causes and consequences". In: *Cellular and Molecular Life Sciences* 71.22 (2014), pp. 4393–4402.

[Gri+20] J.-B. Grill, F. Strub, F. Altché, et al. "Bootstrap your own latent-a new approach to self-supervised learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 21271–21284.

[Gui+24] J. Gui, T. Chen, J. Zhang, et al. "A survey on self-supervised learning: Algorithms, applications, and future trends". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46.12 (2024), pp. 9052–9071.

[HA21] S. H. Haji and A. M. Abdulazeez. "Comparison of optimization techniques based on gradient descent algorithm: A review". In: *PalArch's Journal of Archaeology of Egypt/Egyptology* 18.4 (2021), pp. 2715–2743.

[HKS16] A. A. Hameed, B. Karlik, and M. S. Salman. "Back-propagation algorithm with variable adaptive momentum". In: *Knowledge-Based Systems* 114 (2016), pp. 79–87.

[Han+22] K. Han, Y. Wang, H. Chen, et al. "A survey on vision transformer". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.1 (2022), pp. 87–110.

[HW11] D. Hanahan and R. A. Weinberg. “Hallmarks of cancer: the next generation”. In: *Cell* 144.5 (2011), pp. 646–674.

[HNZ13] R. A. Harouaka, M. Nisic, and S.-Y. Zheng. “Circulating tumor cell enrichment based on physical properties”. In: *Journal of Laboratory Automation* 18.6 (2013), pp. 455–468.

[Hay+06] D. F. Hayes, M. Cristofanilli, G. T. Budd, et al. “Circulating tumor cells at each follow-up time point during therapy of metastatic breast cancer patients predict progression-free and overall survival”. In: *Clinical Cancer Research* 12.14 (2006), pp. 4218–4224.

[He+20] K. He, H. Fan, Y. Wu, et al. “Momentum contrast for unsupervised visual representation learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 9729–9738.

[Hot33] H. Hotelling. “Analysis of a complex of statistical variables into principal components”. In: *Journal of Educational Psychology* 24.6 (1933), p. 417.

[HR21] E. L. Hunt and S. Reffert. “Improving the open cluster census-I. Comparison of clustering algorithms applied to Gaia DR2 data”. In: *Astronomy & Astrophysics* 646 (2021), A104.

[Hus+23] H. Husseini, M. Nielsen, K. Pantel, et al. “Label Efficient Classification in Liquid Biopsy Data by Self-supervision”. In: *Bildverarbeitung für die Medizin 2023* (2023), p. 261.

[Hus+25] H. Husseini-Wüsthoff, S. Riethdorf, A. Schneeweiss, et al. “Cluster-based human-in-the-loop strategy for improving machine learning-based circulating tumor cell detection in liquid biopsy”. In: *Patterns* 6.6 (2025), p. 101285.

[Hus] H. Husseini-Wüsthoff. *CTC-HiL*. `https://github.com/IPMI-ICNS-UKE/CTC-HiL`. Accessed: August 20, 2025.

[Isl+24] S. Islam, H. Elmekki, A. Elsebai, et al. “A comprehensive survey on applications of transformers for deep learning tasks”. In: *Expert Systems with Applications* 241 (2024), p. 122666.

[Iye+20] A. Iyer, K. Gupta, S. Sharma, et al. “Integrative analysis and machine learning based characterization of single circulating tumor cells”. In: *Journal of Clinical Medicine* 9.4 (2020), p. 1206.

[JZX17] X.-X. Jie, X.-Y. Zhang, and C.-J. Xu. “Epithelial-to-mesenchymal transition, circulating tumor cells and cancer metastasis: Mechanisms and clinical applications”. In: *Oncotarget* 8.46 (2017), p. 81558.

[JT20] L. Jing and Y. Tian. "Self-supervised visual feature learning with deep neural networks: A survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.11 (2020), pp. 4037–4058.

[KC09] T. Kavzoglu and I. Colkesen. "A kernel functions analysis for support vector machines for land cover classification". In: *International Journal of Applied Earth Observation and Geoinformation* 11.5 (2009), pp. 352–359.

[KP19] L. Keller and K. Pantel. "Unravelling tumour heterogeneity by single-cell profiling of circulating tumour cells". In: *Nature Reviews Cancer* 19.10 (2019), pp. 553–567.

[Ket+21] N. Ketkar, J. Moolayil, N. Ketkar, et al. "Convolutional neural networks". In: *Deep Learning with Python: Learn Best Practices of Deep Learning Models with PyTorch* (2021), pp. 197–242.

[KM21] N. Ketkar and J. Moolayil. "Feed-forward neural networks". In: *Deep Learning with Python: Learn Best Practices of Deep Learning Models with PyTorch* (2021), pp. 93–131.

[Kha+23] A. Khan, Z. Rauf, A. Sohail, et al. "A survey of the vision transformers and their CNN-transformer based variants". In: *Artificial Intelligence Review* 56.Suppl 3 (2023), pp. 2917–2970.

[Kha+22] S. Khan, M. Naseer, M. Hayat, et al. "Transformers in vision: A survey". In: *ACM Computing Surveys* 54.10s (2022), pp. 1–41.

[Kra+11] J. Kraan, S. Sleijfer, M. H. Strijbos, et al. "External quality assurance of circulating tumor cell enumeration using the CellSearch® system: a feasibility study". In: *Cytometry Part B: Clinical Cytometry* 80.2 (2011), pp. 112–118.

[KK13] O. Kramer and O. Kramer. "K-nearest neighbors". In: *Dimensionality Reduction with Unsupervised Nearest Neighbors* (2013), pp. 13–23.

[KT21] Z. Kuang and X. Tie. "Flow-based video segmentation for human head and shoulders". In: *arXiv preprint arXiv:2104.09752* (2021).

[KL51] S. Kullback and R. A. Leibler. "On information and sufficiency". In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86.

[KR16] K. M. Kumar and A. R. M. Reddy. "A fast DBSCAN clustering algorithm by accelerating neighbor searching using Groups method". In: *Pattern Recognition* 58 (2016), pp. 39–48.

[LSV19] A. Labach, H. Salehinejad, and S. Valaee. "Survey of dropout methods for deep neural networks". In: *arXiv preprint arXiv:1904.13310* (2019).

[LTK16] T. B. Lannin, F. I. Thege, and B. J. Kirby. "Comparison and optimization of machine learning methods for automated classification of circulating tumor cells". In: *Cytometry Part A* 89.10 (2016), pp. 922–931.

[Law+23] R. Lawrence, M. Watters, C. R. Davies, et al. "Circulating tumour cells for early detection of clinically relevant cancer". In: *Nature Reviews Clinical Oncology* 20.7 (2023), pp. 487–500.

[Lee+17] H. J. Lee, M. F. Diaz, K. M. Price, et al. "Fluid shear stress activates YAP1 to promote cancer cell motility". In: *Nature Communications* 8.1 (2017), p. 14122.

[Li+12] C. Li, S. Zhang, H. Zhang, et al. "Using the K-nearest neighbor algorithm for the classification of lymph node metastasis in gastric cancer". In: *Computational and Mathematical Methods in Medicine* 2012.1 (2012), p. 876545.

[Li+21] Z. Li, F. Liu, W. Yang, et al. "A survey of convolutional neural networks: analysis, applications, and prospects". In: *IEEE Transactions on Neural Networks and Learning Systems* 33.12 (2021), pp. 6999–7019.

[Lin+21] D. Lin, L. Shen, M. Luo, et al. "Circulating tumor cells: biology and clinical significance". In: *Signal Transduction and Targeted Therapy* 6.1 (2021), p. 404.

[Lin+17] Z. Lin, M. Feng, C. N. d. Santos, et al. "A structured self-attentive sentence embedding". In: *arXiv preprint arXiv:1703.03130* (2017).

[Lin+12] L. S. Lindström, E. Karlsson, U. M. Wilking, et al. "Clinically used breast cancer markers such as estrogen receptor, progesterone receptor, and human epidermal growth factor receptor 2 are unstable throughout tumor progression". In: *Journal of Clinical Oncology* 30.21 (2012), pp. 2601–2608.

[Liu+23] Y. Liu, Y. Zhang, Y. Wang, et al. "A survey of visual transformers". In: *IEEE Transactions on Neural Networks and Learning Systems* 35.6 (2023), pp. 7478–7498.

[Liu+19] Z. Liu, Y.-Q. Song, V. S. Sheng, et al. "Liver CT sequence segmentation based with improved U-Net and graph cut". In: *Expert Systems with Applications* 126 (2019), pp. 54–63.

[Lov+22] G. Lovisotto, N. Finnie, M. Munoz, et al. "Give me your attention: Dot-product attention considered harmful for adversarial patch robustness". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 15234–15243.

[Lov+21] M. Lovrić, T. Đuričić, H. T. Tran, et al. "Should we embed in chemistry? A comparison of unsupervised transfer learning with PCA, UMAP, and VAE on molecular fingerprints". In: *Pharmaceuticals* 14.8 (2021), p. 758.

[Ma+24] L. Ma, H. Guo, Y. Zhao, et al. "Liquid biopsy in cancer current: status, challenges and future prospects". In: *Signal Transduction and Targeted Therapy* 9.1 (2024), p. 336.

[MM21] J. Majidpoor and K. Mortezaee. "Steps in metastasis: an updated review". In: *Medical Oncology* 38.1 (2021), p. 3.

[MB20] C. Malzer and M. Baum. "A hybrid approach to hierarchical density-based cluster selection". In: *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*. IEEE. 2020, pp. 223–228.

[MYS16] Y. Mao, Z. Yin, and J. Schober. "A deep convolutional neural network trained on representative samples for circulating tumor cell detection". In: *2016 IEEE Winter Conference on Applications of Computer Vision*. IEEE. 2016, pp. 1–6.

[MS18] M. Martinez and R. Stiefelhagen. "Taming the cross entropy loss". In: *German Conference on Pattern Recognition*. Springer. 2018, pp. 628–637.

[Mar24] V. Marx. "Seeing data as t-SNE and UMAP do". In: *Nature Methods* 21.6 (2024), pp. 930–933.

[MO16] J. Massagué and A. C. Obenauf. "Metastatic colonization by circulating tumour cells". In: *Nature* 529.7586 (2016), pp. 298–306.

[MDB23] J. Maurício, I. Domingues, and J. Bernardino. "Comparing vision transformers and convolutional neural networks for image classification: A literature review". In: *Applied Sciences* 13.9 (2023), p. 5521.

[MHM18] L. McInnes, J. Healy, and J. Melville. "Umap: Uniform manifold approximation and projection for dimension reduction". In: *arXiv preprint arXiv:1802.03426* (2018).

[MHA23] L. McInnes, J. Healy, and S. Astels. *hdbscan Documentation*. Release 0.8.1. Accessed: May 1, 2025. 2023.

[McI] L. McInnes. *Clusterable Data*. `https://github.com/lmcinnes/hdbscan/blob/master/notebooks/clusterable_data.npy`. Accessed: May 1, 2025.

[McQ+17] J. A. McQuerry, J. T. Chang, D. D. Bowtell, et al. "Mechanisms and clinical implications of tumor heterogeneity and convergence on recurrent phenotypes". In: *Journal of Molecular Medicine* 95.11 (2017), pp. 1167–1178.

[MSB13] S. A. Medjahed, T. A. Saadi, and A. Benyettou. "Breast cancer diagnosis by using k-nearest neighbor with different distances and classification rules". In: *International Journal of Computer Applications* 62.1 (2013).

[MH20] M. A. Mercioni and S. Holban. "The most used activation functions: Classic versus current". In: *2020 International Conference on Development and Application Systems*. IEEE. 2020, pp. 141–145.

[MDT10] M. C. Miller, G. V. Doyle, and L. W. Terstappen. "Significance of circulating tumor cells detected by the CellSearch system in patients with metastatic breast colorectal and prostate cancer". In: *Journal of Oncology* 2010.1 (2010), p. 617421.

[Mit18] V. Mittal. "Epithelial mesenchymal transition in tumor metastasis". In: *Annual Review of Pathology: Mechanisms of Disease* 13 (2018), pp. 395–412.

[MSP05] A. M. Molinaro, R. Simon, and R. M. Pfeiffer. "Prediction error estimation: a comparison of resampling methods". In: *Bioinformatics* 21.15 (2005), pp. 3301–3307.

[MMC22] O. A. Montesinos López, A. Montesinos López, and J. Crossa. "Fundamentals of artificial neural networks and deep learning". In: *Multivariate Statistical Machine Learning Methods for Genomic Prediction*. Springer, 2022, pp. 379–425.

[MLP24] A. Morozov, V. Levkivskyi, and D. Plechystyy. "Activation functions in neural networks: Overview and comparison". In: *Scientific Notes of VI Vernadsky Taurida National University. Series: Technical Sciences* 35.74 (2024).

[Mur22] I. Muraina. "Ideal dataset splitting ratios in machine learning algorithms: general concerns for data scientists and data analysts". In: *7th International Mardin Artuklu Scientific Research Conference*. 2022, pp. 496–504.

[MMA20] A. Mustapha, L. Mohamed, and K. Ali. "An overview of gradient descent algorithm optimization in machine learning: Application in the ophthalmology field". In: *International Conference on Smart Applications and Data Analysis*. Springer. 2020, pp. 349–359.

[Nae+23] S. Naeem, A. Ali, S. Anam, et al. "An unsupervised machine learning algorithms: Comprehensive review". In: *International Journal of Computing and Digital Systems* (2023).

[Nan+20a] A. Nanou, L. Mol, F. A. Coumans, et al. "Endothelium-derived extracellular vesicles associate with poor prognosis in metastatic colorectal cancer". In: *Cells* 9.12 (2020), p. 2688.

[Nan+20b] A. Nanou, M. C. Miller, L. L. Zeune, et al. "Tumour-derived extracellular vesicles in blood of metastatic cancer patients associate with overall survival". In: *British Journal of Cancer* 122.6 (2020), pp. 801–811.

[Nan+24] A. Nanou, N. H. Stoecklein, D. Doerr, et al. "Training an automated circulating tumor cell classifier when the true classification is uncertain". In: *PNAS Nexus* 3.2 (2024), pgae048.

[Nie+23] M. Nielsen, L. Wenderoth, T. Sentker, et al. "Self-supervision for medical image classification: State-of-the-art performance with˜ 100 labeled training samples per class". In: *Bioengineering* 10.8 (2023), p. 895.

[Nir+22] R. Nirthika, S. Manivannan, A. Ramanan, et al. "Pooling in convolutional neural networks for medical image analysis: a survey and an empirical study". In: *Neural Computing and Applications* 34.7 (2022), pp. 5321–5347.

[El-+21] A. El-Nouby, H. Touvron, M. Caron, et al. "Xcit: Cross-covariance image transformers". In: *arXiv preprint arXiv:2106.09681* (2021).

[Nwa+18] C. Nwankpa, W. Ijomah, A. Gachagan, et al. "Activation functions: Comparison of trends in practice and research for deep learning". In: *arXiv preprint arXiv:1811.03378* (2018).

[ON15] K. O'shea and R. Nash. "An introduction to convolutional neural networks". In: *arXiv preprint arXiv:1511.08458* (2015).

[PA16] K. Pantel and C. Alix-Panabieres. "Functional studies on viable circulating tumor cells". In: *Clinical Chemistry* 62.2 (2016), pp. 328–334.

[PC13] A. Patle and D. S. Chouhan. "SVM kernel functions for classification". In: *2013 International Conference on Advances in Technology and Engineering*. IEEE. 2013, pp. 1–9.

[PBC21] C. Pealat, G. Bouleux, and V. Cheutet. "Improved Time-Series Clustering with UMAP dimension reduction method". In: *2020 25th International Conference on Pattern Recognition*. IEEE. 2021, pp. 5658–5665.

[Pea01] K. Pearson. "LIII. On lines and planes of closest fit to systems of points in space". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572.

[PSC23] Y. Peng, M. Sonka, and D. Z. Chen. "U-net v2: Rethinking the skip connections of u-net for medical image segmentation". In: *arXiv preprint arXiv:2311.17791* (2023).

[Pin+20] W. H. L. Pinaya, S. Vieira, R. Garcia-Dias, et al. "Convolutional neural networks". In: *Machine Learning*. Elsevier, 2020, pp. 173–191.

[Pop+09] M.-C. Popescu, V. E. Balas, L. Perescu-Popescu, et al. "Multilayer perceptron and neural networks". In: *WSEAS Transactions on Circuits and Systems* 8.7 (2009), pp. 579–588.

[RTK24] O. Rainio, J. Teuho, and R. Klén. "Evaluation metrics and statistical tests for machine learning". In: *Scientific Reports* 14.1 (2024), p. 6086.

[Ram+25] M. Ramesh, M. T. Selvan, P. Sreenivas, et al. "Advanced machine learning-driven characterization of new natural cellulosic Lablab purpureus fibers through PCA and K-means clustering techniques". In: *International Journal of Biological Macromolecules* 306 (2025), p. 141589.

[RM12] R. Ramteke and K. Y. Monali. "Automatic medical image classification and abnormality detection using k-nearest neighbour". In: *International Journal of Advanced Computer Research* 2.4 (2012), p. 190.

[Ran+23a] X. Ran, Y. Xi, Y. Lu, et al. "Comprehensive survey on hierarchical clustering algorithms and the recent developments". In: *Artificial Intelligence Review* 56.8 (2023), pp. 8219–8264.

[Ran+23b] V. Rani, S. T. Nabi, M. Kumar, et al. "Self-supervised learning: A succinct review". In: *Archives of Computational Methods in Engineering* 30.4 (2023), pp. 2761–2775.

[Rao+05] C. G. Rao, D. Chianese, G. V. Doyle, et al. "Expression of epithelial cell adhesion molecule in carcinoma cells present in blood and primary and metastatic tumors". In: *International Journal of Oncology* 27.1 (2005), pp. 49–57.

[Ray+24] M. E. Rayed, S. S. Islam, S. I. Niha, et al. "Deep learning for medical image segmentation: State-of-the-art advancements and challenges". In: *Informatics in Medicine Unlocked* 47 (2024), p. 101504.

[Raz+23] R. Raza, F. Zulfiqar, M. O. Khan, et al. "Lung-EffNet: Lung cancer classification using EfficientNet from CT-scan images". In: *Engineering Applications of Artificial Intelligence* 126 (2023), p. 106902.

[Rie+07] S. Riethdorf, H. Fritsche, V. Müller, et al. "Detection of circulating tumor cells in peripheral blood of patients with metastatic breast cancer: a validation study of the CellSearch system". In: *Clinical Cancer Research* 13.3 (2007), pp. 920–928.

[Rie+18] S. Riethdorf, L. O'Flaherty, C. Hille, et al. "Clinical applications of the CellSearch platform in cancer patients". In: *Advanced Drug Delivery Reviews* 125 (2018), pp. 102–121.

[Rin+23] A. Ring, B. D. Nguyen-Sträuli, A. Wicki, et al. "Biology, vulnerabilities and clinical applications of circulating tumour cells". In: *Nature Reviews Cancer* 23.2 (2023), pp. 95–111.

[RFB15] O. Ronneberger, P. Fischer, and T. Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical Image Computing and Computer Assisted Intervention*. Springer. 2015, pp. 234–241.

[Rud16] S. Ruder. "An overview of gradient descent optimization algorithms". In: *arXiv preprint arXiv:1609.04747* (2016).

[SK23a] I. Salehin and D.-K. Kang. "A review on dropout regularization approaches for deep neural networks within the scholarly domain". In: *Electronics* 12.14 (2023), p. 3106.

[SA18] M. Sarigül and M. Avci. "Performance comparison of different momentum techniques on deep reinforcement learning". In: *Journal of Information and Telecommunication* 2.2 (2018), pp. 205–216.

[SL11] A. Satelli and S. Li. "Vimentin in cancer and its potential as a molecular target for cancer therapy". In: *Cellular and Molecular Life Sciences* 68.18 (2011), pp. 3033–3046.

[Sch+18] U. Schmidt, M. Weigert, C. Broaddus, et al. "Cell detection with star-convex polygons". In: *International Conference on Medical Image Computing and Computer Assisted Intervention*. Springer. 2018, pp. 265–273.

[Sha48] C. E. Shannon. "A mathematical theory of communication". In: *The Bell System Technical Journal* 27.3 (1948), pp. 379–423.

[SS23] D.-S. Shim and J. Shim. "A modified stochastic gradient descent optimization algorithm with random learning rate for machine learning and deep learning". In: *International Journal of Control, Automation and Systems* 21.11 (2023), pp. 3825–3831.

[Shl14] J. Shlens. "Notes on kullback-leibler divergence and likelihood". In: *arXiv preprint arXiv:1404.2000* (2014).

[SK19] C. Shorten and T. M. Khoshgoftaar. "A survey on image data augmentation for deep learning". In: *Journal of Big Data* 6.1 (2019), pp. 1–48.

[SK23b] R. Siddalingappa and S. Kanagaraj. "K-nearest-neighbor algorithm to predict the survival time and classification of various stages of oral cancer: a machine learning approach". In: *F1000Research* 11 (2023), p. 70.

[Sid+21] N. Siddique, S. Paheding, C. P. Elkin, et al. "U-net and its variants for medical image segmentation: A review of theory and applications". In: *IEEE Access* 9 (2021), pp. 82031–82057.

[SS19] J. A. Sidey-Gibbons and C. J. Sidey-Gibbons. "Machine learning in medicine: a practical introduction". In: *BMC Medical Research Methodology* 19.1 (2019), p. 64.

[SB19] J. Singh and R. Banerjee. "A study on single and multi-layer perceptron neural network". In: *2019 3rd International Conference on Computing Methodologies and Communication*. IEEE. 2019, pp. 35–40.

[SVM14] C. O. S. Sorzano, J. Vargas, and A. P. Montano. "A survey of dimensionality reduction techniques". In: *arXiv preprint arXiv:1403.2877* (2014).

[SC19] A. Starczewski and A. Cader. "Determining the EPS parameter of the DBSCAN algorithm". In: *International Conference on Artificial Intelligence and Soft Computing*. Springer. 2019, pp. 420–430.

[Ste+22] M. Stevens, A. Nanou, L. W. Terstappen, et al. "StarDist image segmentation improves circulating tumor cell detection". In: *Cancers* 14.12 (2022), p. 2916.

[SA22] G. Stewart and M. Al-Khassaweneh. "An implementation of the HDBSCAN* clustering algorithm". In: *Applied Sciences* 12.5 (2022), p. 2405.

[Sun+17] M. Sun, Z. Song, X. Jiang, et al. "Learning pooling for convolutional neural network". In: *Neurocomputing* 224 (2017), pp. 96–104.

[Sun+21] H. Sung, J. Ferlay, R. L. Siegel, et al. "Global cancer statistics 2020: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries". In: *CA: A Cancer Journal for Clinicians* 71.3 (2021), pp. 209–249.

[Sve+14] C.-M. Svensson, S. Krusekopf, J. Lücke, et al. "Automated detection of circulating tumor cells with naive Bayesian classifiers". In: *Cytometry Part A* 85.6 (2014), pp. 501–511.

[Swe+13] J. F. Swennenhuis, J. Reumers, K. Thys, et al. "Efficiency of whole genome amplification of single circulating tumor cells enriched by CellSearch and sorted by FACS". In: *Genome Medicine* 5.11 (2013), p. 106.

[Swe+16] J. F. Swennenhuis, G. van Dalum, L. L. Zeune, et al. "Improving the CellSearch® system". In: *Expert Review of Molecular Diagnostics* 16.12 (2016), pp. 1291–1305.

[TL19] M. Tan and Q. Le. "Efficientnet: Rethinking model scaling for convolutional neural networks". In: *International Conference on Machine Learning*. Vol. 97. PMLR. 2019, pp. 6105–6114.

[Tan+23] M. Tanemura, M. Kashiwazaki, K. Matsumoto, et al. "Functional studies on viable circulating tumor cells (v-CTCs) and frequent expression of PD-L1 on v-CTCs in pancreatic cancer". In: *Cancer Research* 83.7_Supplement (2023), pp. 5603–5603.

[TB22] B. C. Taylor and J. M. Balko. "Mechanisms of MHC-I downregulation and role in immunotherapy response". In: *Frontiers in Immunology* 13 (2022), p. 844866.

[Ter+25] J. Terven, D.-M. Cordova-Esparza, J.-A. Romero-González, et al. "A comprehensive survey of loss functions and metrics in deep learning". In: *Artificial Intelligence Review* 58.7 (2025), p. 195.

[TD24] A. Thakur and C. Dhawale. "Activation Functions: Bridging the Gap Between Theory and Application in Deep Learning". In: *2024 International Conference on Innovations and Challenges in Emerging Technologies*. IEEE. 2024, pp. 1–6.

[TZZ23] Y. Tian, Y. Zhang, and H. Zhang. "Recent advances in stochastic gradient descent in deep learning". In: *Mathematics* 11.3 (2023), p. 682.

[VH08] L. Van der Maaten and G. Hinton. "Visualizing data using t-SNE". In: *Journal of Machine Learning Research* 9.11 (2008), pp. 2579–2605.

[Vas+17] A. Vaswani, N. Shazeer, N. Parmar, et al. "Attention is all you need". In: *Advances in Neural Information Processing Systems* 30 (2017).

[WWL13] S. Wager, S. Wang, and P. S. Liang. "Dropout training as adaptive regularization". In: *Advances in Neural Information Processing Systems* 26 (2013).

[Wan+16] D. Wang, A. Khosla, R. Gargeya, et al. "Deep learning for identifying metastatic breast cancer". In: *arXiv preprint arXiv:1606.05718* (2016).

[Wan+18a] M. Wang, S. Lu, D. Zhu, et al. "A high-speed and low-complexity architecture for softmax function in deep learning". In: *2018 IEEE Asia Pacific Conference on Circuits and Systems*. IEEE. 2018, pp. 223–226.

[Wan+18b] W.-C. Wang, X.-F. Zhang, J. Peng, et al. "Survival mechanisms and influence factors of circulating tumor cells". In: *BioMed RResearch International* 2018.1 (2018), p. 6304701.

[WYZ21] X. Wang, L. Yan, and Q. Zhang. "Research on the application of gradient descent algorithm in machine learning". In: *2021 International Conference on Computer Network, Electronic and Automation*. IEEE. 2021, pp. 11–15.

[Wan+21] Y. Wang, S. Yu, Y. Gu, et al. "Fast parallel algorithms for euclidean minimum spanning tree and hierarchical spatial clustering". In: *Proceedings of the 2021 International Conference on Management of Data*. 2021, pp. 1982–1995.

[Was+22] M. Wassel, A. M. Hamdi, N. Adly, et al. "Vision transformers based classification for glaucomatous eye condition". In: *2022 26th International Conference on Pattern Recognition*. IEEE. 2022, pp. 5082–5088.

[WS22] M. Weigert and U. Schmidt. "Nuclei instance segmentation and classification in histopathology images with stardist". In: *2022 IEEE International Symposium on Biomedical Imaging Challenges*. IEEE. 2022, pp. 1–4.

[Wen+04] P. T. Went, A. Lugli, S. Meier, et al. "Frequent EpCam protein expression in human carcinomas". In: *Human Pathology* 35.1 (2004), pp. 122–128.

[Wen+24] R. Wenta, J. Richert, A. Muchlińska, et al. "Measurable morphological features of single circulating tumor cells in selected solid tumors—A pilot study". In: *Cytometry Part A* 105.12 (2024), pp. 883–892.

[WDT14] S. de Wit, G. van Dalum, and L. W. Terstappen. "Detection of circulating tumor cells". In: *Scientifica* 2014.1 (2014), p. 819362.

[Wol+23] S. Wolf, M. Lalit, K. McDole, et al. "Unsupervised learning of object-centric embeddings for cell instance segmentation in microscopy images". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 21263–21272.

[XG18] Y. Xu and R. Goodacre. "On splitting training and validation set: a comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning". In: *Journal of Analysis and Testing* 2.3 (2018), pp. 249–262.

[Yam+18] R. Yamashita, M. Nishio, R. K. G. Do, et al. "Convolutional neural networks: an overview and application in radiology". In: *Insights into Imaging* 9.4 (2018), pp. 611–629.

[Yan+22] S. Yang, W. Xiao, M. Zhang, et al. "Image data augmentation for deep learning: A survey". In: *arXiv preprint arXiv:2204.08610* (2022).

[YY17] K. T. Yeung and J. Yang. "Epithelial-mesenchymal transition in tumor metastasis". In: *Molecular Oncology* 11.1 (2017), pp. 28–39.

[Yeu+22] M. Yeung, E. Sala, C.-B. Schönlieb, et al. "Unified focal loss: Generalising dice and cross entropy-based losses to handle class imbalanced medical image segmentation". In: *Computerized Medical Imaging and Graphics* 95 (2022), p. 102026.

[ZZR19] R. Zemouri, N. Zerhouni, and D. Racoceanu. "Deep learning in the biomedical applications: Recent and future status". In: *Applied Sciences* 9.8 (2019), p. 1526.

[Zeu+17] L. Zeune, G. van Dalum, C. Decraene, et al. "Quantifying HER-2 expression on circulating tumor cells by ACCEPT". In: *PlOS One* 12.10 (2017), e0186562.

[Zeu+20] L. L. Zeune, Y. E. Boink, G. van Dalum, et al. "Deep learning of circulating tumour cells". In: *Nature Machine Intelligence* 2.2 (2020), pp. 124–133.

[Zha+17] D. Zhang, L. Zhao, P. Zhou, et al. "Circulating tumor microemboli (CTM) and vimentin+ circulating tumor cells (CTCs) detected by a size-based platform predict worse prognosis in advanced colorectal cancer patients during chemotherapy". In: *Cancer Cell International* 17.1 (2017), p. 6.

[Zha19] J. Zhang. "Gradient descent based optimization algorithms for deep learning models training". In: *arXiv preprint arXiv:1903.03614* (2019).

[Zha12] Y. Zhang. "Support vector machine classification algorithm and its application". In: *International Conference on Information Computing and Applications*. Springer. 2012, pp. 179–186.

[ZZ24] L. Zhao and Z. Zhang. "A improved pooling method for convolutional neural networks". In: *Scientific Reports* 14.1 (2024), p. 1589.

[Zho24] J. Zhou. "Predicting Stock Price by Using Attention-Based Hybrid LSTM Model". In: *Asian Journal of Basic Science & Research* 6.2 (2024), pp. 145–158.

Appendix A

Overview of publications

During the course of this doctoral research, the following journal publications and conference papers have been published or are in revision:

[Hus+23] **H. Husseini**, M. Nielsen, K. Pantel, H. Wikman, S. Riethdorf, and R. Werner. "Label Efficient Classification in Liquid Biopsy Data by Self-supervision". In: *Bildverarbeitung für die Medizin 2023* (2023), p. 261.

[Woe+24] L.-M. Woelk, D. Kovacevic, **H. Husseini**, F. Förster, F. Gerlach, F. Möckl, M. Altfeld, A. H. Guse, B.-P. Diercks, and R. Werner. "DARTS: an open-source Python pipeline for Ca_{2+} microdomain analysis in live cell imaging data". In: *Frontiers in Immunology* 14 (2024), p. 1299435.

[Hus+25] **H. Husseini-Wüsthoff**, S. Riethdorf, A. Schneeweiss, A. Trumpp, K. Pantel, H. Wikman, M. Nielsen, and R. Werner. "Cluster-based human-in-the-loop strategy for improving machine learning-based circulating tumor cell detection in liquid biopsy". In: *Patterns* 6.6 (2025), p. 101285.

[Ott+25] L. F. Ott, L. Keller, N. Bentley, **H. Husseini-Wüsthoff**, R. Werner, M. Zinggeler, J. Heidler, P. Mossahebi Mohammadi, C. Coith, A. Pradines, N. H. Stoecklein, M. Löptien, S. Peine, M. Geffken, C. Güsmer, M. Netkova-Heintzen, V. Müller, E. Laakmann, V. Thewes, T. M. Deutsch, L. L. Michel, A. Schneeweiss, A. Trumpp, J. Rühe, T. Brandtstetter, S. Riethdorf, and K. Pantel. "Fast enrichment and detection of circulating tumor cells from large volumes of whole blood of breast cancer patients utilizing a functionalized bioaffinity CTC filtration membrane". In: *International Journal of Cancer* (2025).

Eidesstattliche Versicherung/ Affidavit

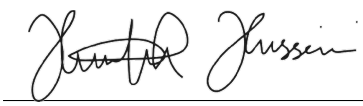
Eidesstattliche Versicherung:
Hiermit versichere ich an Eides statt, die vorliegende Dissertationsschrift selbst verfasst und keine anderen als die angegebenen Hilfsmittel und Quellen benutzt zu haben.

Sofern im Zuge der Erstellung der vorliegenden Dissertationsschrift generative Künstliche Intelligenz (gKI) basierte elektronische Hilfsmittel verwendet wurden, versichere ich, dass meine eigene Leistung im Vordergrund stand und dass eine vollständige Dokumentation aller verwendeten Hilfsmittel gemäß der Guten wissenschaftlichen Praxis vorliegt. Ich trage die Verantwortung für eventuell durch die gKI generierte fehlerhafte oder verzerrte Inhalte, fehlerhafte Referenzen, Verstöße gegen das Datenschutz- und Urheberrecht oder Plagiate.

Affidavit:
I hereby declare and affirm that this doctoral dissertation is my own work and that I have not used any aids and sources other than those indicated.

If electronic resources based on generative artificial intelligence (gAI) were used in the course of writing this dissertation, I confirm that my own work was the main and value-adding contribution and that complete documentation of all resources used is available in accordance with good scientific practice. I am responsible for any erroneous or distorted content, incorrect references, violations of data protection and copyright law or plagiarism that may have been generated by the gAI.

Hamburg, February 2, 2026



Hümeyra Husseini-Wüsthoff