Methods for Robust and Efficient Deep Learning and Improved Data Utilization in Large-Scale Data Generation Settings

Zur Erlangung des akademischen Grades eines

DOKTORS DER INGENIEURWISSENSCHAFTEN
(Dr.-Ing.)

von der KIT-Fakultät für Maschinenbau des
Karlsruher Instituts für Technologie (KIT)

angenommene

DISSERTATION

von

M.Sc. Luca Rettenberger

Tag der mündlichen Prüfung:	21. Oktober 2025
Hauptreferent:	Prof. Dr. Markus Reischl
Korreferenten:	Prof. Dr. Veit Hagenmeyer
	Prof. Dr. Christof Niemeyer

Luca Rettenberger
Methods for Robust and Efficient Deep Learning and Improved Data Utilization in Large-Scale Data Generation Settings
Dissertation, 21. Oktober 2025
Gutachter: Prof. Dr. Markus Reischl, Prof. Dr. Veit Hagenmeyer und Prof. Dr. Christof Niemeyer
Betreuer: Prof. Dr. Markus Reischl

Karlsruher Institut für Technologie (KIT)
Maschinelles Lernen für Hochdurchsatzverfahren und Mechatronik (ML4HOME)
Automation für Labore (ATLAS)
Institut für Automation und angewandte Informatik (IAI)
Fakultät für Maschinenbau
Hermann-von-Helmholtz-Platz 1
76344 Eggenstein-Leopoldshafen

Kurzfassung

Die bahnbrechenden Fortschritte im Deep Learning Bereich haben viele Fachgebiete revolutioniert und die Arbeitsweisen grundlegend verändert. Auch im Maschinenbau gewinnt Deep Learning zunehmend an Bedeutung, zum Beispiel in der biomedizinischen und materialwissenschaftlichen Bildgebung. Diese Anwendungen erfordern spezielles Fachwissen bei der Entwicklung moderner Geräte, Technologien und Analysemethoden. Diese Arbeit widmet sich den zentralen Herausforderungen, die Robustheit und Effizienz von Deep Learning-Verfahren zu steigern. Dabei stehen drei Kernaspekte im Mittelpunkt: die Verbesserung von Annotationsprozessen, die effektive Nutzung nicht annotierter Daten und die Anpassung neuronaler Netze an die Komplexität realer Anwendungen. Der hier vorgestellte ganzheitliche Ansatz umfasst die gesamte Deep Learning Pipeline. Zunächst werden neuartige Methoden zur automatischen Erstellung von Segmentierungsmasken aus Rohdaten entwickelt, wodurch der manuelle Annotationsaufwand erheblich sinkt. Zudem werden innovative Trainingsansätze eingeführt, die es ermöglichen, mit Daten zu arbeiten, bei denen unterschiedliche Proben jeweils verschiedene oder nur teilweise vorhandene Annotationen besitzen, einschließlich einer neuen Zielfunktion und mehrerer Bewertungsmetriken. Ein weiterer Schwerpunkt liegt auf der Extraktion von Wissen aus nicht annotierten Daten, insbesondere für anspruchsvolle Aufgaben wie die semantische Segmentierung. Umfassende Frameworks zur strukturierten Evaluierung neuartiger Methoden werden entwickelt und in umfangreichen Untersuchungen mit unterschiedlichen Datenmengen und in verschiedenen Anwendungsgebieten getestet. Weiter werden neuronale Netzwerkarchitekturen optimiert, um reale Herausforderungen, wie überlappende Objekte und Unsicherheiten bei der Annotation, zu bewältigen. Hierfür werden Verfahren zur Quantifizierung und Einbeziehung von Unsicherheiten in den Trainingsprozess entwickelt, was die

Zuverlässigkeit der Modelle in komplexen Szenarien steigert. Die Funktionsweise der entwickelten Ansätze wird anhand einer breiten Palette etablierter und neu erstellter Datensätze aus verschiedenen Fachgebieten ausgiebig nachgewiesen. Diese Arbeit leistet einen bedeutenden Beitrag zur Erweiterung der Einsatzmöglichkeiten von Deep Learning in Bereichen mit begrenzter Verfügbarkeit von annotierten Daten. Sie ermöglicht eine deutlich effizientere Nutzung vorhandener Daten und verbessert die Robustheit von Machine-Learning-Modellen in anspruchsvollen Szenarien erheblich. Die entwickelten Methoden und gewonnenen Erkenntnisse tragen dazu bei, wissenschaftliche Entdeckungen zu beschleunigen und die Anwendbarkeit von Deep Learning maßgeblich zu verbessern.

Abstract

Deep learning has revolutionized numerous fields, including mechanical engineering, where it is increasingly important for applications such as biomedical and materials science imaging that rely on expertise in developing advanced devices, technologies, and analytical methods. This work addresses critical challenges in enhancing the robustness and efficiency of deep learning methods by focusing on three key areas: optimizing annotation processes, leveraging unannotated data, and refining neural networks for real-world complexities. To tackle these challenges, this work presents a comprehensive approach that spans the entire deep learning pipeline. First, novel methods are developed for generating segmentation masks from raw data, drastically reducing manual annotation efforts. Innovative approaches are introduced that enable training with data where different samples have varying or only partial annotations, including a new objective function and multiple evaluation metrics. These advancements enable more efficient use of available data and improve annotation processes. Second, the work explores techniques for extracting knowledge from unannotated data, with a focus on challenging tasks such as semantic segmentation. Multiple comprehensive frameworks for the structured evaluation of novel methods for extracting knowledge from raw data are developed and thoroughly tested in extensive studies across diverse application domains and data volumes. Finally, neural network architectures are evaluated and refined to address real-world challenges, including overlapping instances and annotator uncertainties. Methods for quantifying and incorporating uncertainties into the training process are developed, enhancing model robustness in complex scenarios. The effectiveness of the developed approaches is extensively demonstrated using a wide variety of established and novel datasets across different domains, including both commonly used and a diverse number of newly created ones, featuring unique annotations such as uncertainty estimations to simulate challenging real-world applications. This work makes a substantial contribution to enhancing the applicability of deep learning in fields where annotated data is limited. It enables a more efficient use of existing data and greatly improves the robustness of machine learning models in challenging scenarios. The developed methods and findings have extensive implications for accelerating scientific discovery and increasing the practical applicability of deep learning.

Preface

Looking back at my time at the Institute for Automation and Applied Informatics, I am genuinely astonished by how much this place has shaped not only my research but also my perspective on what makes a scientific community thrive. What began as a pursuit of technical knowledge evolved into something far more valuable: a collection of meaningful friendships, unexpected adventures, and countless moments that reminded me why I love what I do.

Beyond the formal structures of academia, it was the people and experiences that made this time truly special. The *Diversity Scanner meetings*, which began together with Lorenz as actual work meetings, quickly evolved into cherished coffee breaks and became a highlight of my weeks. Such events made the hot summers in the *Altbau* bearable. The summer school in Gran Canaria stands out as a particularly memorable chapter. Traveling with André, Marcel, Oli, and Moritz (or was it Maurice?), I learned as much from the conversations and laughter as from the formal lectures. We explored not only the technical content but also the local culture, discovering hidden gems like tropical beer bars and catching the rhythm of local festivals. Other scientific trips, like the BMT conferences, from Innsbruck over Duisburg (or *Dusiburg*, as we call it) to Stuttgart, reinforced that good science happens when people genuinely enjoy working together. Back at home, Markus's Christmas gatherings became legendary affairs, with multi-course dinners that left us all delighted and slightly too full. We even made it to the Canstatter Wasen together, embracing local traditions with the enthusiasm of true Swabians.

This journey would not have been possible without Markus's constant support and guidance; he truly is an exceptional mentor. My heartfelt thanks also go to my colleagues at the IAI: Jan, Marcel, Ralf, Oli, Klaus-Martin, Tim, Yanke,

Angelo, Stephan, Nathalie, Hossein, Nils, Moritz, and Friedrich. Not to forget the special people with whom I shared my office: Hawo, André, Joaquin, and Roman. I'll miss our great conversations. A special thank-you also goes out to the *Scribble Squad* Lorenz, Roman, and (the other) Luca, whose companionship during the writing process made tough moments much easier to bear. Also, Mark deserves a special mention for bringing me to IAI and setting this entire adventure in motion. I would also like to acknowledge Nathan, Andrea, and Olympia from the University of Berkeley for their collaboration, as well as Andreas, Marina, Manuela, Carola, and Bernadette for their indispensable organizational support.

Lastly and above all, I am deeply grateful for the immeasurable and constant support of my great love, Maike. I can't believe you've put up with me for over 13 years now.

This thesis represents not just years of research but also valuable time spent in a community where work and friendship were beautifully intertwined. For that, I am profoundly grateful.

Stuttgart, November 2025 *Luca Rettenberger*

Notation

General Notation

s, α	Scalar (italic Roman and Greek lowercase letter)
$\mathcal{S}$	Set (calligraphic Roman uppercase letter)
$\boldsymbol{v}$	Vector (bold Roman lowercase letter)
v_i	Vector at position i (italic Roman lowercase letter)
$\boldsymbol{M}$	Matrix/Tensor (bold Roman uppercase letter)
$M_{i,j}$	Matrix/Tensor at position (i,j) (italic Roman uppercase letter)
$f : \mathcal{S} \rightarrow \mathcal{V}$	Function mapping from the set $\mathcal{S}$ to the set $\mathcal{V}$
$f_\theta(\boldsymbol{s})$	Function f with vector argument $\boldsymbol{s}$ and parameters θ

Machine Learning

$\mathcal{X}$	Set of samples
$\mathcal{Y}$	Set of labels
$\mathcal{D}$	Dataset
$\mathcal{M}$	Domain
$\boldsymbol{x}$	Single sample
$\boldsymbol{y}$	Single label

$\tilde{\boldsymbol{y}}$	Ground-truth label
$\hat{\boldsymbol{y}}$	Predicted label
θ	Model parameters
$\mathcal{L}$	Loss function
ϕ	Activation function
ω	Weight matrix

Contents

1 Introduction

1.1 Motivation

Modern automated systems generate vast amounts of complex data, fundamentally transforming our interaction with technology and approaches to problem-solving across virtually all domains. Consequently, Deep Learning (DL) has become essential in numerous applications, as it relies on data-driven learning. For example, it enables machines to interpret visual information with unprecedented accuracy in image recognition [Ret23a, Ret24a] and powers systems that facilitate seamless human-machine communication [An22]. The impact of DL extends to virtually all fields of research, such as autonomous systems, where it operates autonomous vehicles [Ber24a, Ber24b] and intelligent robots [Kum17], as well as medical diagnostics [Sch23], aiding in the early detection of diseases and personalization of treatment strategies. In biomedical research, DL enhances the analysis and segmentation of 3D cell cultures [Bru23, Bru25]. In materials science, it enables the development of autonomous laboratories for the accelerated synthesis of novel compounds [Szy23, Ret24c].

The remarkable success of DL stems from its ability to learn complex patterns directly from data. As a subset of Machine Learning (ML), DL acquires knowledge through implicit learning from examples, in contrast to traditional automation processes that rely on explicit instructions in the form of human-written programs, where programmers must anticipate and encode model physics and processes [Goo16]. The learning supervision for DL systems is embedded within the input, which is realized through annotations (labels) that provide the necessary guidance for the model to learn the desired mapping between inputs and outputs. This approach enables DL models to discover intricate structures in high-dimensional data without manual feature engineering, allowing them to adapt successfully to a wide range of tasks and domains.

To achieve optimized performance, recent trends in DL tend towards increasing the number of carefully labeled data points to achieve state-of-the-art results. One example of such an approach is the ImageNet [Rus15] dataset, which comprises over 14 million hand-annotated images of natural scenes and is considered one of the most crucial datasets in modern ML. Systems built on ImageNet achieve outstanding results and acquire general knowledge, enabling strong performance even in areas unrelated to the dataset's content [Xie18, Mat21]. Such results demonstrate that sufficient labeled data allows the construction of generalist ML systems that solve a wide range of problems. While this is theoretically true, labeling the whole world is unfeasible. Hence, the full potential of DL remains untapped, constrained by a critical bottleneck: the availability of *annotated* data. High-quality annotations are rare, especially in areas such as biomedicine or materials science, where labeling is labor-intensive and requires expert knowledge [Sch22b]. The challenge of annotation scarcity is even more pronounced in particularly complex tasks, such as precisely identifying and outlining different objects or regions within an image. This process is especially time-consuming and requires specialized expertise. For instance, materials scientists require meticulous manual effort to identify and delineate individual particles in Scanning Electron Microscopy (SEM) images. Similarly, in biomedical imaging, segmenting anatomical structures relies on the expert judgment of trained radiologists.

As automation continues to evolve, the gap between the amount of data generated and its annotated share continues to widen. For example, advancements in laboratory automation have greatly accelerated material characterization, generating vast quantities of high-resolution images [Hil16, Szy21]. While this surge in data has opened new avenues for research, it has also presented considerable challenges in efficient analysis [Ret24c]. This rapid data generation far outpaces the human capacity for annotation, creating a noticeable bottleneck in the ML pipeline. In fields such as materials science and biomedical research, automated techniques like microscopy systems and high-throughput screening generate thousands of images daily, far exceeding what is feasible to annotate manually by experts

[Sch21b]. This discrepancy between data generation and annotation capabilities results in vast amounts of valuable data remaining unused or underutilized in ML applications. Therefore, the challenge lies in developing efficient annotation methods and creating DL approaches to effectively leverage partially labeled or unlabeled data to extract meaningful insights and patterns.

Given the elaborated challenges, this work aims to enhance the robustness of ML by addressing key bottlenecks in the current DL practice. It focuses on optimizing annotation through algorithmic methods to support and accelerate manual processes, strategically utilizing available data to enhance quality for DL training, extracting knowledge from raw, unannotated data, and refining ML algorithms to handle real-world complexities such as overlapping instances and annotator uncertainties. The goal is to bridge the gap between vast data generation and limited annotation capacity, enabling more efficient and effective DL use across domains, particularly in challenging fields such as biomedicine and materials science.

1.2 Problem Categorization

The preceding motivation highlights the transformative impact of DL across various domains, while underscoring a critical challenge: the scarcity of high-quality, annotated data. To fully grasp the scope of this challenge and formulate targeted research goals, it is essential to examine the current approach in typical DL projects in detail.

Figure 1.1 illustrates the current DL pipeline, highlighting the challenges and inefficiencies in utilizing available information. The process begins with an *automated process* that generates *extensive data*, which undergoes a crucial step of manual *annotation* by an expert, who labels a portion of it. The *annotated data* is subsequently used to train a *deep learning* model, which learns to perform a desired task. However, a considerable amount of *unannotated data* remains and is discarded, essentially *thrown away*. This unused portion contains valuable information that, when properly leveraged, could further enhance the performance. The final trained model, which has only been trained on the annotated subset,

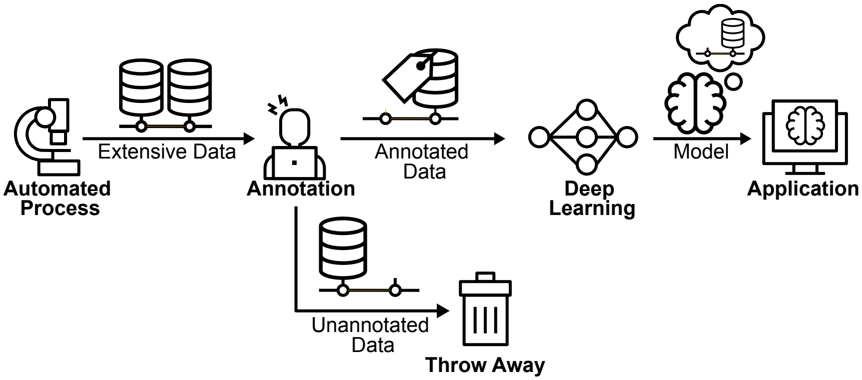


Figure 1.1: The current Deep Learning (DL) pipeline with processes that generate extensive data. The data is partially annotated and used in a model to learn the desired behavior, while the unannotated portion remains unused. The final model, trained on the annotated data, is then utilized in the target application.

is then employed in the target *application*. This pipeline underscores a critical limitation in current DL practices: the underutilization of available resources due to the constraints of manual annotation. It emphasizes the need for more efficient methods to leverage unannotated data and optimize the annotation process, thereby improving the overall effectiveness of DL systems.

The specific critical limitations identifiable in the current DL pipeline are as follows:

1. **Annotation bottlenecks:** The manual annotation process is time-consuming and expensive, especially for complex tasks.
2. **Data underutilization:** Large volumes of unannotated data are discarded, containing valuable information that has the potential to enhance model performance.
3. **Model limitations:** Current DL algorithms structurally fail to address real-world complexities such as overlapping instances and annotator uncertainties.

Given these limitations, this work aims to address the following key research questions:

1. How can the annotation process be optimized to reduce manual effort and improve data quality?

2. How can unannotated data be effectively leveraged to enhance the performance of deep learning models?

3. How can deep learning algorithms be refined to handle real-world complexities and uncertainties better?

By addressing these research questions, this work aims to advance the robustness and efficiency of DL in domains with limited annotated data. The following chapters introduce novel methods for dataset enhancement, unsupervised knowledge extraction, and neural network optimization, enabling more effective handling of real-world challenges. Specifically, the main objectives of this dissertation are:

1. A novel approach for improved information extraction from large-scale unannotated data, enhancing learning in scenarios where manual labeling is impractical.
2. An innovative method for integrating uncertainty into annotation and training, allowing models to address ambiguities in expert labels.
3. A new learning framework facilitating the merging of multiple datasets with incomplete or varying types of annotations for combined training.
4. An automated workflow that generates labels directly from raw data, significantly minimizing manual annotation requirements.
5. A comprehensive benchmarking framework for systematic comparison and refinement of methods that learn from unannotated data.
6. A systematic evaluation protocol tailored for assessing models when data is only partially or inconsistently annotated.
7. A practical guideline for selecting and optimizing neural network architectures, particularly for challenging scenarios such as images with overlapping objects.
8. A collection of curated datasets incorporating partial annotations, uncertainty measures, and complex real-world cases to support thorough evaluation.

9. A series of extensive empirical studies demonstrating reductions in manual effort and notable improvements in model performance achieved through the proposed methods.

These advancements promise not only academic value but also substantial practical impact, accelerating progress in biomedical imaging, materials science, and other critical fields.

1.3 Thesis Organization

According to the established goals, the remainder of this work is organized as follows: Chapter 2 examines the deep learning pipeline in detail to identify areas with significant weaknesses and potential for methodological advancement, providing an overview that sets the stage for the novel contributions in the following chapters. Chapter 3 outlines the essential technical foundations, introducing key concepts in machine learning, deep learning, and computer vision, including neural network architectures, object detection methods, and evaluation metrics. Chapter 4 details the datasets used throughout this work, both established and newly created, outlining their selection criteria and characteristics, and emphasizing their relevance to the challenges addressed in this dissertation. Chapter 5 presents methods for algorithmic dataset enhancement, including frameworks for generating segmentation masks from raw data and approaches for leveraging datasets with mixed or incomplete annotations, thereby minimizing manual annotation and maximizing the use of existing labels. Chapter 6 explores deep learning techniques that leverage unannotated data, introducing novel evaluation frameworks and metrics, and investigating the effectiveness of current methods for extracting knowledge from raw data, particularly in challenging domains such as biomedical imaging and materials science. Chapter 7 focuses on the selection and refinement of neural network models for real-world applications, addressing challenges such as overlapping objects and annotator uncertainties, and presenting strategies for architecture selection and uncertainty estimation. Finally, Chapter 8 summarizes the key contributions of this work, offering concluding remarks and a comprehensive discussion on the advancements made in this work.

2 Challenges and Potentials in the Deep Learning Pipeline

To enable the effective and structured development of strategies for improving the current DL pipeline and answer the research questions of this work, an analysis must first be conducted to determine which areas would benefit the most from enhancements. For this purpose, the various modules comprising the DL pipeline are examined from an overview perspective, allowing those with the most significant potential for improvement to be identified and enabling them to be explored in detail and systematically expanded upon with novel methods in the following chapters.

2.1 Problem Specification

One striking observation is that typical DL approaches do not effectively harness the full potential of available data, as they rely on annotations, which are virtually never provided for each sample. This is particularly pronounced in projects driven by automated processes that generate large volumes of data, as annotating each produced sample is impractical. Consequently, most of the data remains unused, and DL training is conducted on only a small portion of the data. This is especially true for complex tasks, such as providing pixel-wise annotations (segmentation masks), which are particularly burdensome to produce. Subsequently, data containing valuable information goes unused despite being available. Furthermore, the actual training of DL models often relies on suboptimal standard methods that are not tailored to the respective problem, and therefore do not fully utilize the available data for training. To counteract the weaknesses outlined above, the

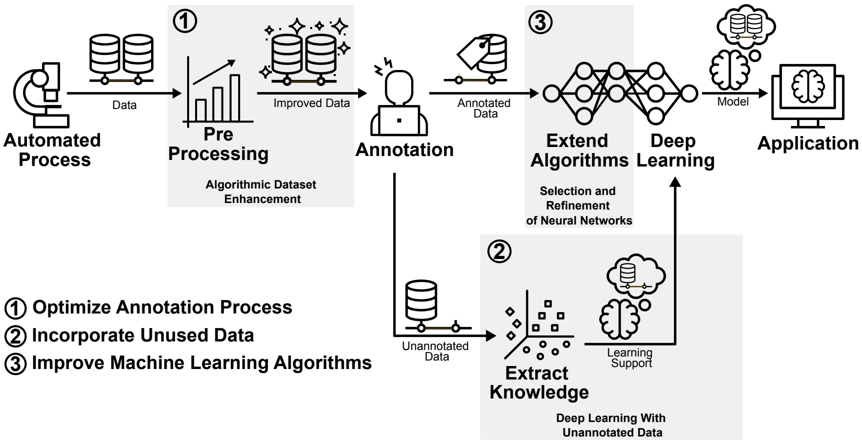

①
③
Automated Process
Data
Pre Processing
Improved Data
Algorithmic Dataset Enhancement
Annotation
Annotated Data
Extend Algorithms
Deep Learning
Selection and Refinement of Neural Networks
Model
Application
②
Unannotated Data
Extract Knowledge
Learning Support
Deep Learning With Unannotated Data
① Optimize Annotation Process
② Incorporate Unused Data
③ Improve Machine Learning Algorithms

Figure 2.1: Conceptual overview of the deep learning pipeline, highlighting the scientific contributions of this thesis with gray backgrounds. An automated process generates extensive data that undergoes pre-processing and is subsequently partially annotated by domain experts. The annotated portion is fed into the primary deep learning pipeline, which learns to perform the main task, such as classification. Meanwhile, the unannotated segment extracts knowledge supporting the primary process. Ultimately, the trained DL system finds application in real-world scenarios.

general DL pipeline is examined for opportunities to achieve more efficient utilization of the data. Figure 2.1 highlights three key areas for scientific contributions (shown in gray): (1) Optimization of annotation processes through algorithmic support. (2) Strategic incorporation of unused raw data through knowledge extraction. (3) Enhancement of ML algorithms to handle real-world complexities. These components work synergistically throughout the entire DL pipeline, from automated data generation and annotation support to unsupervised learning and final model refinement, creating a comprehensive approach to improving robustness. This work enables all data-driven domains to maximize their resources by establishing a universally applicable framework that reduces manual annotation burdens, unlocks value from raw data, and enhances algorithmic robustness. As a result, it accelerates ML deployment, helping to overcome key scalability barriers and increase real-world impact. While the conceptual framework and main ideas are outlined in the following, the respective chapters will systematically explore detailed methodologies and related works for these processes.

2.2 Algorithmic Dataset Enhancement

The first module of the DL process in which potential for methodological development can be identified is the support and acceleration of the manual annotation process and strategic use of the available data to increase the employable data volume and quality for DL training (see Figure 2.2).

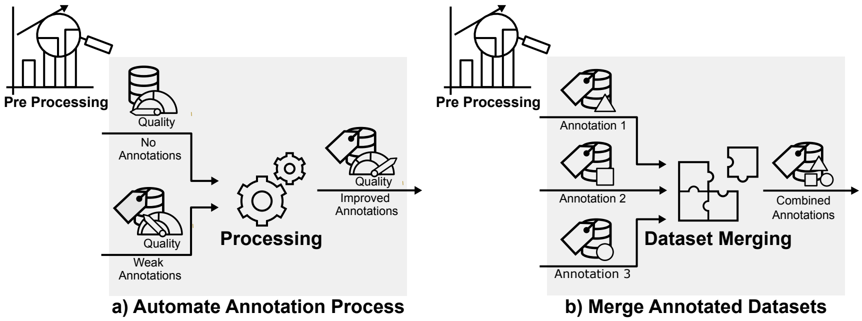


Figure 2.2: Detailed view for the algorithmic dataset enhancement. a) The input dataset is either unannotated or only weakly labeled and is processed to achieve higher-quality annotations. b) Multiple datasets stemming from the same (or similar) domain but with incompatible annotations, such as different class labels, are merged to form a combined, larger dataset that includes all samples and is usable for ML.

The quantity and quality of data annotations are crucial for all subsequent elements in developing ML-based projects. Each sample is valuable, especially when domain experts provide annotations, as time limitations typically constrain their availability. Furthermore, if large amounts of raw data are produced, simplifying the annotation process not only enhances the quality of the annotations but also enables a larger proportion of the data samples to be processed with the same time investment.

Since annotations follow structural patterns, one approach to enhancing both speed and quality involves the development of methods that algorithmically support or partially automate the annotation process (see Figure 2.2a). This includes automating the generation of annotations from raw data or enhancing the quality of annotations by transitioning from those with low information content to more complex ones. For example, classification annotations can be utilized to generate segmentation masks, enabling a transition from simpler to more detailed annotation types.

Another component with high potential to increase the amount of usable data for DL is to enable MLs systems to be trained with multiple datasets simultaneously, rather than being optimized with just one at a time (see Figure 2.2b). However, this integration faces a critical challenge: datasets often define different annotation targets, making direct combination impossible because DL models typically require consistent labeling across all samples. For instance, if one dataset provides segmentation masks for cars and trees, while another offers masks for pedestrians and traffic lights, combining these datasets directly is not feasible because the class labels do not align across all samples. However, since combining multiple datasets into a single, larger dataset usable for ML systems drastically increases the available data for training without requiring additional annotation efforts, methods that propose solutions for training similar datasets in combination are urgently sought.

Given the great potential for the entire DL process, Chapter 5 investigates the abovementioned challenges in-depth and presents approaches and solutions for algorithmic dataset enhancements.

2.3 Deep Learning With Unannotated Data

Since potentially vast amounts of valuable data remain unused if they cannot be annotated due to a lack of resources, including such raw, unannotated data into the ML training process is the second key component for improving the DL pipeline (see Figure 2.3).

In DL-based projects, the primary emphasis is on supervised training using annotated samples. Having annotations allows for formulating specific task objectives and providing clear instructions for the learning process. However, this leads to much data being left unused, as usually, not every sample can be annotated, which is especially pronounced if the annotations are particularly intricate. This issue

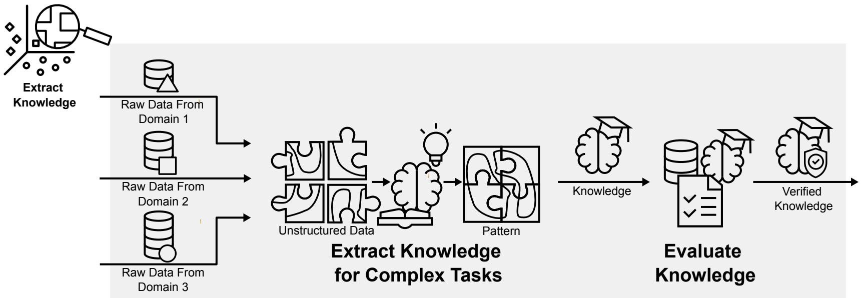


Figure 2.3: Details on deep learning with unannotated data. Raw data from various domains is analyzed to identify patterns without relying on human annotations, enabling the extraction of meaningful knowledge. This knowledge supports supervised learning tasks.

is becoming increasingly significant due to the rapid growth of data from diverse sources, including sensors in millions of cars, billions of cell phones, automated laboratories, and extensive global networks of surveillance and industrial cameras. Consequently, using raw, unannotated data to increase performance on smaller, annotated sets of samples is an emerging field that has received much attention within DL in recent years.

When learning from raw data, supervision is weaker than with annotations, as there are no human-provided labels to guide the learning process. The ML system must independently identify patterns, without explicit instructions about important features or outputs. In contrast, annotated data provides clear objectives. Consequently, models trained on raw data require significantly larger datasets, and the specific content of the data has a strong influence on the training outcome. Profound studies evaluating methods using unannotated data beyond benchmark datasets with classification labels are lacking, presenting considerable potential for investigation in this area, particularly for challenging tasks like semantic segmentation. Furthermore, given the numerous uncertainties and gaps in empirical insights, studies are urgently needed to investigate how much information can be learned from raw data across various application domains and data volumes.

Considering the considerable opportunities within the entire DL pipeline, Chapter 6 comprehensively investigates DL with unannotated data and introduces studies as well as methods that focus on training beyond annotated data.

2.4 Selection and Refinement of Neural Networks

Enhancing the underlying DL algorithms for solving the target task is the third crucial aspect in advancing the effectiveness and efficiency of the ML pipeline. As the final element of the entire process, the algorithms used to solve the target task determine the overall performance and must be carefully selected and optimized to avoid becoming a bottleneck. Therefore, evaluating and refining these algorithms is imperative for achieving superior performance (see Figure 2.4).

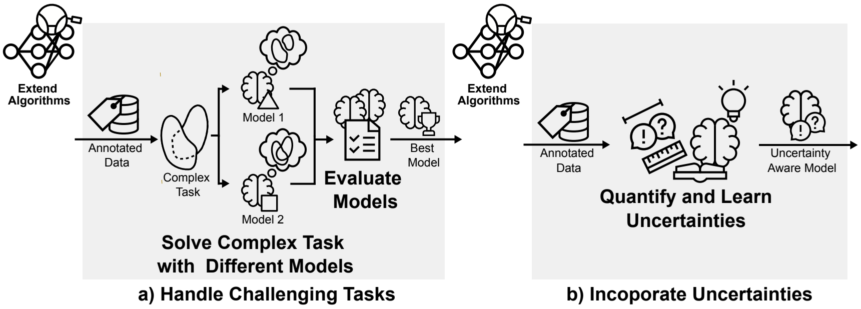


Figure 2.4: Detailed view of the selection and refinement of neural networks. a) An annotated dataset is used to solve a complex task with overlapping instances. One model is able to capture the complexity, while the other one fails. The models are evaluated, and the most fitting one is employed. b) An annotated dataset includes uncertainties that are quantified and integrated into the model to obtain an uncertainty-aware model.

Traditionally, ML methods are evaluated on well-defined datasets characterized by consistency and absence of unusual annotations. However, real-world scenarios often present challenges that reveal limitations in conventional approaches. A key consideration is evaluating whether established methods are capable of effectively addressing the complexities of real-world challenges. For example, as shown in Figure 2.4a, different models exhibit varying capabilities when handling complex tasks, such as overlapping objects. While one model successfully captures the complexities of such a task, the other fails to distinguish between adjacent objects and is unable to detect that the two objects overlap.

Another frequently overlooked element is the consideration of annotator uncertainties in the training process (see Figure 2.4b). Providing unambiguous annotations for ML models is not possible in many domains. For example, different experts annotating the same dataset may interpret certain situations differently, introducing conflicting information within the data. Typically, such discrepancies are solved by streamlining and simplifying the annotations to obtain clear signals for the ML system. Consequently, uncertainties are overlooked in training. Neglecting or eliminating such uncertainties may introduce conflicting information into the dataset, which impedes the training process. Hence, methods are urgently needed to incorporate uncertainties, particularly in domains characterized by substantial discrepancies among experts.

Given the significant opportunities for selecting and refining neural networks, Chapter 7 examines these challenges in detail and presents studies and methods to improve ML algorithms for real-world applications.

3 Technical Foundations

This chapter provides an overview of the fundamentals necessary for understanding the content of this work, introducing the basics of ML including terminology, learning paradigms, and object detection principles, as well as defining uncertainty in the context of this study. Additionally, it presents the various types of datasets, the fundamentals of ML and DL, and the evaluation metrics used.

3.1 Machine Learning

A program is considered to learn if it improves through experience [Mit97]. In ML, a computer program is trained to solve specific tasks without explicit programming but rather by learning to address challenges implicitly. By using data, the program gains experience and enhances its performance over time. Therefore, ML is applied in scenarios where traditional programming is impractical, too time-consuming, or inaccurate. This section explores the core concepts and principles of ML, as it is a fundamental technique employed in this work.

3.1.1 Terminology & Definitions

Key ML terms used in this work are defined in the following. All uses of these terms throughout this work are understood in accordance with the following definitions.

Feature: A feature is an individual measurable property or characteristic that is used for a ML system to learn [Moh18].

Sample: A sample $\boldsymbol{x}$ consists of n features that are depicted as a (multidimensional) vector $\boldsymbol{x} = [x_1, x_2, \dots x_n]^{\mathrm{T}}$. Each component x_i of the vector represents an individual feature [Moh18]. For instance, an RGB image with height h, width w, and three color channels $c = 3$ is encoded as a sample vector $\boldsymbol{x} \in \mathbb{N}^{c \times h \times w}$, where each $x_{i,u,v}$ is a single pixel feature.

Label: A label, denoted as $\boldsymbol{y}$ or $\tilde{\boldsymbol{y}}$, is a descriptive identifier associated with a corresponding sample $\boldsymbol{x}$. The label provides a semantic interpretation of the sample as a numeric value [Moh18]. Within computer vision, labels describe the contents of an image. This can be a description of the objects within the image or a classification of each pixel.

Dataset: A dataset $\mathcal{D}$ is a collection of n samples $\mathcal{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$, which may have a corresponding set of labels $\mathcal{Y} = \{\boldsymbol{y}_1, \dots, \boldsymbol{y}_n\}$. ML algorithms use the samples of a dataset to gain experience in solving a specific task that is either defined by the labels or from the samples themselves.

Input Space: The input space $\mathcal{I}$ is the set of all possible input samples and provides a mathematical framework for analyzing the distribution and relationships of the data [Moh18]. The dimensionality of $\mathcal{I}$ corresponds to the number of features in each sample.

Output Space: The output space $\mathcal{O}$ is the set of all possible outputs or predictions that a ML model produces for a given task [Moh18]. It represents the range of potential results or labels assigned to input samples. The structure of $\mathcal{O}$ varies depending on the task type.

Domain: A domain $\mathcal{M}$ refers to a specific area of knowledge or a field of expertise. It is defined as a pair $\mathcal{M} = \{\mathcal{I}, P(\mathcal{X})\}$, where $P(\mathcal{X})$ is the marginal probability distribution of $\mathcal{X}$ [Pan09]. This encompasses the particular characteristics, terminologies, and data patterns unique to that field. Domain knowledge is crucial for interpreting data correctly, selecting relevant features, and making meaningful predictions.

Model: A model is the core output of an ML algorithm. It embodies the knowledge acquired by the program and represents its understanding of the observed domain. A model can be mathematically defined as a function $f_\theta : \mathcal{I} \to \mathcal{O}$. The function f_θ maps input samples $\boldsymbol{x} \in \mathcal{X}$ to their corresponding outputs or predictions $\hat{\boldsymbol{y}} \in \mathcal{Y}$, where θ represents the parameters that define the model's behavior. For a given input $\boldsymbol{x}$, the model's output is referred to as the prediction $\hat{\boldsymbol{y}} = f_\theta(\boldsymbol{x})$.

Loss Function: A loss function or objective function $\mathcal{L}\colon \tilde{\boldsymbol{y}} \times \hat{\boldsymbol{y}} \to \mathbb{R}_+$ quantifies the discrepancy between the true label $\tilde{\boldsymbol{y}}$ and the prediction $\hat{\boldsymbol{y}}$ by a ML model [Goo16]. It maps the pair of true and predicted labels to a non-negative real number, measuring how well the model's predictions align with the actual values. The goal of optimizing a model is to minimize this loss function over the dataset.

Optimization: Optimization is the iterative process of refining a model's parameters to achieve the best possible performance. It involves searching through a vast space of potential solutions to find the optimal configuration that minimizes the discrepancy between the model's predictions and the actual outcomes. Mathematically, this is expressed as: $\theta^* = \underset{\theta}{\operatorname{argmin}}\ \mathcal{L}(f_\theta)$, where θ^* represents the optimal set of parameters.

Generalization: Generalization refers to a model's ability to perform well on previously unseen data drawn from the same domain as the data on which the ML model was optimized. A machine learning system can apply knowledge learned from a specific dataset to new, unfamiliar samples, making accurate predictions or decisions beyond the data used for optimization [Moh18].

Bias: Bias refers to the error introduced by approximating a complex domain by a simplified model. It is a systematic error that occurs due to incorrect assumptions in the ML algorithm [Gem92]. High bias causes an algorithm to miss relevant relations between features and target outputs.

Variance: Variance refers to the error introduced by a model's sensitivity to small fluctuations in the data. It measures how much the model's predictions would change if it were trained on a different dataset [Gem92]. High variance causes an ML algorithm to model the noise in the data instead of capturing characteristics.

Bias-Variance Trade-Off: The trade-off between bias and variance is a fundamental challenge in ML. Ideally, a model has low bias and low variance, allowing it to accurately capture the underlying patterns in the data and generalize well to new, unseen data. However, in practice, reducing bias often increases variance and vice versa, necessitating a balance between the two to minimize the total error [Gem92, Goo16].

Underfitting: Underfitting occurs if the parameter set θ for a ML model f_θ is insufficient to capture the underlying patterns in the data or when the ML algorithm is inadequately constructed. This situation typically arises when the model exhibits high bias and low variance [Moh18].

Overfitting: Overfitting happens if the ML models incorporate noise and intricate details from the dataset $\mathcal{X}$ into their acquired knowledge to such an extent that it negatively affects their performance on new, unseen data. This phenomenon typically arises when the number of parameters θ is much higher than needed for a task, resulting in an excessively complex model that leads to high variance and low bias [Moh18].

Data Augmentation: Data augmentation is a technique used to artificially increase the size of a dataset by creating modified versions of existing data. This involves transformations such as rotations, flips, and color adjustments for images, as well as other domain-specific modifications. Data augmentation helps improve the generalization ability of machine learning models and reduces overfitting by providing more diverse examples [Sho19].

Latent Space: A latent space, or latent feature space, is a vector space where data is represented in a transformed form, capturing its essential features in a compressed manner. This space enables the encoding of data into representations that highlight the intrinsic patterns and relationships within the data. Mathematically, it is defined by a transformation function $f_\theta : \mathbb{R}^n \to \mathbb{R}^m$, where $n > m$, mapping high-dimensional data to a lower-dimensional latent representation. Each point in this space corresponds to a unique vector representing the data's core structure, facilitating tasks such as clustering and dimensionality reduction by revealing patterns that are not immediately apparent in the original data space [LeC15].

Classification: In classification tasks, the ML system learns to assign samples $\boldsymbol{x}$ to discrete categorical annotations $\boldsymbol{y}$. The model constructs a decision boundary that partitions the latent feature space $\mathbb{R}^m$ into regions corresponding to distinct categories. Formally, this involves learning a hypothesis $f_\theta : \mathbb{R}^m \to \mathcal{C}$ where $\mathcal{C}$ is the set of target classes, with $\boldsymbol{y} \in \mathcal{C}$ being the predicted category for sample $\boldsymbol{x}$ [Mik08, Moh18].

Regression: In regression tasks, the ML system learns to estimate continuous numerical values $\boldsymbol{y} \in \mathbb{R}^m$ associated with input samples $\boldsymbol{x} \in \mathbb{R}^n$. The model constructs a functional relationship $f_\theta : \mathbb{R}^n \to \mathbb{R}^m$ that maps feature vectors to target vectors, where $m \geq 1$ determines the dimensionality of the predicted outputs [Mik08, Moh18].

Clustering: In clustering, patterns and characteristics within the input data are grouped into k distinct groups. The number of clusters k may be provided, but does not have to be, and unlike classification and regression, there are no labels $\boldsymbol{y}$. The ML system learns to categorize the input data based on inherent similarities [Mik08].

Dimensionality Reduction: Dimensionality reduction involves transforming a sample $\boldsymbol{x}$ from a high-dimensional space to a lower-dimensional space $\tilde{\boldsymbol{x}}$ while preserving the essential characteristics and information of the original sample. The model is optimized to obtain a useful mapping from the input space $\mathbb{R}^n$ to $\mathbb{R}^m$, where $n > m$, $f_\theta : \mathbb{R}^n \rightarrow \mathbb{R}^m$ [Moh18].

3.1.2 Learning Paradigms

Supervised: In an Supervised Learning (SL) task, experience is gained by modeling a specific domain $\mathcal{M}$. This is achieved by evaluating the model's output for specific samples and adjusting the ML system to ensure the model's mapping closely aligns with the desired output, as indicated by the labels and optimized using the loss function. Figure 3.1a shows a visualization of supervised learning. It is assumed that the input set $\mathcal{X}$ and the output set $\mathcal{Y}$ sufficiently describe the problem, allowing the ML system to generalize beyond the training data [Rus02]. Formally, $\mathcal{X}$ is sampled from the input space $\mathcal{I}$ and $\mathcal{Y}$ from the output space $\mathcal{O}$. The SL algorithm aims to learn the desired mapping of inputs to outputs of the domain $\mathcal{M}$ specified with the mapping $f_\theta \colon \mathcal{I} \rightarrow \mathcal{O}$ and optimized through the mapping for the given data $f_\theta \colon \mathcal{X} \rightarrow \mathcal{Y}$. For instance, if the task involves classifying images of cats and dogs, the domain $\mathcal{M}$ encompasses all possible images of cats and dogs, and the output space is represented as 1 for cats and 2 for dogs: $\mathcal{O} = \{1, 2\}$. Since obtaining all existing images of cats and dogs is impractical, the ML system must construct a model capable of solving the task with a limited set of inputs $\mathcal{X}$ and outputs $\mathcal{Y}$.

SL methods are called task-centric, as they model a mapping that is specifically crafted to solve a specific task [Tri22]. The function f_θ is typically regarded as a black box since the internal modeling of the data is not the focus, provided it satisfactorily approximates $\mathcal{I} \rightarrow \mathcal{O}$. Therefore, the generalization capability of

SL methods is inherently limited by the specific task and the data quality. If the provided samples $\mathcal{X}$ or labels $\mathcal{Y}$ do not adequately represent the respective domain $\mathcal{M}$, the ML system will fail to approximate the domain $\mathcal{M}$.

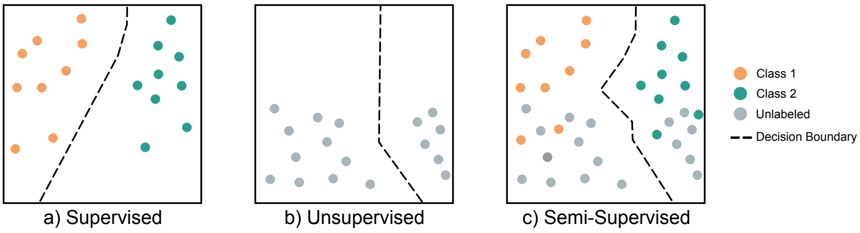


Figure 3.1: Visualization of the different types of classic ML algorithms. The dashed lines show a possible decision boundary a respective ML system might learn. a) Supervised learning, the decision boundary is learned through the two labeled classes. b) Unsupervised learning, the decision boundary is learned by clustering the unlabeled data points. c) Semi-supervised learning, the decision boundary is learned by combining the annotated classes and unlabeled data points together.

Unsupervised: In Unsupervised Learning (UL), information must be extracted by the ML algorithm solely from the given data $\mathcal{X}$, unlike in SL, where experience is acquired through a specifically defined output set [Rus02]. Figure 3.1b displays supervised learning visually. In UL tasks, the focus is uncovering internal structures within the input. The absence of an output set $\mathcal{Y}$ makes evaluating UL methods challenging. To provide guidance to the ML system, an abstract definition of the output space $\mathcal{O}$ can be constructed [Goo16]. For instance, in a common UL task like clustering, an abstract definition of $\mathcal{O}$ might specify that the number of groups to be identified is k, leading to a desired mapping $f : \mathcal{I} \to \{1, 2, \dots k\}$. Suppose the task involves categorizing a set of images into two groups based on an input set $\mathcal{X}$. In that case, the resulting output space $\mathcal{O}$ will depend entirely on the structures identified by the ML system in $\mathcal{X}$[1]. A

[1] Suppose $\mathcal{X}$ contains only pictures of cats and dogs. In that case, the system might learn to differentiate between the two species by recognizing features such as the typically larger snouts of dogs than cats. If $\mathcal{X}$ includes images of cats, dogs, and humans, the resulting model might separate humans from animals.

specific type of UL is Autoencoders (AEs), which are ML systems designed to learn efficient encodings of the provided data [Kra91]. For example, an AE could be trained to represent images in a compressed format. AEs can be viewed as a special case of dimensionality reduction [Kra91].

UL approaches are data-centric [Tri22]. They model mappings by identifying features within the data without the guidance of labels. The primary goal is to gain insights into the data's structure. Since no output set $\mathcal{Y}$ exists, unsupervised learning approaches are constrained only by the samples $\mathcal{X}$. If $\mathcal{X}$ does not adequately represent the data domain from which it was sampled, the ML system will be unable to model $\mathcal{M}$.

Semi-Supervised: Semi-Supervised Learning (Semi-SL) is positioned between SL and UL. The task to be addressed involves some labels $\mathcal{Y}$, but not for all samples in $\mathcal{X}$. Typically, the quantity of unlabeled samples drastically exceeds that of labeled ones. Semi-SL methods are generally adopted in fields where labeling is costly, incorporating unlabeled data into the training process [Rus02]. In many fields, obtaining labeled data can be challenging, especially when annotation requires specialized expertise. For example, in biomedical image processing, labeling often relies on domain experts with deep knowledge, such as medical professionals. Because these experts have limited time to devote to annotation tasks, the number of labeled samples is often scarce. Semi-SL methods optimize the mapping of the labeled segment of the dataset $f_\theta : \mathcal{X}_\mathrm{l} \rightarrow \mathcal{Y}_\mathrm{l}$, while also leveraging and learning the underlying structure of the unlabeled portion $\mathcal{X}_\mathrm{u}$. Figure 3.1c visualizes how Semi-SL finds decision boundaries.

Semi-SL methods are considered both task-centric and data-centric. They aim to model $\mathcal{M}$ by utilizing the underlying structure of the data with the unlabeled portion $\mathcal{X}_\mathrm{u}$ of the dataset $\mathcal{X}$ while also addressing a specific task defined by the labeled portion $\mathcal{X}_\mathrm{l}$.

Self-Supervised: Self-Supervised Learning (Self-SL) is a subclass of UL. However, distinctive features by Self-SL clearly differentiate it from the general UL scenario. In UL, a given task is solved, and a mapping from unlabeled data is learned as a final objective. Conversely, Self-SL is employed as a so-called pretext task to bolster a subsequent SL method, referred to as the downstream task. Self-SL is not intended for independent use but is integrated into a general learning framework. The pretext task is formulated as an SL challenge, which is solved exclusively with unlabeled data. To be able to achieve this, pseudo labels are utilized to construct an output set, which is automatically generated from the available unlabeled data [Owe18, Jin20]. Within computer vision, a potential pretext task is to produce suitable vector encodings for a collection of images. In this scenario, pseudo labels are created by altering the appearance of existing images to obtain two perspectives of the same image. With such a pseudo label, the given pretext task is to ensure that the original and modified images are encoded as similarly as possible. The learned encodings are then applied to downstream tasks that benefit from such encoding vectors. The concept of Self-SL is visualized in Figure 3.2, featuring the aforementioned pretext task. In the pretext task, a sample $\boldsymbol{x}$ is modified in two ways, generating two views $\boldsymbol{x}'$ and $\boldsymbol{x}''$, which serve as the pseudo labels. These views are mapped by the ML system f_θ to the respective predicted labels $\boldsymbol{y}'$ and $\boldsymbol{y}''$, called the encodings. The ML system f_θ is optimized through the loss function $\mathcal{L}$, ensuring that $\boldsymbol{y}'$ and $\boldsymbol{y}''$ are similar according to a specified criterion. Knowledge of the unlabeled dataset is established through the model obtained from the pretext task, which is then transferred to the downstream task. The downstream task is trained using a set of samples $\mathcal{X}$ and annotations $\mathcal{Y}$ from the same (or similar) domain as the pretext task. While the downstream task can encompass any ML challenge, it is typically trained as a traditional SL task. This is accomplished by minimizing the discrepancy between the true label $\boldsymbol{y}$ and the predicted label $\hat{\boldsymbol{y}}$ using the SL loss function $\mathcal{L}'$. As the machine learning model f'_θ of the downstream task has acquired knowledge from the pretext task, fewer labeled samples are required to approximate the domain $\mathcal{M}$.

A solution to the costly labeling problem is proposed by Self-SL through the independent generation of a supervised pretext task [LeC21]. Pretext tasks are similar to UL, and when combined with a downstream task, strong parallels to Semi-SL can be observed. However, due to the shared knowledge across the

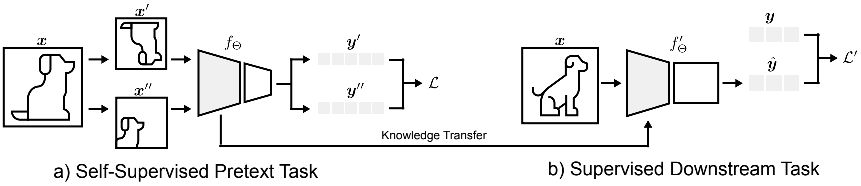


Figure 3.2: Concept of the Self-SL framework. a) An exemplary pretext task that transforms an input image x into two views $\boldsymbol{x}'$ and $\boldsymbol{x}''$. The ML system f_θ maps $\boldsymbol{x}'$ and $\boldsymbol{x}''$ to the representations $\boldsymbol{y}'$ and $\boldsymbol{y}''$, respectively, which are optimized to be similar by an objective function $\mathcal{L}$. Parts of f_θ are transferred into the ML system f'_θ of the downstream task. b) The downstream task is optimized via SL with sample label pairs $\boldsymbol{x}$ and $\boldsymbol{y}$. f'_θ is optimized with the objective function $\mathcal{L}'$ that minimizes the error between the prediction $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}$.

pretext and downstream tasks, the unlabeled data is not viewed as a distinct task, as in Semi-SL. Furthermore, a model derived from a well-formulated pretext task can be applied to a wide array of downstream tasks, positioning it as a general model of the data it was trained on. Nevertheless, the crux of Self-SL lies in the pretext task and its associated pseudo labels, rendering the learned model far from self-sufficient. As the central component of Self-SL, the pretext task still necessitates substantial human guidance and invariably contains bias [LeC21].

3.1.3 Object Detection

Detection: In computer vision, detection refers to the task of locating objects of interest. Formally, given an input image $\boldsymbol{x} \in \mathbb{N}^{c \times h \times w}$, where c, h, and w represent the number of channels, height, and width respectively, the goal is to output a set of bounding boxes $\mathcal{B} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \dots\}$, where each $\boldsymbol{b}_i = [x_i, y_i, w_i, h_i]^\mathrm{T}$ represents the coordinates, width, and height of a detected object. Figure 3.3a shows an example of bounding box detection. Additionally, detection often includes assigning class labels $\mathcal{C} = \{l_1, l_2, \dots\}$ to each bounding box, where $l_i \in 1, 2, \dots, k$ for k predefined object classes [Sze22].

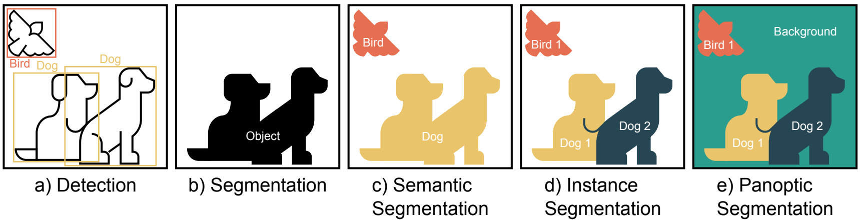


Figure 3.3: Example for different detection types. a) Detection with bounding boxes around every detected object. b) Segmentation, where the dogs are defined as the object of interest. c) Semantic segmentation illustrated with the two classes, dog and bird, differentiated from each other. d) Instance segmentation with each instance delineated. e) Panoptic segmentation with countable objects, dog, bird, and the background.

Segmentation: Segmentation refers to the process of dividing an image into multiple distinct regions or segments. Figure 3.3b displays a segmentation example. This task goes beyond simply detecting objects, aiming to precisely delineate their boundaries at the pixel level. Segmentation algorithms partition an image by grouping pixels with common attributes such as color, intensity, or texture. The goal is to transform the raw image data into a more meaningful representation that facilitates further analysis and understanding [Gir15b].

Semantic Segmentation: Semantic segmentation is the challenge of partitioning an image into semantic components. Figure 3.3c shows an example of semantic segmentation. It is a subset of segmentation, where each pixel in an image is classified as belonging to a specific class. Consider an RGB image $\boldsymbol{x} \in \mathbb{N}^{c \times h \times w}$. Semantic segmentation maps $\boldsymbol{x}$ to a label map $\boldsymbol{y} \in \mathbb{N}^{h \times w}$, where each element of $\boldsymbol{y}$ corresponds to a pixel in $\boldsymbol{x}$ and is classified into one of the predefined classes $k = \{1, 2, \ldots, n\}$. To achieve this, a ML model learns a function $f_\theta \colon \mathbb{N}^{c \times h \times w} \rightarrow \mathbb{N}^{h \times w}$ that performs the mapping $\boldsymbol{x} \mapsto \boldsymbol{y}$ with $\forall y_{u,v} \in \{0, 1, \ldots, n\}$ [Sze22].

Instance Segmentation: Instance segmentation is a fine-grained form of segmentation that classifies each pixel into a specific class and differentiates between distinct objects of the same class. Figure 3.3d illustrates an example of instance segmentation. Given an input image $\boldsymbol{x} \in \mathbb{N}^{c \times h \times w}$, the goal is to produce a set of k binary masks $\mathcal{M} = \{\boldsymbol{m}_1, \boldsymbol{m}_2, \dots, k\}$, where each mask $\boldsymbol{m}_i \in \mathbb{N}^{h \times w}$ corresponds to a single object instance. This task combines object detection and semantic segmentation objectives by identifying the class and the precise boundary of each object instance. Formally, instance segmentation can be described by a function $f_\theta \colon \mathbb{N}^{c \times h \times w} \to \mathbb{N}^{k \times h \times w}$ with $\boldsymbol{x} \mapsto \mathcal{M}$, that maps the input image to a tensor representing the instance masks and their respective class labels.

Panoptic Segmentation: Panoptic segmentation is a comprehensive image segmentation technique [Kir19]. It assigns each pixel a semantic label and, for countable objects, a unique instance ID. This approach provides a holistic understanding of the scene by classifying objects and background elements and distinguishing between individual instances of the same class. Unlike instance segmentation, which focuses solely on identifying and delineating individual object instances, panoptic segmentation classifies every pixel in the image, including those belonging to background or *stuff* regions, such as roads or sky. This broader contextual understanding enhances scene analysis and facilitates higher-level reasoning tasks.

3.1.4 Model Uncertainty

In ML, uncertainty represents the degree of doubt or lack of confidence in a model's predictions, which is essential for developing reliable systems, particularly in critical domains such as healthcare or autonomous driving. The uncertainty of a model is closely linked to its generalization capabilities, as models prone to overfitting typically exhibit high uncertainty [Vie22]. Within ML, uncertainty comprises two primary components: aleatoric and epistemic uncertainty [Ken17].

Aleatoric Uncertainty: Aleatoric uncertainty, or statistical uncertainty, describes the inherent variability in experimental outcomes due to random effects. This type of uncertainty cannot be reduced by collecting more data or improving the model. It represents the natural variability in the domain being modeled. An example of aleatoric uncertainty in computer vision is the interpretation of low-resolution images. Even with perfect knowledge of the image capture conditions and advanced models, low-resolution images inherently contain ambiguity due to a lack of detail. This ambiguity leads to unpredictable outcomes, reflecting the inherent randomness and variability in the data itself [Hül21].

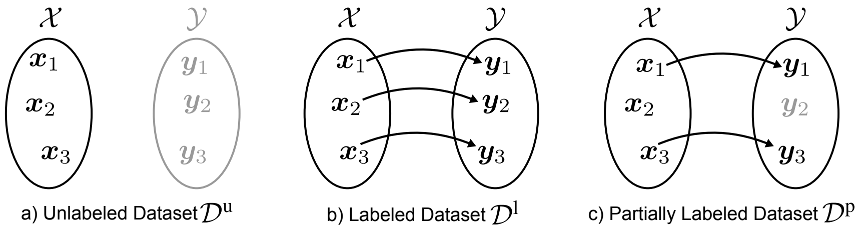
Epistemic Uncertainty: Epistemic uncertainty, also known as model uncertainty, stems from a lack of knowledge of the model. It is the type of uncertainty that can be reduced by acquiring more information or improving the used model. It is particularly critical in areas where the training data might not fully represent all possible scenarios. An example of epistemic uncertainty is predicting the weather. Weather predictions are uncertain due to incomplete knowledge of the weather system and limitations of employed models. This uncertainty can be reduced by improving data collection and weather models [Hül21].

3.2 Datasets

As data is fundamental for ML, this section outlines the general characteristics of datasets.

3.2.1 Dataset Types

Unlabeled dataset: Expanding upon the general definition of datasets in Section 3.1.1, a dataset $\mathcal{D}^{\mathrm{u}} = \{\mathcal{X}\}$ without an output set $\mathcal{Y}$ is called unlabeled (see Figure 3.4a). In other words, this dataset comprises solely the input data $\mathcal{X}$ without any corresponding labels or target values $\mathcal{Y}$. Such datasets are commonly used in UL or Self-SL tasks where the goal is to identify patterns or structures within the data without predefined labels. Examples include clustering and dimensionality reduction tasks.



$\mathcal{X}$
$\mathcal{Y}$
$\boldsymbol{x}_1$
$\boldsymbol{x}_2$
$\boldsymbol{x}_3$
y_1
y_2
y_3
a) Unlabeled Dataset $\mathcal{D}^{\mathrm{u}}$
b) Labeled Dataset $\mathcal{D}^{\mathrm{l}}$
c) Partially Labeled Dataset $\mathcal{D}^{\mathrm{p}}$

Figure 3.4: Depiction of the three dataset types as function diagrams. a) The input set $\mathcal{X}$ of $\mathcal{D}^{\mathrm{u}}$ lacks an output set $\mathcal{Y}$, though a potential $\mathcal{Y}$ is shown in grey. b) $\mathcal{D}^{\mathrm{l}}$ includes a mapping from each $\boldsymbol{x}_i \in \{1, 2, 3\}$ to a corresponding $\boldsymbol{y}_i$. c) $\mathcal{D}^{\mathrm{p}}$ maps $\boldsymbol{x}_1$ / $\boldsymbol{x}_3$ to $\boldsymbol{y}_1$ / $\boldsymbol{y}_3$, but $\boldsymbol{x}_2$ has no corresponding output in $\mathcal{Y}$ (a potential $\boldsymbol{y}_2$ is shown in gray).

Labeled dataset: A dataset $\mathcal{D}^{\mathrm{l}} = \{\mathcal{X}, \mathcal{Y}\}$ is termed labeled if for every $\boldsymbol{x}_i \in \mathcal{X}$ there is a corresponding $\boldsymbol{y}_i \in \mathcal{Y}$, meaning $\forall \boldsymbol{x}_i \in \mathcal{X},\ \exists \boldsymbol{y}_i \in \mathcal{Y}$ (see Figure 3.4b). This means that for each input data point $\boldsymbol{x}_i$, there needs to be an associated label $\boldsymbol{y}_i$, which is crucial for SL tasks. Labeled datasets are used to train models to make predictions or classifications based on the input data. Examples include image classification datasets where each image has a corresponding label indicating the object in the image.

Partially labeled dataset: A dataset $\mathcal{D}^{\mathrm{p}} = \{\mathcal{X}, \mathcal{Y}\}$ is called partially labeled if there exists at least one $\boldsymbol{x}_i \in \mathcal{X}$ without a corresponding $\boldsymbol{y}_i \in \mathcal{Y}$, meaning $\exists \boldsymbol{x}_i \in \mathcal{X},\ \nexists \boldsymbol{y}_i \in \mathcal{Y}$, but there is also at least one labeled $\boldsymbol{x}_i \in \mathcal{X}$, meaning $\exists \boldsymbol{x}_i \in \mathcal{X},\ \exists \boldsymbol{y}_i \in \mathcal{Y}$ (see Figure 3.4c). This dataset type is often used in Semi-SL or Self-SL, where the model is trained using labeled and unlabeled data. The presence of some labeled data aids the learning process, while the unlabeled data can enhance the model's generalization by utilizing additional information.

Dataset-split: When an ML system is tasked with solving a challenge, it is usually provided with only a subset $\mathcal{D}_{\text{train}} \subset \mathcal{D}$ of the entire data, known as the training dataset. The remaining samples form the test dataset $\mathcal{D}_{\text{test}} = \mathcal{D} \setminus \mathcal{D}_{\text{train}}$ used for performance evaluation. This division enables the model to learn from the training data and be evaluated on the test data to assess its performance. The split is essential for preventing overfitting and ensuring the model can generalize well to unseen data. Additionally, a validation set $\mathcal{D}_{\text{val}} \subset \mathcal{D}_{\text{train}}$ is often used during training to fine-tune the model's hyperparameters and prevent overfitting.

3.2.2 Labeling

Homogenously labeled datasets: Homogenously labeled datasets are those where all samples are annotated consistently, using the same labeling scheme or classification system across the entire dataset [Sch22d]. This uniformity ensures that every sample is treated equally and can be directly compared or analyzed. Formally, a homogenously labeled dataset is defined as $\mathcal{D}^{\text{ho}} = \{\mathcal{X}, \mathcal{Y}\}$, where all $\boldsymbol{y}_i \in \mathcal{Y}$ are drawn from a single, predefined set of categories $\mathcal{C} \subseteq \mathcal{Y}$, such that $\boldsymbol{y}_i \in \mathcal{C}$ for all i. This consistency in labeling is usually required for the training of ML models, as most systems require inputs to be consistent. For example, if a dataset is used for multi-class semantic segmentation. In that case, each image in the dataset will have annotations for all the classes of interest, ensuring complete and consistent labeling across the entire dataset. This means that if the dataset includes classes like *car*, *pedestrian*, and *road*, every image will be annotated for these three classes, even if some images do not contain all of them.

Hetereogenously labeled datasets: Heteregenously labeled datasets contain samples annotated using different labeling schemes or classification systems [Sch22d]. This diversity in labeling may arise from various factors, such as differences in annotators, evolving labeling standards, or the combination of datasets from multiple sources. A heterogeneously labeled dataset is defined as $\mathcal{D}^{\text{he}} = \{\mathcal{X}, \mathcal{Y}\}$. The labels $\boldsymbol{y}_i \in \mathcal{Y}$ are drawn from multiple, potentially incompatible categories. Some label $\boldsymbol{y}_j$ may be drawn from a set of corresponding annotations $\mathcal{C}_j \subseteq \mathcal{Y}$, and some other label $\boldsymbol{y}_k$ from $\mathcal{C}_k \subseteq \mathcal{Y}$. Here, $\mathcal{C}_j$ and $\mathcal{C}_k$ may contain some identical classes or overlap completely, but may also be without any intersections. This inconsistency in labeling poses unique challenges for ML, as it necessitates methods to reconcile or effectively utilize the diverse labeling schemes. For instance, some images in a biomedical dataset might be annotated for specific organs. In contrast, others might be annotated for different organs, and some might lack annotations for certain organs altogether. Working with heterogeneously labeled datasets often requires additional preprocessing steps or specialized ML techniques. While heterogeneously labeled datasets present challenges, they can also offer opportunities. They may provide a more comprehensive view of the problem domain, capture different aspects of the data, and potentially lead to more robust and versatile models if handled correctly. However, avoiding biases and ensuring fair evaluation across different labeling schemes is essential.

3.3 Deep Learning

The development of neural networks has gone through multiple major phases. Initially, researchers focused on biologically inspired models that mimicked brain function. Today, large and complex DL networks are widely used, which are sophisticated and extensive Neural Networks (NNs) that have drastically advanced the field [Goo16]. While the initial idea of better understanding the human brain remains evident in modern NN techniques, they are primarily used more practically as task-solving tools nowadays. This section addresses foundations and concepts related to DL.

3.3.1 Neural Networks

Feed Forward Networks: A neuron, the basic unit of the nervous system, processes and transmits information via electrochemical signals. It consists of dendrites that receive signals, a cell nucleus that integrates them, and an axon that sends out impulses if the signals are strong enough [Kan00]. Artificial Neurons (ANs) mimic biological neurons. They receive inputs $\boldsymbol{x} = [x_1, x_2, \dots x_m]^{\mathrm{T}}$, a bias b, and have weights $\boldsymbol{\omega} = [\omega_1, \omega_2, \dots \omega_m, \omega_{m+1}]^{\mathrm{T}}$. These weights $\boldsymbol{\omega}$ are the learnable parameters θ that define the general ML model $f_\theta()$, as introduced in Section 3.1.1. An activation function ϕ is used to represent the electrochemical impulses sent through the axon, and the final output is calculated as $y = \phi(\sum_{i=1}^{m} \omega_i x_i + \omega_{m+1})$ [Goo16]. The weights $\boldsymbol{\omega}$ are optimized for a given task. A visual comparison between the biological neuron and ANs is shown in Figure 3.5.

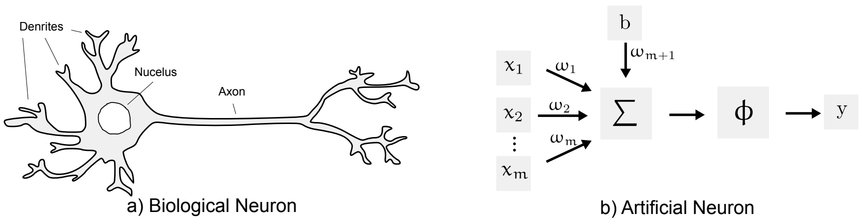


Figure 3.5: a) A biological neuron receives input signals through dendrites, processes them in the nucleus, and transmits impulses via the axon. b) An Artificial Neuron (AN) receives inputs $x_1, x_2, \dots x_m$, combines them with weights $\omega_1, \omega_2, \dots \omega_m$ through a weighted operation, incorporates an offset (bias) b, and processes the result through an activation function ϕ to produce the output y.

Multiple ANs can be interconnected to form an Artificial Neural Network (ANN). ANNs are classified into two types: Feed Forward Networks (FFNs) and Recurrent Neural Networks (RNNs). RNNs allow cycles, exhibiting temporal dynamic behavior. FFNs are directed acyclic graphs where information flows in one direction. This work focuses on FFNs. Details about RNNs are available in introductory literature on ANNs [Goo16].

Figure 3.6 shows a FFN with one hidden layer. If every AN is connected to every AN of the previous layer, the network is called a fully connected FFN. The weight matrices $\boldsymbol{\omega}^j$ and $\boldsymbol{\omega}^{j+1}$ connect layers, where $\boldsymbol{\omega}^j_{i,k}$ links the i-th AN of layer $(j-1)$ to the k-th AN of layer j. $\phi_{j,h}$ is the activation function of the h-th AN in layer j. FFNs are functions, hence the outputs y_i are directly calculable. FFNs are optimized using gradient-based approaches, typically backpropagation with gradient descent, to calculate gradients of the loss function $\mathcal{L}$ w.r.t each weight ω [Goo16].

Convolutional Neural Networks: Convolutional Neural Networks (CNNs) are a special kind of FFN, which are mostly applied to processing imaging data. Additionally to the typical building blocks of ANNs, are CNNs constructed of

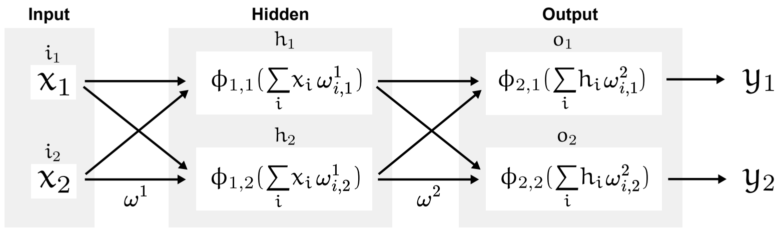

Input
Hidden
Output
i_1
x_1
i_2
x_2
ω^1
h_1
$\phi_{1,1}(\sum_i x_i \omega^1_{i,1})$
h_2
$\phi_{1,2}(\sum_i x_i \omega^1_{i,2})$
ω^2
o_1
$\phi_{2,1}(\sum_i h_i \omega^2_{i,1})$
o_2
$\phi_{2,2}(\sum_i h_i \omega^2_{i,2})$
y_1
y_2

Figure 3.6: A FFN with one hidden layer. The input has two ANs $\mathbf{i_1}$ and $\mathbf{i_2}$ with values $\mathbf{x_1}$ and $\mathbf{x_2}$. The weight matrix $\boldsymbol{\omega}^1$ connects the input layer to the hidden layer, which has two ANs $\mathbf{h_1}$ and $\mathbf{h_2}$. The hidden layer connects to the output layer via $\boldsymbol{\omega}^2$, and the output layer has two ANs $\mathbf{o_1}$ and $\mathbf{o_2}$, corresponding to output variables y_1 and y_2.

two types of layers: convolution and pooling layers. Convolution layers use kernel matrices Ω to detect feature maps by convoluting the kernel over an input image or a feature map. The convolution g for an image $\mathbf{I}$ is defined as:

$$g(u,v) = \Omega * I(u,v) = \sum_{du=-a}^{a} \sum_{dv=-b}^{b} \Omega(du,dv) I(u+du, v+dv)^2, \qquad (3.1)$$

where a and b are the width and height of the kernel, respectively [**?**]. For example, the Sobel operator [Ans17] along the x-axis, a filter used for edge detection, is defined by the kernel:

$$\Omega = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}. \qquad (3.2)$$

This kernel approximates the partial derivative of each pixel in the input image. CNNs are optimized to learn such filters. Figure 3.7 shows an example of the Sobel operator.

In CNN networks, pooling layers are used to reduce the dimensionality of learned feature maps. By condensing these features, pooling layers enhance the network's

[2] It must be ensured that the kernel does not convolute out of the boundaries of the image.

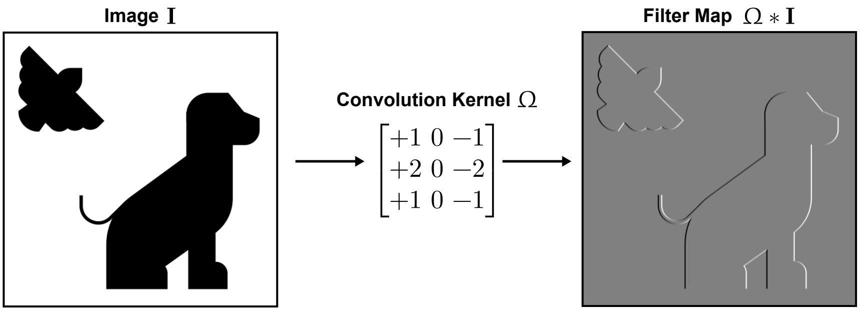


Image $\mathbf{I}$
Convolution Kernel Ω
$\begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}$
Filter Map $\Omega * \mathbf{I}$

Figure 3.7: The Sobel filter applied to an example image. The input image $\mathbf{I}$ is convolved with the Sobel operator Ω to yield the filter map $\Omega * \mathbf{I}$.

generalization ability and decrease computational requirements. Max- or average-pooling is typically applied in CNNs [Goo16]. These layers do not have learnable weights. Figure 3.8 shows an example of a CNN with three convolutional and two pooling layers. Following the second pooling layer, a fully connected layer is employed to utilize the learned feature maps for solving a given task. Unlike conventional FFNs, where every output AN interacts with every other AN, CNNs use smaller filter kernels than the input image, allowing small but meaningful features to be detected independently of input size [Goo16]. Furthermore, weight sharing is employed, meaning the same weights are used multiple times across the input image, rather than just once [Goo16]. Lastly, the equivariance of convolutions is exploited, ensuring that learned filters preserve the algebraic structure of transformations. For instance, a slight translation of an image will result in a similarly translated feature mapping [Goo16].

Fully Convolutional Networks (FCNs) are a type of CNN that solely employ convolutional operations and lack fully connected layers entirely, further enhancing the benefits of CNNs and accelerating both training and inference times [Lon15]. FCNs are usually used for tasks requiring pixel-level image evaluation, like segmentation tasks (see Section 3.1.3). For segmentation, Encoder-Decoder Networks (EDNs) are the most popular choice, which consist of an encoder (contracting path) and a decoder (expanding path). The encoder resembles a typical CNN with convolution and pooling layers. The decoder then reconstructs

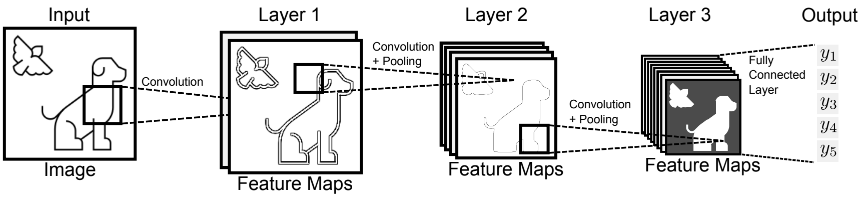


Figure 3.8: Architecture of an exemplary CNN. The input image undergoes multiple convolutions to generate initial feature maps in Layer 1, which are then convoluted and pooled to reduce size in Layer 2. This is followed by another convolution and pooling operation (Layer 3). Finally, a Fully Connected Layer generates an output vector y that can be optimized to solve a given task. Adapted from [Ret25b].

a detailed output map by applying additional convolutional layers and replacing pooling layers with upsampling layers [Ron15].

Vision Transformers: Vision Transformers (ViTs) [Dos20] adapt the transformer architecture, initially designed for natural language processing, to computer vision tasks. For a comprehensive introduction to transformers, refer to [Tun22]. ViTs operate by dividing an image into fixed-size patches, which are linearly embedded and treated as a sequence of tokens. These tokens and positional embeddings are fed into a standard transformer encoder. The self-attention mechanism allows ViTs to capture global dependencies across the entire image. While ViTs have shown remarkable performance in various vision tasks [Jee22], they typically require larger datasets for training compared to CNNs. This is due to their lack of built-in inductive biases, such as translation equivariance, inherent in CNNs [Akk24, Pap24]. Consequently, the data-hungry nature of ViTs limits their applicability in scenarios with limited data availability [Liu21a].

3.3.2 Self-Supervised Learning

Self-SL techniques in DL can be categorized into three main types: generative, generative-discriminative, and discriminative methods.

Generative approaches utilize EDNs and assess their quality using a loss function based on reconstruction error to develop a pre-trained backbone for downstream tasks [Vin08, Rez14]. Within the generative approaches, Masked Autoencoders (Masked-AEs) have gained attention for their simplicity and effectiveness. They involve masking a considerable portion of the input data and reconstructing it using EDNs. This process encourages the model to capture spatial dependencies and contextual information within the data [He22].

Generative-discriminative (adversarial) techniques employ Generative Adversarial Networks (GANs) [Goo14] to learn representations [Don16, Dum16, Don19]. While generative-discriminative methods can achieve state-of-the-art results, they suffer from the inherent challenges of GAN-based approaches: adversarial training is complex, computationally intensive, and requires extensive datasets [Gri20].

Discriminative methods directly apply metrics to the representations and are the prevailing paradigm in Self-SL [Nor16, Che20b, Che20a, Gri20, Mis20, Zbo21]. Among discriminative approaches, Contrastive Learning (CL) is the most popular. All CL techniques share a common architectural framework. They utilize a set of permissible image transformations known as augmentations $\mathcal{T}$ for computer vision datasets. Two distinct augmentation operations $t\sim\mathcal{T}$ and $t'\sim\mathcal{T}$ are applied to a data sample $\boldsymbol{x}$ from a dataset $\mathcal{X}$, generating two views, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ (see Figure 3.9). These views are then processed by the base encoders f_θ and f_ξ,

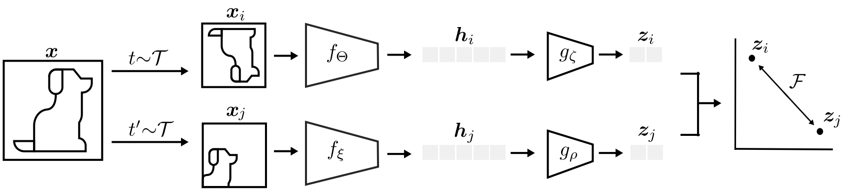


Figure 3.9: The general CL framework involves a sequence of transformations $t\sim\mathcal{T}$ and $t'\sim\mathcal{T}$ which generate two distinct views, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, from the original image $\boldsymbol{x}$. These views are processed through the encoders f_θ and f_ξ to derive the encodings h_i and h_j. Subsequently, the projection heads g_ζ and g_ρ convert these encodings into embeddings z_i and z_j. These embeddings are then input into the similarity function $\mathcal{F}$, which computes the distance between them, denoted as $d(\boldsymbol{z}_i, \boldsymbol{z}_j)$.

which learn vector representations of the images. Consequently, two encodings, $\boldsymbol{h}_i = f_\theta(\boldsymbol{x}_i)$ and $\boldsymbol{h}_j = f_\xi(\boldsymbol{x}_j)$, are obtained. Given that these encodings are typically high-dimensional, which can lead to complications due to the curse of dimensionality [Bel66], two NN-based projection heads g_ζ and g_ρ are used to map the embeddings into a lower-dimensional space producing two embeddings, $\boldsymbol{z}_i = g_\zeta(\boldsymbol{h}_i)$ and $\boldsymbol{z}_j = g_\rho(\boldsymbol{h}_j)$. The vectors $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ are evaluated using a objective function $\mathcal{F}$ that produces a similarity score $s \in \mathbb{R}_+$:

$$\mathcal{F} : \boldsymbol{z}_i \times \boldsymbol{z}_j \mapsto s. \tag{3.3}$$

The function $\mathcal{F}$ is designed to generate high scores for similar pairs ($\boldsymbol{z}_i$ and $\boldsymbol{z}_j$) and low scores for pairs that are dissimilar (any other image) [LeC05]. In the context of DL, $\mathcal{F}$ is typically implemented using distance-derived metrics, such as the cosine similarity [Gom13] or Euclidean distance. To avoid the issue of model collapse, where $\mathcal{F}$ would consistently produce a distance of 0, it is essential to train the model with a diverse set of dissimilar samples.

3.3.3 Deep Learning Architectures

Residual Neural Networks (ResNets): The encoder of an EDN is typically referred to as the backbone [Amj20]. One widely used backbone in EDNs is the family of Residual Neural Networks (ResNets) [He16]. ResNets incorporate residual blocks featuring shortcut connections, which allow inputs to bypass multiple layers directly. A standard layer in a ResNet can be represented as $f(\boldsymbol{x}) =$

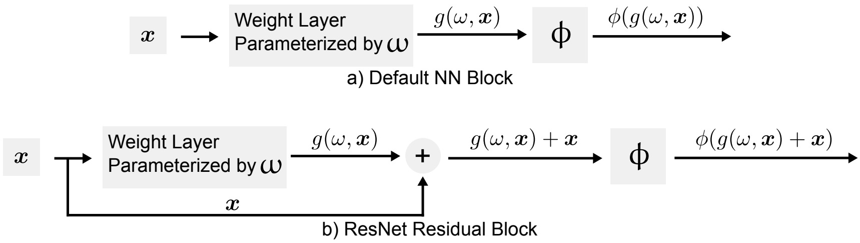

$\boldsymbol{x}$
Weight Layer Parameterized by ω
$g(\omega, \boldsymbol{x})$
ϕ
$\phi(g(\omega, \boldsymbol{x}))$
a) Default NN Block
$\boldsymbol{x}$
Weight Layer Parameterized by ω
$g(\omega, \boldsymbol{x})$
+
$g(\omega, \boldsymbol{x}) + \boldsymbol{x}$
ϕ
$\phi(g(\omega, \boldsymbol{x}) + \boldsymbol{x})$
$\boldsymbol{x}$
b) ResNet Residual Block

Figure 3.10: Example of a default NN block and a ResNet residual block. The default NN block displayed in a) processes the input $\boldsymbol{x}$ with a weight layer parametrized by ω ($g(\omega, \boldsymbol{x})$ and applies the activation function ϕ to obtain the output $\phi(g(\omega, \boldsymbol{x}))$. The residual block does the same but additionally adds the input to the output of the weight layer ($g(\omega, \boldsymbol{x}) + \boldsymbol{x}$) to obtain the output as $\phi(g(\omega, \boldsymbol{x}) + \boldsymbol{x})$.

$\phi(g(\omega, \boldsymbol{x}) + \boldsymbol{x})$, enabling the network to learn residual functions and facilitating the training of deeper networks [He16]. Figure 3.10 presents a comparison between a default NN block and a ResNet residual block. ResNets are available in various depths, including ResNet-18, ResNet-34, ResNet-50, ResNet-101, and ResNet-152. Further details about the architecture and its variants are available in Appendix A.3.1.

U-Net: The U-Net architecture, a type of FCN, is specifically designed for biomedical image segmentation and is particularly effective when working with limited datasets [Ron15]. Figure 3.11 illustrates the U-Net. This architecture is

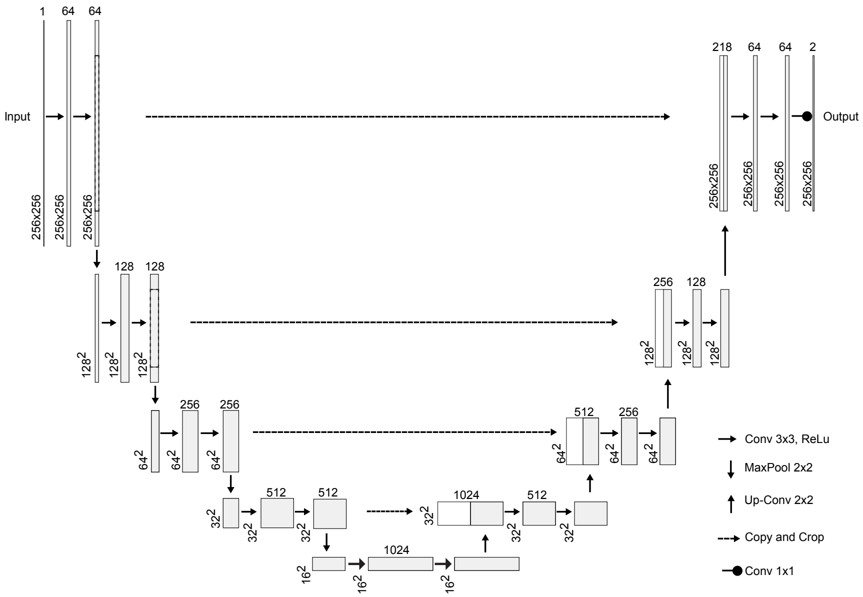


Figure 3.11: The default U-Net with five layers [Ron15]. Convolutional layers utilize padding to preserve the dimensions of the feature map. The architecture follows a U-shape with a contracting encoder path and an expansive decoder path. Details can be found in Appendix A.3.2. Adapted from [Ron15].

an EDN and is proficient in delivering precise pixel-level classifications. U-Net effectively identifies complex patterns and structures by systematically reducing the resolution of the feature maps. The U-Net consists of an encoder and a decoder, with skip connections that preserve high-resolution information. The backbone of a U-Net can be replaced by other FCNs, such as variations of the ResNet architecture, by establishing skip connections. For instance segmentation tasks, the U-Net can be adapted with post-processing steps to produce instance segmentation masks [Sch20]. Details about the U-Net architecture are available in Appendix A.3.2.

Mask R-CNN: Mask R-CNN [He17] is an advanced architecture for object detection and instance segmentation that extends the popular Faster R-CNN [Ren15] by adding a branch for predicting segmentation masks. Detailed descriptions of the Faster R-CNN architecture are available in the literature [Ren15]. A visualization of the overall Mask R-CNN architecture, including its components and loss calculations, is presented in Figure 3.12. Mask R-CNN employs an Feature Pyra-

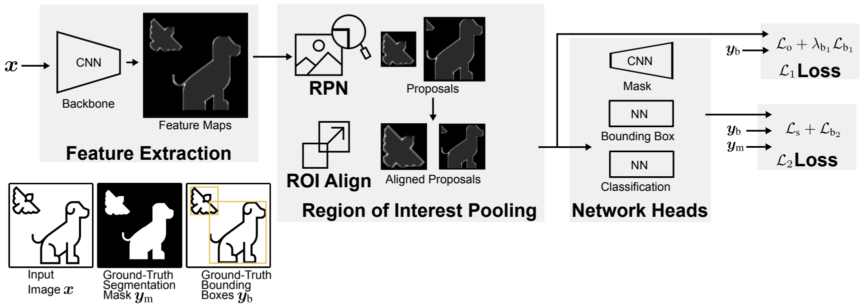


Figure 3.12: The Mask R-CNN architecture. The input $\boldsymbol{x}$ is fed into the CNN backbone to produce feature maps. The Region Proposal Network (RPN) generates Region of Interest (ROI) proposals from the feature maps that are aligned by ROI align. The network heads then use the aligned proposals to produce individual predictions. The ground truth segmentation masks $\boldsymbol{y}_{\mathrm{m}}$ and bounding boxes $\boldsymbol{y}_{\mathrm{b}}$ are used with the predictions to calculate the $\mathcal{L}_1$ ($\mathcal{L}_{\mathrm{o}} + \lambda_{\mathrm{b}_1}\mathcal{L}_{\mathrm{b}_1}$) and $\mathcal{L}_2$ ($\mathcal{L}_{\mathrm{s}} + \mathcal{L}_{\mathrm{b}_2}$) losses.

mid Network (FPN) [Lin17] to enhance feature extraction through multi-scale feature representation, which is crucial for detecting objects of varying sizes. The FPN leverages lateral connections within a ResNet backbone to build hierarchical feature pyramids, preserving fine-grained information at multiple scales. These multi-scale features are then used by the RPN, which generates region proposals indicating potential object locations. After proposal generation, non-maximum suppression filters redundant proposals. Subsequently, ROI aligns crops and resizes these proposals to a fixed spatial dimension (usually 7×7 pixels), enabling standardized processing by subsequent network heads. These heads include the bounding box regression head, which refines bounding box coordinates; the classification head, which predicts object class labels; and the mask head, which generates pixel-wise binary segmentation masks. The overall loss function of Mask R-CNN combines two main components, $\mathcal{L}_1$ and $\mathcal{L}_2$, each addressing specific aspects of the model's performance. Component $\mathcal{L}_1$ includes objectness classification ($\mathcal{L}_\mathrm{o}$) and bounding box regression ($\mathcal{L}_{\mathrm{b}_1}$). Component $\mathcal{L}_2$ refines bounding box regression ($\mathcal{L}_{\mathrm{b}_2}$) and computes segmentation mask loss ($\mathcal{L}_\mathrm{s}$). The total loss is calculated as: $\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2$. Detailed descriptions of the feature pyramid construction, region proposal generation, ROI alignment procedures, and the mathematical formulations of individual loss components can be found in Appendix A.3.3.

ConvNeXt: ConvNeXt [Woo23] modernizes the classic ResNet design by incorporating insights from ViTs while maintaining the efficiency of CNNs. It introduces both high-level structural (macro) and low-level (micro) changes, enhancing performance and efficiency and narrowing the gap to ViTs [Liu22]. The macro design changes in ConvNeXt include adopting transformer-inspired block distributions and a stem block for efficient downsampling. It also incorporates depthwise convolutions and an inverted bottleneck structure, similar to MobileNetV2 [San18]. The micro design changes involve replacing Rectified Linear Unit (ReLU) with Gaussian Error Linear Unit (GELU) activation and using Layer Normalization instead of Batch Normalization, aligning with the design of transformer architectures. These changes collectively enhance ConvNeXt's performance, enabling it to compete with transformer-based models while retaining the efficiency of CNNs. ConvNeXt models are available in various scales, catering to different computational budgets and performance needs. Detailed explanations of the design changes are detailed in Appendix A.3.4.

ConvNeXtV2: ConvNeXtV2 [Woo23] builds upon ConvNeXt by introducing two key components: the Fully Convolutional Masked Autoencoder (FCMAE) and Global Response Normalization (GRN). The FCMAE is a Self-SL approach that adapts masked autoencoders (MAEs) for fully convolutional networks (FCNs). It utilizes sparse convolutions to process the visible parts of the input image, thereby preventing the dissipation of mask patterns and promoting generalized feature extraction [Cho19]. The GRN enhances inter-channel feature competition and addresses feature collapse by computing global statistics, normalizing feature responses, and applying a learnable affine transformation. This process maintains diverse feature representations throughout the network, crucial for both Self-SL and supervised learning tasks. The ConvNeXtV2 architecture incorporates GRN into its block structure to enhance feature diversity. ConvNeXtV2 successfully integrates MAE-based Self-SL and novel normalization methods into the ConvNeXt architecture. The FCMAE enables effective learning from unlabeled data, while the GRN enhances feature representation and prevents feature collapse. For detailed explanations of FCMAE and GRN, see Appendix A.3.5.

3.4 Metrics

Methods need to be evaluated to make statements about the quality and meaningfulness of the knowledge obtained from ML training. Hence, multiple metrics used in this work are introduced.

3.4.1 Quantitative Metrics

Pixel Accuracy (PA): In computer vision, the Pixel Accuracy (PA) is a metric used to evaluate the performance of image segmentation models [Ios22]. It measures the proportion of correctly classified pixels across all classes in an image. The PA is defined as:

$$\mathrm{PA} = \frac{\sum_{i=1}^{n} \mathrm{TP}_i}{\sum_{i=1}^{n}(\mathrm{TP}_i + \mathrm{FP}_i + \mathrm{FN}_i)}, \tag{3.4}$$

where n is the number of classes, TP_i is the number of true positive pixels for class i, FP_i is the number of false positive pixels, and FN_i is the number of false negative pixels. The PA ranges from 0 to 1, with 1 indicating perfect pixel-wise classification. While simple to compute, PA can be biased towards dominant classes in imbalanced datasets.

Intersection-Over-Union (IoU): The IoU (Intersection over Union), or Jaccard Index, is a metric used to estimate the similarity of samples [Jac12]. It is defined as the size of the intersection divided by the size of the union of two sets:

$$\mathrm{IoU} = \frac{|\mathcal{A} \cap \mathcal{B}|}{|\mathcal{A} \cup \mathcal{B}|}, \tag{3.5}$$

where $\mathcal{A}$ and $\mathcal{B}$ are two sets. The IoU ranges from 0 to 1, where 1 indicates perfect overlap and 0 indicates no overlap. In image segmentation, $\mathcal{A}$ represents the predicted segmentation mask $\boldsymbol{X}$ and $\mathcal{B}$ the ground truth mask $\boldsymbol{Y}$, which specifies the calculation of the IoU to:

$$\mathrm{IoU} = \frac{\sum_{i=1}^{n} |\boldsymbol{X}_i \cap \boldsymbol{Y}_i|}{\sum_{i=1}^{n} |\boldsymbol{X}_i \cup \boldsymbol{Y}_i|}, \tag{3.6}$$

where n is the number of classes and $\boldsymbol{X}_i$ and $\boldsymbol{Y}_i$ are the ground truth and predicted segmentation masks for class i, respectively.

Dice-Sørensen Coefficient (DSC): To train semantic segmentation mappings, the DSC loss is a popular objective function. The DSC divides the area of overlap between the segments of two segmentation masks by the overall number of pixels in both masks [Sud17]. It is defined as:

$$\mathrm{DSC} = \frac{2\sum_{i=1}^{n} |\boldsymbol{X}_i \cap \boldsymbol{Y}_i|}{\sum_{i=1}^{n} |\boldsymbol{X}_i| + |\boldsymbol{Y}_i|}, \tag{3.7}$$

where n is the number of classes and $\boldsymbol{X}$ and $\boldsymbol{Y}$ are the predicted and ground truth segmentation masks, respectively. The DSC ranges from 0 to 1, with a value of 1 indicating perfect overlap. It is less sensitive to imbalanced data compared to IoU.

Aggregated Jaccard Index (AJI$^+$): In the case of instance segmentation, the Aggregated Jaccard Index or the improved form AJI$^+$ is often used [Gra19, Kum19]. The AJI+ is an extension of the Jaccard Index for multiple instances:

$$\mathrm{AJI}^+ = \frac{\sum_{i=1}^{n} |\boldsymbol{Y}_i \cap \boldsymbol{X}_{j(i)}|}{\sum_{i=1}^{n} |\boldsymbol{Y}_i \cup \boldsymbol{X}_{j(i)}| + \sum_{i=1}^{m} |\boldsymbol{P}_i|} \tag{3.8}$$

where n is the number of classes, $\boldsymbol{Y}_i$ are the ground truth instances, $\boldsymbol{X}_{j(i)}$ are the predicted instances that best match the ground truth, and $\boldsymbol{P}$ is the matrix comprising all m false positive predictions. This metric handles both segmentation quality and detection accuracy.

Pearson Correlation Coefficient (PCC): The PCC measures the linear correlation and essentially describes the normalized covariance of two variables [Coh09]. It is defined as:

$$\mathrm{PCC} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}, \tag{3.9}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are two samples, $\overline{x}$ and $\overline{y}$ are their respective means, and n is the number of elements in each sample. If $\boldsymbol{x}$ and $\boldsymbol{y}$ are images, each x_i and y_i corresponds to the intensity values of pixels at matching spatial locations. In this context, the PCC quantifies the linear correlation between the pixel intensities of the two images, reflecting their similarity in spatial patterns or co-localization of features. The PCC ranges from -1 to 1, where 1 indicates perfect positive correlation, -1 perfect negative correlation, and 0 no linear correlation.

Centered Kernel Alignment (CKA): The CKA is a robust similarity metric for comparing learned features of two CNNs [Kor19]. It is invariant to orthogonal transformations (rotations or reflections of the feature space), ensuring that the similarity remains unchanged regardless of neuron order or orientation. Additionally, it is invariant to isotropic scaling (uniform scaling of all features), meaning that overall scaling factors, such as those resulting from different normalization practices, do not affect the measured similarity. CKA observes the compared NNs at multiple feature layers. For two feature matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$, the similarity is defined as:

$$\mathrm{CKA}(\boldsymbol{K_X}, \boldsymbol{K_Y}) = \frac{\mathrm{HSIC}(\boldsymbol{K_X}, \boldsymbol{K_Y})}{\sqrt{\mathrm{HSIC}(\boldsymbol{K_X}, \boldsymbol{K_X}), \mathrm{HSIC}(\boldsymbol{K_Y}, \boldsymbol{K_Y})}}, \tag{3.10}$$

where $\boldsymbol{K_X}$ and $\boldsymbol{K_Y}$ are kernel matrices derived from $\boldsymbol{X}$ and $\boldsymbol{Y}$ and HSIC is the Hilbert-Schmidt Independence Criterion [Gre05]. CKA measures feature similarity regardless of how filters are ordered or arranged within the layers of the networks being compared.

Error Reduction Rate (ERR): The ERR [McA05] is a metric used to measure the relative improvement in error rates between two systems or methods. It quantifies how much an error rate has been reduced for a method ϵ_m compared to a baseline ϵ_b. The ERR is typically calculated as

$$\mathrm{ERR} = \frac{\epsilon_\mathrm{b} - \epsilon_\mathrm{m}}{\epsilon_\mathrm{b}} * 100, \tag{3.11}$$

where the result is expressed as a percentage, indicating the proportional reduction in errors.

3.4.2 Qualitative Metrics

Class Activation Map (CAM): Observing the CAMs of a CNN enables the visual interpretation of the learned features with attention maps [Zho16]. CAMs highlight the regions in the input image that are most important for making a particular classification decision. For instance, in an image classification task of identifying dogs, a high-quality CAM highlights the dog's body, head, and legs, while ignoring background elements. The intensity of the highlighting indicates the importance of each region for the classification decision.

Nearest Neighbor Comparisons (NNCs): Considering a vector representation of a sample $\boldsymbol{x}$, the neighbors calculated with the Euclidean distance provide valuable insights since samples of the same latent class tend to cluster together and be located far away from dissimilar samples. The k-Nearest Neighbors algorithm can be used to visualize this clustering in a lower-dimensional space. For example, in a feature space learned by a Self-SL model on a dataset of animal images, clusters are expected to form around different animal species, with similar species (e.g., different breeds of dogs) being placed closer together than dissimilar species (e.g., dogs and fish). This clustering behavior can be quantified using metrics like the silhouette score or visualized using techniques like t-SNE [Hin02] or UMAP [McI18].

4 Data

The field of ML offers a wide range of datasets with diverse characteristics. Hence, this chapter defines criteria based on the requirements to answer the research questions of this work and identifies existing datasets that fit. Further, in cases where no suitable options are available for specific criteria, new data is collected to fill these gaps.

4.1 Dataset Requirements

Since the three modules defined in Chapter 2 need specific qualities within the data, the requirements enabling rigorous evaluation of the developed methods are organized according to these distinct needs.

Algorithmic Dataset Enhancement: The first module focuses on optimizing annotation processes in DL (Section 2.2). Methods that aim to transform low-information annotations into more complex ones or merge heterogeneous datasets require datasets with specific characteristics, which ensure the methods can be evaluated adequately without introducing domain compatibility issues. Specifically, the datasets must fulfill the following criteria:

- **Hybrid:** Datasets must include at least two annotation types (for example, classification and segmentation) with varying complexity levels.
- **Adaptability:** The datasets should allow for modification from heterogeneous to homogeneous annotations, enabling evaluation of merging techniques.

- **Controllability:** The progression of annotation complexity (for example, weak to dense labels) must be adjustable.

Deep Learning With Unannotated Data: The second module uses raw data to extract knowledge (Section 2.3). Evaluating methods within this module requires datasets that provide sufficient scale and diversity across domains. The datasets must satisfy these requirements:

- **Composite:** Contain large-scale raw data paired with smaller annotated subsets from similar domains.
- **Task-Difficulty:** Include annotations for challenging tasks like semantic/instance segmentation.
- **Diversity:** Feature samples from diverse domains (biomedical, materials science) and varying data volumes.

Selection and Refinement of Neural Networks: The third module enhances algorithms for real-world challenges (Section 2.4). Evaluating these improvements requires datasets that capture domain complexities and annotation uncertainties. The datasets must meet these criteria:

- **Scenario-Difficulty:** Contain difficult situations (overlapping instances, transient features) common in real applications.
- **Uncertainty-Awareness:** Provide high-quality annotations with uncertainty estimations (inter-annotator disagreements/confidence scores).
- **Domain-Specificity:** Must stem from complex domains like biomedical imaging and materials science.

4.2 Selection of Existing Datasets

Based on the defined criteria, datasets that meet these requirements are presented and subsequently used in this work. All considered datasets are summarized in Table 4.1, and samples for each dataset are shown in Figure 4.1.

Synthetic Cervical Cytology Dataset (Cyto-DS): The Cyto-DS was created for the IEEE ISBI 2014 Image Segmentation Challenge [Lu16]. It includes 945 synthetic grayscale images; each 512×512 pixels, generated from 16 real Extended Depth of Field (EDF) cervical cytology images from four specimens. Individual cells were extracted from the EDF images, randomly transformed (rotated, translated, scaled), and placed on backgrounds to create varying degrees of cell overlap. This synthetic approach enables the controlled generation of images with varying overlap levels, replicating the characteristics of authentic cervical cytology images, which typically contain 2 to 10 cells per image. Ground truth annotations for cell nuclei and cytoplasm boundaries are provided as instance segmentations. The dataset's focus on overlapping instances and its high-quality instance segmentations align directly with the requirements for handling challenging tasks, particularly overlapping objects in real-world scenarios.

Medaka Fish Heart Segmentation Dataset (Heart-DS): The Heart-DS comprises RGB images of the heart region of Medaka (Oryzias latipes) hatchlings viewed from a consistent ventral angle [Sch19]. The transparency of the Medaka's body allows for noninvasive studies of its internal organs and circulatory system [Wak01]. The heart consists of three components: the atrium, which receives deoxygenated blood; the ventricle, which pumps blood into the bulbus; and the bulbus, which directs blood flow to the gill arches. Blood flow through these chambers was recorded in 63 brief videos, each about 11 seconds long at 24 frames per second, from which individual image samples were extracted. The resolution of the RGB images is 640×480 pixels. Only the ventricle is segmented for each image, while the other two chambers remain unannotated. This makes it well-suited for studies on heterogeneously annotated datasets, as it provides a natural example of partial labeling that can be extended or merged with additional annotations.

Droplet Microarray Dataset (Droplet-DS): The Droplet-DS is a collection of binary segmentation masks delineating spheroids within a high-throughput Droplet Microarray platform [Pop19]. This dataset is particularly valuable for experiments in personalized oncology, as it provides crucial data for evaluating tumor spheroids, which are 3D in vitro models that closely mimic the in vivo tumor microenvironment. The dataset includes 588 samples, each representing the precise location of a spheroid through binary segmentation. Despite its relatively small size, its focus on biomedical imaging challenges makes it ideal for research into automated annotation generation from raw data. Its limited size also highlights its potential for testing methods that improve annotation efficiency.

Intl. Skin Imaging Collaboration Melanoma Dataset (ISIC-DS): The ISIC-DS originates from the 2017 ISIC challenge and includes 2.600 close-up RGB images of skin lesions [Cod18]. The segmentation task involves generating binary masks to locate lesions within each respective image. Additionally, it contains a categorization task with three classes: *Seborrheic Keratosis*, *Melanoma*, and *Unknown*. Thus, this dataset provides semantic segmentation masks and classification labels for detectable lesions. Its dual annotation types (segmentation and classification) fulfill the requirements for generating improved annotations. Furthermore, as a large-scale biomedical dataset with challenging lesion boundaries, it supports evaluations on extracting knowledge from unannotated data.

Multi-Organ Nucleus Segmentation Dataset (MoNu-DS): As part of the 2018 MICCAI challenge, the MoNu-DS dataset contains histopathological samples from various organ types [Kum19]. A total of 44 individual images are included. The challenge involves segmenting and identifying each nucleus in the multi-organ images (instance segmentation). Additionally, classification annotations are provided to differentiate the respective organs. The organ classes include *Kidney*, *Colon*, *Breast*, *Bladder*, *Prostate*, *Liver*, *Stomach*, *Brain*, and *Lung*. The classes *Liver* and *Stomach* are present only in the training set, while *Brain* and *Lung* are found only in the test set. Similar to the ISIC-DS, its combination of segmentation and classification annotations fulfills requirements for generating improved annotations and extracting knowledge from unannotated data. Moreover, its smaller sample size and increased complexity (more classes and instance segmentations) make it an even greater challenge than the ISIC-DS.

Name	Segmentation Task	#Detection Classes	Classification	#Samples	Special Characteristics
Cyto-DS [Lu16]	Instance	One	No	945	Overlapping Instances
Heart-DS [Sch19]	Semantic	One	No	730	Unannotated Heart Chambers
Droplet-DS [Pop19]	Semantic	One	No	588	Difficult due to limited sample quantity
ISIC-DS [Cod18]	Semantic	One	Yes	2.600	Both segmentation and classification available
MoNu-DS [Kum19]	Instance	One	Yes	606	Both segmentation and classification available

Table 4.1: Overview of the existing datasets used in this work, listed with their name and relevant characteristics. *Segmentation Task* describes the type of annotations available in the respective dataset, *#Detection Classes* is the number of individual classes contained, *Classification* is whether there are classification labels for the individual images available, *#Samples* lists the number of samples, and *Special Characteristics* highlights peculiarities of the datasets.

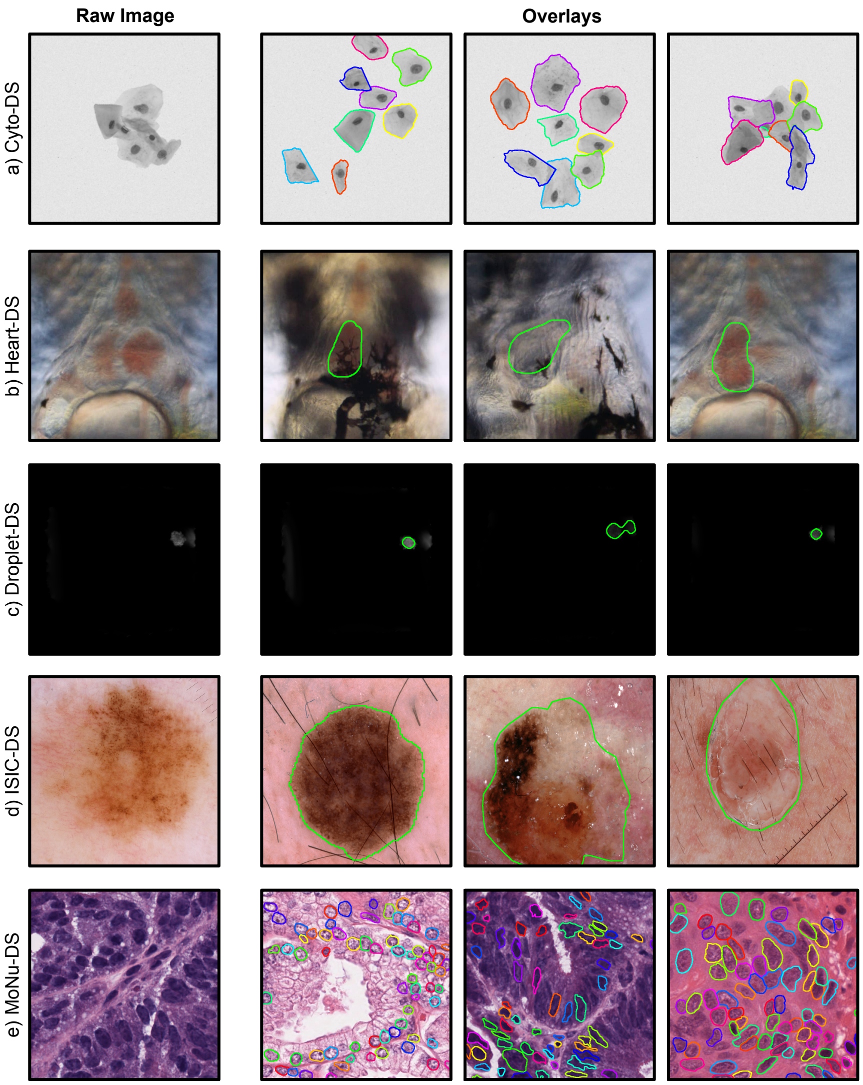


Figure 4.1: Examples of the existing datasets used. A raw image without annotation is shown on the left for each dataset. To illustrate the variety and difficulty of the data, three images with annotations are also shown. Semantic segmentation masks are visualized with green borders and color-coded instance segmentations. Since the instances in the MoNuSeg dataset are particularly small, the sample shown is cropped.

4.3 Novel Datasets

The selected existing datasets lack critical characteristics required for this work: they do not provide heterogeneous or partially labeled annotations, uncertainty quantification, or multi-annotator disagreement information, nor do they include realistic complex scenarios such as overlapping instances, or large-scale unlabeled raw data representative of automated acquisition workflows. To address these gaps, this section introduces novel datasets generated and annotated according to the defined requirements, enabling a comprehensive investigation and improvement of the DL pipeline. All novel datasets are publicly available, with detailed access information provided in Appendix A.4.1.

4.3.1 Extended Heart Segmentations

The original Heart-DS is limited by the absence of comprehensive annotations for all heart chambers. Specifically, the atrium and bulbus are not included in the original labels. This omission restricts the dataset's utility for research that requires a complete anatomical understanding of the heart, particularly in scenarios where chamber visibility fluctuates due to physiological changes during the cardiac cycle. The extended Heart-DS addresses this limitation by adding annotations for the atrium and bulbus across all samples, creating a more complete and realistic representation of cardiac anatomy. Figure 4.2 shows two samples of the dataset. This extension of the Heart-DS is central to the focus of this work, as it provides a controlled setting to study heterogeneous labeling. Heart chambers are ideal for such analysis because visibility fluctuates naturally during the cardiac cycle, creating partial/missing annotations that reflect real-world annotation challenges. Hence, this extended dataset enables the structured analysis and development of algorithms that address data heterogeneity.

A single label mask for a specific class is generated in approximately 25 seconds. The total time required to enhance the Heart-DS with the atrium and bulbus classes is approximately 10 hours. Despite the near transparency of Medaka, the heart chambers can only be distinctly identified in recordings if filled with blood.

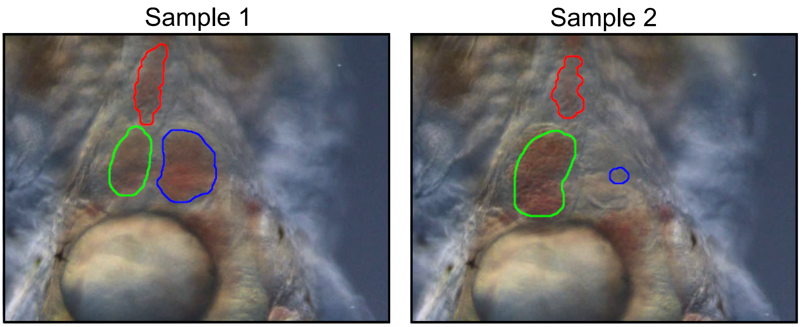


Figure 4.2: Two samples of the extended Heart-DS. The bulbus is marked in red, the ventricle in green, and the atrium in blue. In Sample 1, all three heart chambers are visible; in Sample 2, the atrium is hardly recognizable. Adapted from [Sch22d].

The cardiac cycle poses difficulties in consistently labeling a chamber when it is partially empty, and this becomes particularly challenging when the chamber contains almost no blood (end-systolic). This makes annotating burdensome, but investigations into heterogeneously annotated data sets are even more interesting. If the heart chambers are not visible, this inherently leads to missing annotations of the individual heart chambers. This difficulty in recognizing the heart chambers is visible in Sample 2 in Figure 4.2, where the atrium is almost invisible.

4.3.2 Particle Segmentations With Uncertainties

For the selection and refinement of NNs (Section 2.4), there is a need for datasets that quantify uncertainty in complex domains, include challenging tasks such as instance segmentations, and encompass multiple levels of sample difficulty to enable the expressive evaluation of methods. Additionally, rapid sample generation is required to allow the assessment of DL with unannotated data (Section 2.3). However, datasets that combine complex tasks with explicit uncertainty information and a range of sample difficulties are not yet available. To address this gap, the novel Powder Uncertainty Scanning Electron Microscopy Dataset (SEM-DS) dataset is created and annotated with uncertainty quantifications. This new resource enables the development and evaluation of algorithms that can handle real-world annotation uncertainty, directly benefiting both machine learning researchers working on robust segmentation models and materials scientists who require reliable automated analysis in complex imaging scenarios.

A Phenom Desktop SEM from Thermo Fisher Scientific is used to obtain a collection of powder samples stemming from battery science. Each image contains one of the following compounds: $NaAlSiO_4$, $Cu_3(PO_4)_2$, MgO, Mn_3O_4, Na_2CO_3, TiO_2, $BaCO_3$, SiO_2, $CaTiO_3$, $BaCuO_2$, and $LiCoO_2$. It contains scans of different magnifications, which are separated into two categories: the first comprises images obtained at magnifications not exceeding $10,000\times$, referred to as low magnification captures. Conversely, the second category consists of images acquired at magnifications exceeding the $10,000\times$ threshold, known as high magnification captures. Figure 4.3 presents two samples of both magnifications with the ground-truth annotations overlaid. The image acquisition process captures

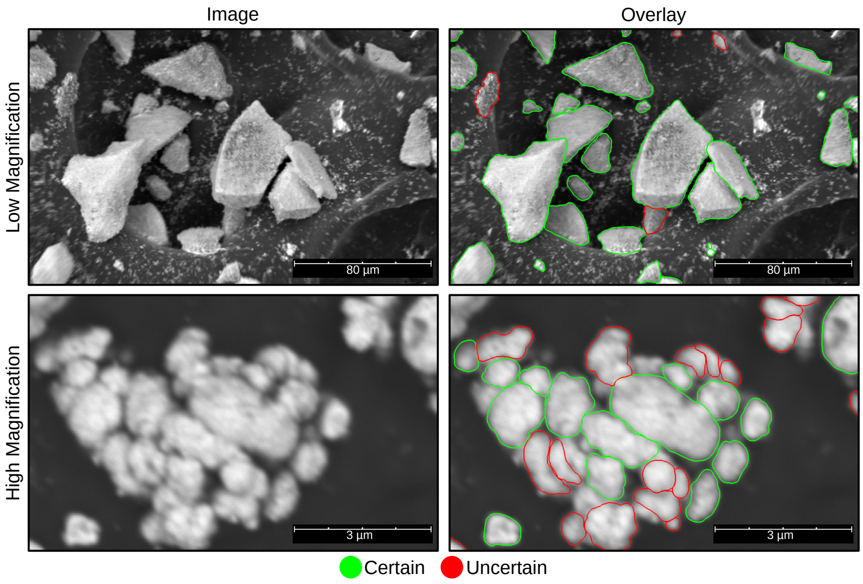


Figure 4.3: Two examples of the SEM-DS acquired at low (top) and high magnification (bottom). In the right panels, colored curves represent particle boundaries outlined by domain experts, with green indicating certain labels and red denoting uncertain labels. Adapted from Adapted from [Ret24c].

various particle configurations, ranging from well-dispersed individual particles to densely packed agglomerates. This approach ensures that the dataset encompasses diverse particle arrangements, sizes, and morphologies, comprehensively representing the different structures observable in SEM scans of powder samples.

All images are hand-labeled by domain experts tasked with segmenting distinct particles within each sample (instance segmentations). Given the challenging nature of labeling desktop SEM images, especially when overlapping particles are present, and to evaluate uncertainty estimation approaches, each segmented particle is divided into *certain* and *uncertain* particles. Particles with well-defined edges, clear contrast, and minimal overlap, allowing domain experts to delineate their boundaries confidently, are given *certain* labels by annotators. In contrast, *uncertain* labels are assigned when boundaries are less clear, often due to image blur, agglomeration, or complex morphology. The labeling team consists of three domain experts (two graduate students and a staff scientist) from Lawrence Berkeley National Laboratory, all with backgrounds in materials science and chemical engineering. The SEM images are evenly distributed among these researchers for labeling. They are instructed to manually designate the areas of any primary particles that are $\geq 1\,\mu\text{m}$ in diameter. While all three domain experts receive identical instructions, some variation in label assignment may occur due to personal experience, time spent on the task, and individual interpretation of particle boundaries in challenging cases. These variations in the training set are intentionally preserved to ensure robustness and prevent biases in the data that follow the preferences of a single researcher. This approach aligns with the inherent subjectivity in particle segmentation, especially when dealing with complex morphologies and varied image quality.

Adding new annotated samples, the low and high magnification datasets are progressively enlarged. The process continues until no further performance improvements are observed in the Mask R-CNN model. This results in 90 images: 50 images at low magnification, each with a resolution of 1920×1200 pixels, and 40 images at high magnification, each with a resolution of 7680×4800 pixels. The high magnification split contains 153 particles with certain boundaries and 221 uncertain ones, while the low magnification split contains 260 certain particles and 2514 uncertain particles. All images are resized to a uniform resolution of 1920×1200 pixels[1].

4.3.3 Multi-Annotator Particle Benchmark

To further assess the generalization abilities of the developed DL approaches, a second SEM dataset is created, which is not intended for training purposes but rather to serve as a benchmark. This dataset, called $LiCoO_2$ Grid SEM Dataset (LGS-DS), is generated from a sample of $LiCoO_2$, a compound not found in the SEM-DS. The powder, sourced from Sigma Aldrich, is carefully dispensed onto a sample stub in preparation for SEM. Utilizing the Phenom XL desktop SEM, 288 images are captured. Unlike most existing particle segmentation datasets, which usually rely on a single expert annotation per image, three distinct domain experts independently annotate the LGS-DS. This approach enables the systematic capture of inter-annotator variability, which is often overlooked due to the significant manual effort required. By explicitly quantifying expert disagreement, the LGS-DS offers a novel benchmark for evaluating the robustness and reliability of DL models under realistic annotation variability. This benefits both ML researchers, who can use the dataset to develop and test uncertainty-aware algorithms, and materials scientists, who require reliable automated analysis that reflects the complexities of expert interpretation in practical scenarios.

Each image encompasses an 80×80 μm area of the sample. Three distinct domain experts are tasked with annotating the entire set of images. This approach differs from the SEM-DS method, where the annotation workload is distributed among the annotators. In this case, each expert independently labels all samples, which enables the capturing of inter-annotator variations and a more in-depth comparison to DL methods, compared to having only a single annotation per sample. Figure 4.4 illustrates two representative samples from the dataset, along with the three annotations provided. In this dataset, the experts often disagree on the precise boundaries and number of individual objects, particularly in cases

[1] Details regarding the downscaling process are elaborated in Appendix A.4.2.

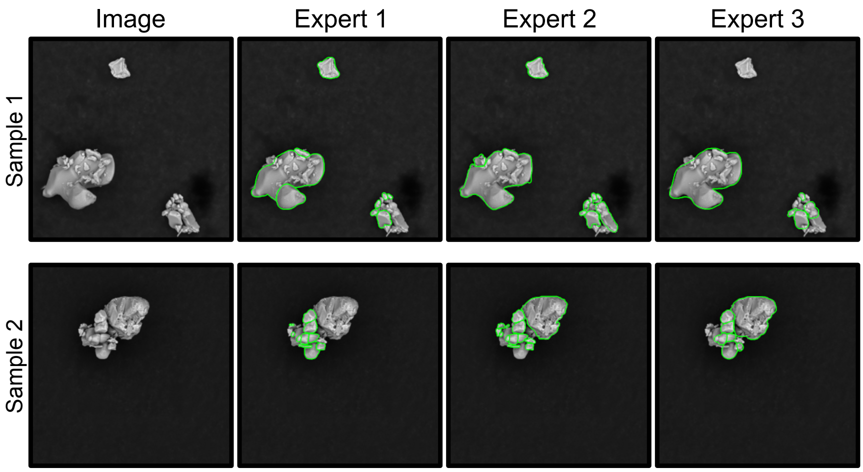


Figure 4.4: Two samples from the LGS-DS with the ground-truth segmentation masks of each expert overlayed in green.

where particles are closely clustered or overlapping. This disagreement is most noticeable in complex regions where edges are less distinct, leading to variations in how each expert interprets and delineates individual instances. Additionally, for each segmentation mask, the annotator records the time taken to complete the annotation task. Furthermore, no distinction is made between certain and uncertain particles for this dataset.

4.3.4 Expanded Unlabeled Particle Scans

Using the same setup as with the SEM-DS, a Phenom Desktop SEM from Thermo Fisher Scientific is used to obtain a wide variety of raw samples in an automated manner. The resulting dataset is referred to as the A-SEM-DS. Unlike most existing SEM datasets, which are typically small and curated with manual annotations, this unlabeled SEM dataset realistically reflects the diversity and imperfections encountered in automated imaging workflows. The A-SEM-DS provides a substantial set of 24.751 raw SEM images, generated automatically in three separate runs. The dataset is intended for DL with unannotated data and used in combination with the SEM-DS to evaluate methods that leverage raw samples.

The images are not filtered or screened during or after capturing and often contain blurred particles or only background noise. Figure 4.5 shows six dataset examples, illustrating the range from clearly recognizable particles to samples containing only background noise or blurred structures. Even though samples with little

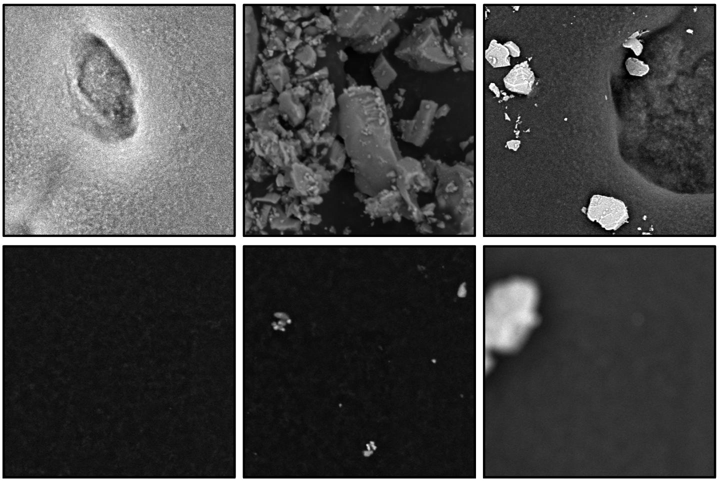


Figure 4.5: Six examples of the Automated Scanning Electron Microscopy Scans Dataset (A-SEM-DS) dataset, reflecting the types of samples contained. The dataset contains easily recognizable particles (top row) and samples with only background noise or blurred structures (bottom row). Adapted from [Ret25b].

to no learnable content can hinder NN training, the dataset is intentionally left unprocessed to reflect a realistic scenario of raw data as encountered in high-throughput and automated laboratory settings. This makes the A-SEM-DS a unique and valuable resource for developing and benchmarking methods that must handle large volumes of uncurated, unlabeled data, benefiting researchers working on Self-SL, representation learning, and robust automated analysis in materials science.

5 Algorithmic Dataset Enhancement

Parts of this chapter are based on:

- M. Schutera, L. Rettenberger, C. Pylatiuk, and M. Reischl. Methods for the frugal labeler: Multi-class semantic segmentation on heterogeneous labels. *PLoS ONE*, volume 17, page e0263656, Public Library of Science, 2022. DOI: 10.1371/journal.pone.0263656.
- L. Rettenberger, M. Schilling, and M. Reischl. Annotation efforts in image segmentation can be reduced by neural network bootstrapping. In *Current Directions in Biomedical Engineering*, volume 8, pages 329–332, Innsbruck, Austria, 2022. De Gruyter. DOI: 10.1515/cdbme-2022-1084.

5.1 Overview

The success of DL approaches relies heavily on large, well-annotated datasets for training. In practice, generating such datasets poses considerable challenges since expert domain knowledge and extensive time investments are often required for accurate annotations. Here, the process of creating pixel-wise segmentation masks is especially labor-intensive. Two key challenges arise in this context. First, databases often contain partially annotated samples or multiple similar datasets that cannot be easily combined due to differences in annotation types or completeness. This heterogeneity in labeling presents a significant obstacle for traditional SL approaches, which typically require fully and consistently labeled data. Training on such heterogeneous data drastically reduces usable samples, limiting the task scope to a reduced set of classes or rendering the training process unfeasible. Second, generating annotations is burdensome and time-consuming, resulting in valuable data being unused or underutilized due to limited time for annotation. Recognizing these challenges, this chapter introduces methods to enhance datasets algorithmically. First, a technique called the Neural Bootstrapping Framework (NeuBoot) is proposed, which utilizes unsupervised machine learning to automatically generate segmentation masks from raw data, drastically reducing the need for manual annotation. Secondly, approaches for using heterogeneous labels are presented, including training on partially annotated data and jointly combining datasets with different annotation types. This includes a novel approach for dataset evaluation called Controlled Heterogeneity Generator (CoHeG), which enables the artificial creation of heterogeneous datasets from homogeneous sources. In addition, this section introduces a new metric, Performance Frugality Ratio (ψ_{PFR}), and a combined objective function, Class Asymmetric Loss (CAL), both specifically tailored for datasets with incomplete annotations. This paves the way for more efficient and flexible use of datasets, which, in consequence, increases the robustness, efficiency, and accuracy of DL models.

5.2 Related Works

Segmentation Mask Generation: Multiple approaches have been developed within computer vision to address the challenge of generating annotations from raw data. Some approaches aim to minimize the effort required for pixel-wise segmentations by employing less granular annotations than full masks. One such method, the ilastik toolkit [Ber19], implements interactive machine learning for bioimage analysis, enabling segmentation via sparse pixel annotations paired with random forest classifiers. An alternative technique, ScribbleSup [Lin16], utilizes sparse scribbles to supervise CNNs in generating segmentation masks.

Further, BoxSup [Dai15] draws inspiration from localization algorithms by providing bounding boxes to the CNN instead of masks. Further, a practical approach using thresholding techniques and human verification efficiently generates masks but lacks quantitative evaluation [Wol20]. While these frameworks reduce the annotation effort, human interaction is still necessary for each sample, and the resulting annotations are often limited to specific frameworks, lacking general applicability. Other approaches focus on enhancing learning rules rather than relying on explicit annotator support. A commonly employed technique involves using classification labels as weak annotations for object segmentation in images [Kol16, Zha19]. Another promising strategy involves mining and erasing the most discriminative regions in an image for classification tasks, followed by assessing the impact on classification performance to gradually develop segmentation masks for the classified object [Wei17]. However, weakly-supervised models often yield unsatisfactory results, potentially due to the neglect of spatial structures. This issue has been addressed within contrastive learning and is equally applicable in this context [Wan21a]. Furthermore, these methods typically require extensive datasets and complex additional training procedures. Additionally, all mentioned methods still necessitate some form of human interaction, fail to generate complete segmentation masks, demand large datasets, and are frequently challenging to implement. These limitations underscore the need for further research and development in this area.

Heterogeneous Labeling: In supervised DL, training usually relies on large, labeled datasets [Hal09, Lin14, Rus15, Mün24]. However, real-world data is often heterogeneous, for example, originating from multiple domains or containing only partial annotations. Transforming heterogeneous data into a homogeneous format by filtering out conflicts often reduces performance, decreases the number of usable classes, or even makes training impractical [Hal09]. This problem is particularly pronounced with heterogeneous annotations, since standardizing conflicting labels into a uniform format often means that valuable domain knowledge is discarded [Lin14, Rus15]. Multiple approaches have been developed to enable the usage of heterogeneously annotated datasets for supervised NN training. One field of research explores multi-modal learning, where input samples originate from multiple source modalities such as audio, image, text, and video. Another domain concentrates on multi-task learning, requiring the NN to learn multiple tasks. Most commonly, NNs in this domain are based on learning a joint representation or a shared feature space, coupled with either separate encoding paths in the multi-modal setting [Bal18], or discrete classification output layers in multi-task learning [Mel18, Leo19, Brb20]. As a result, a substantial amount of ground truth information is lost if data is only partially labeled, which is common in heterogeneous data sources. The most straightforward approach to handling missing labels involves excluding unlabeled entities from the loss computation within the objective function. However, this addresses only the technical challenge of an undefined loss value when ground truth data is unavailable [Gon18]. A marginally more sophisticated approach is to treat missing class labels as background, which, however, introduces the assumption that all unlabeled elements are background, which is often not the case [Dmi19, Vu21, Vu21]. A more refined approach is to train a shared feature space across diverse datasets or classes, while introducing fine-tuned sub-classifier heads, which enables the merging of multiple datasets. However, this approach still treats unknown labels as background without leveraging additional knowledge [Fan20]. More data-centric approaches attempt to incorporate knowledge by utilizing class similarities for merging, necessitating the precondition that multiple classes exhibit similar appearances. Furthermore, merging multiple classes results in training a more general (and less complex) task [Shi21]. Finally, a promising yet straightforward approach capitalizes on the fact that virtually all semantic segmentation tasks are multi-class problems, with all data points assigned a single, unambiguous class. This realization enables an objective function that exploits the mutually exclusive nature of ground truth masks in semantic segmentation [Pet18, Kon19]. The methods mentioned above address unlabeled information in datasets by either modifying established training techniques or exploiting supervision cues present in the data. However, to date, no approach has combined method modification and supervision cue exploitation to fully leverage the explicit and implicit information contained within the annotations.

5.3 Methods

This chapter presents methods to algorithmically enhance datasets and reduce manual effort in training SL models. The focus is on two main components: generating annotations from raw samples and integrating heterogeneous data sources. The goal is to minimize manual annotation efforts while maximizing the utility of existing labels. The first component introduces the NeuBoot framework, which automatically generates high-quality segmentation masks from raw data, reducing the need for manual annotation through iterative refinement of predictions. The second component addresses the challenge of training neural networks for multi-class semantic segmentation using datasets with heterogeneous labels. It proposes the novel objective function CAL, introduces the new similarity metric ψ_{PFR}, and establishes the heterogeneous label generation framework CoHeG. Together, these approaches aim to streamline the annotation process and improve the effectiveness of SL models by leveraging both automated annotation techniques and advanced methods for combining diverse datasets.

5.3.1 Segmentation Mask Generation

The novel NeuBoot approach enables the generation of segmentation masks from raw data without human intervention by using a bootstrapping approach, in which a simple autonomous process is used to initiate and support the development of a more complex one. Through this interaction, spatial features are learned without human input, and the complex process can generalize beyond the erroneous and noisy annotations provided by the simple one[1]. This approach yields a set of approximated annotations, which can be manually refined or directly applied in ML applications. Compared to previous approaches, this framework requires no

[1] The generalization capability is a reasonable assumption, as it has been demonstrated that moderate annotation noise does not negatively impact the learning process [Sch22b].

human intervention, generates complete segmentation masks, applies to small datasets, and has no complicated additional steps for mask generation[2].

A conceptual overview of the NeuBoot framework is shown in Figure 5.1. It involves the utilization of an unannotated, raw dataset $\mathcal{D}^u$ and a naïve process $\mathcal{T}$ that generates imperfect segmentation masks $\mathcal{L}^t = \mathcal{T}(\mathcal{D}^u)$ due to its simple nature. These masks $\mathcal{L}^t$ are subsequently employed in a NN $\mathcal{N}$ (or any process more sophisticated than $\mathcal{T}$) to derive segmentation masks $\mathcal{L}^n = \mathcal{N}(\mathcal{D}^u, \mathcal{L}^t)$. It is anticipated that $\mathcal{N}$, capable of learning correlations and generalization, will filter out noise from erroneous annotations, thereby enhancing the quality of the generated segmentation masks. The resulting learned masks, $\mathcal{L}^n$, can be integrated into post-processing steps, manually refined, or directly applied in various applications, depending on the specific requirements of the task and the quality of the generated dataset.

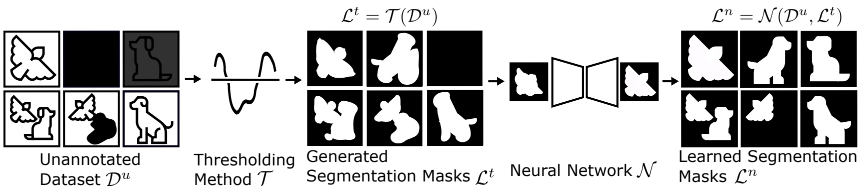

$\mathcal{L}^t = \mathcal{T}(\mathcal{D}^u)$
$\mathcal{L}^n = \mathcal{N}(\mathcal{D}^u, \mathcal{L}^t)$
Unannotated Dataset $\mathcal{D}^u$
Thresholding Method $\mathcal{T}$
Generated Segmentation Masks $\mathcal{L}^t$
Neural Network $\mathcal{N}$
Learned Segmentation Masks $\mathcal{L}^n$

Figure 5.1: Overview of the Neural Bootstrapping Framework (NeuBoot). An unannotated dataset $\mathcal{D}^u$ is processed using a thresholding technique $\mathcal{T}$, resulting in a set of generated segmentation masks $\mathcal{L}^t$ that likely contain noise and imperfections. These masks $\mathcal{L}^t$ are subsequently utilized as ground truth annotations in a supervised NN training. The NN is expected to generalize beyond the errors and noise of the provided segmentation masks $\mathcal{L}^t$ during training. The outcome of this process is the generation of enhanced learned masks $\mathcal{L}^n$, which can be further manually enhanced or directly applied in applications. Adapted from [Ret22].

Here, the naïve method $\mathcal{T}$ is implemented as either the Otsu algorithm [Ots75] applied to image intensity values or k-means clustering [Mac67, Llo82], where the largest cluster is assumed to represent the background. The NN $\mathcal{N}$ is the

[2] In this work, simple processes are defined as methods that produce segmentation masks by thresholding pixel intensity values, while the complex process is realized by a CNN. However, both processes may be any algorithm that generates segmentation masks.

U-Net [Ron15] [3]. The evaluation of the results is conducted using the DSC, IoU, and PA metrics. Details regarding the metrics are explained in Section 3.4.1.

5.3.2 Heterogenous Labeling

While the NeuBoot framework enables the generation of segmentation masks from raw data, the focus here shifts to combining existing datasets for joint training, thereby further enabling training with available data. Firstly, the CoHeG framework converts fully annotated datasets into partially labeled ones, enabling controlled studies of label incompleteness. Secondly, the novel ψ_{PFR} metric measures the similarity between ground truth and predictions, accounting for data heterogeneity, and the CAL objective function enabled robust training on incompletely annotated datasets. Together, these contributions allow systematic analysis and training with incomplete annotations.

Heterogeneous Label Dataset Generation: The CoHeG framework generates heterogeneous datasets from a fully annotated source by systematically varying missing labels. This creates a controlled environment for ablation experiments under diverse label availability conditions. Compared to relying on multiple heterogeneous datasets, CoHeG isolates the effects of missing labels, enabling precise and systematic analysis.

A generic dataset $\mathcal{X}$ with annotations $\tilde{\mathcal{Y}}$ is employed for heterogeneous annotation generation. Each sample label pair $(\boldsymbol{x}, \tilde{\boldsymbol{y}})$ is composed of an RGB image $\boldsymbol{x}$ with dimensions $(w, h, 3)$, where w and h denote width and height, respectively. The corresponding ground truth label $\tilde{\boldsymbol{y}}$ is given with dimensions (w, h, c), where c represents the number of classes. The total number of samples in the dataset is defined as s. To obtain a heterogeneous dataset from the homogeneous source $\mathcal{X}$ and $\tilde{\mathcal{Y}}$, a subset of classes $\mathcal{S} \subseteq \{1, 2, \ldots, c\}$ is defined, from which labels may be removed. For each class in $\mathcal{S}$, a specified absolute number of labels $p \in \mathbb{N}$

[3] Even though only a few potential methods are considered in this study, it is important to note that both $\mathcal{T}$ and $\mathcal{N}$ may be arbitrary processes generating segmentation masks of varying quality.

with $p \leq s$ is removed from the ground truth labels $\tilde{\mathcal{Y}}$. The relative share of removed labels is computed as $\rho = p/s$. This procedure enables the generation of numerous unique, synthetically produced heterogeneous datasets originating from the same data source, which is essential for structured label ablation experiments on heterogeneous datasets.

To support robust experimentation, labels are dynamically removed during training to generate unique label configurations in real time, avoiding redundancies. Data is always sourced from the original homogeneous dataset, with the label removal rate defined as a relative value (e.g., 20% per class) to accommodate unknown or variable sample sizes. This enables the systematic analysis of label incompleteness through the generation of real-time synthetic datasets. A sample memory $\mathbf{M}$ tracks modified labels, ensuring repeated accesses to samples never expose the dataset's complete ground truth. For instance, if label $\tilde{\boldsymbol{y}}$ is altered during one request, $\mathbf{M}$ ensures the same modification applies to subsequent requests. Without the memory $\mathbf{M}$, repeated accesses could eventually reveal all heterogeneous variations, exposing the original homogeneous dataset. After each training iteration[4] $\mathbf{M}$ is reset, preserving online capabilities (e.g., dynamic adjustments during training) while maintaining complete user control over dataset generation.

Performance Frugality Ratio: To evaluate prediction accuracy while explicitly accounting for data efficiency and heterogeneity, a novel metric, the Performance Frugality Ratio (ψ_{PFR}), is introduced. It extends existing statistics, such as the DSC or IoU (denoted as $\mathbf{S}$), by penalizing low scores and rewarding models that achieve comparable performance with a smaller number of annotated samples. This approach enables a nuanced assessment of how efficiently the employed model utilizes available data.

To formalize this, a penalty parameter n is introduced, which determines the severity of lower scores. The ψ_{PFR} metric is calculated by dividing $\mathbf{S}$ to the power of n by the number of available labels, which is determined by subtracting the share of dropped labels p from the total number of sample-label pairs s:

[4] A training iteration refers to one cycle of updating the model with a batch of data.

$$\psi_{\text{PFR}}(\mathbf{S}) = \frac{\mathbf{S}^n}{s-p}, \tag{5.1}$$

with $n > 1$. The ψ_{PFR} metric provides a more comprehensive evaluation than $\mathbf{S}$ alone by incorporating dataset heterogeneity and using exponential weighting to penalize lower performance scores strictly. Specifically, if a model achieves similar performance using fewer labeled samples, it proves more efficient by delivering comparable results with less available information, thereby showing superior performance.

Combined Objective Function: The novel combined loss function is a modified adaptation of a sample dropping objective function [Vu21], enhanced by incorporating the Class Asymmetric Loss (CAL) [Kon19]. This function is specifically designed to facilitate training on heterogeneous, labeled datasets, enabling learning from varying annotation rates and stabilizing the learning process by leveraging implicit information from unlabeled classes. The channels (corresponding to class labels) and the normalized predicted output for all classes at each pixel are used to calculate the output of the objective function. This approach enables a comprehensive and robust training process that takes into account the complexities of heterogeneous data while maintaining stability and coherence in the learning algorithm.

In the following, $\tilde{\boldsymbol{Y}}_{w \times h \times c}$ is the ground truth tensor, while $\hat{\boldsymbol{Y}}_{w \times h \times c}$ represents the NN prediction. As NNs usually require the input to be of constant size, missing annotations must be marked differently than simply omitting them. To address this, a binary label mask vector $\boldsymbol{m} \in \mathbb{R}^k$ is introduced, where $k = \{1, \ldots, c\}$. This vector indicates the presence or absence of a ground truth mask for each of the c classes. Specifically, m_i is assigned a value of 1 when a ground truth mask exists for a given class and 0 when it does not.

The objective function proposed is composed of two distinct components. The first part aims to optimize each channel of the prediction tensor $\hat{\boldsymbol{Y}}$ to closely approximate the ground truth $\tilde{\boldsymbol{Y}}$ with each iteration, referred to as channel optimization. For this purpose, the Dice-Sørensen Coefficient (DSC)[5] is utilized. Specifically, the DSC is calculated by determining the ratio of the overlapping area between the segments of two segmentation masks at a given class index i to the total number of pixels in both masks. To prevent division by zero, a small positive value ϵ is added to both the numerator and denominator:

$$DSC(i, \tilde{\boldsymbol{Y}}, \hat{\boldsymbol{Y}}) = \frac{2 \sum_{u=1}^{w} \sum_{v=1}^{h} (\tilde{Y}_{u,v,i} \cdot \hat{Y}_{u,v,i}) + \epsilon}{\sum_{u=1}^{w} \sum_{v=1}^{h} \tilde{Y}_{u,v,i} + \sum_{u=1}^{w} \sum_{v=1}^{h} \hat{Y}_{u,v,i} + \epsilon}. \tag{5.2}$$

This formulation enables a robust evaluation of the similarity between the predicted and ground-truth segmentation masks.

The DSC statistic is embedded into a modified Dice loss [Sud17]. In the proposed modification, for each mask, the value of $(1 - DSC)$ is multiplied by m_i, ensuring that the current value is only considered if the specific mask is present ($m_i = 1$). Subsequently, the accumulated loss is normalized with respect to the number of present masks, resulting in the modified $\mathcal{L}_{DSC}$ that only considers values that are actually present and are consistently scaled according to the number of available masks:

$$\mathcal{L}_{DSC}(\tilde{\boldsymbol{Y}}, \hat{\boldsymbol{Y}}) = \frac{\sum_{i=1}^{c} m_i \, (1 - DSC(i, \tilde{\boldsymbol{Y}}, \hat{\boldsymbol{Y}}))}{\sum_{i=1}^{c} m_i}. \tag{5.3}$$

The second component of the combined objective function leverages the fact that pixels are not simultaneously assigned to multiple classes in a multi-class segmentation problem, referred to as probability distribution optimization. For this, the CAL [Kon19] is adapted for multi-class segmentation challenges. The modified CAL only observes masks that do not have a ground truth value. The

[5] a high-level introduction to the DSC metric is provided in Section 3.4.1.

function will produce high values for cases where the NN predicts that a specific pixel belongs to a class known to be wrong since the pixel has a ground-truth value for a different, annotated class. This is because a pixel cannot simultaneously belong to two distinct classes; hence, the prediction of the NN must be incorrect. However, if a pixel lacks a defined ground truth for all classes, it is impossible to assert whether it truly belongs to the unlabeled mask.

To calculate CAL, the given mask index i values are summed over all coordinates. For each coordinate (u, v), the pixel at the same position in all mask indices z, excluding the current mask i, is observed, and the predicted value $\hat{Y}_{u,v,z}$ at that coordinate is examined. An activation function $f(.)$ is applied to constrain the output value range. The result is then multiplied by $(1 - m_z)$ to consider only masks without a set ground truth. Finally, the summation is multiplied by the ground truth value $\tilde{Y}_{u,v,i}$ to ensure that the summation is only considered if $\tilde{Y}_{u,v,i}$ represents the correct class for the pixel at position (u, v):

$$CAL(i, \tilde{\boldsymbol{Y}}, \hat{\boldsymbol{Y}}) = \sum_{u=1}^{w} \sum_{v=1}^{h} \left(\tilde{Y}_{u,v,i} \cdot \sum_{z \in \{1,\ldots,c\} \setminus \{i\}} f(\hat{Y}_{u,v,z}) \cdot (1 - m_z) \right). \tag{5.4}$$

CAL is then utilized in $\mathcal{L}_{CAL}$ to compute the second component of the composed objective function. Similar to $\mathcal{L}_{DSC}$, $\mathcal{L}_{CAL}$ is determined by summing over all classes, considering only those for which a mask exists ($m_i = 1$):

$$\mathcal{L}_{CAL}(\tilde{\boldsymbol{Y}}, \hat{\boldsymbol{Y}}) = \sum_{i=1}^{c} m_i \, CAL(i, \tilde{\boldsymbol{Y}}, \hat{\boldsymbol{Y}}). \tag{5.5}$$

If the background class is defined as the class where none of the c defined classes is present, it must be excluded from the $\mathcal{L}_{CAL}$ calculation since such a naive definition of the background class makes it impossible to differentiate between unlabeled pixels and actual background.

The combined objective function is formulated as follows:

$$\mathcal{L}_\lambda = (1-\alpha)\,\mathcal{L}_{DSC} + \alpha\mathcal{L}_{CAL}, \tag{5.6}$$

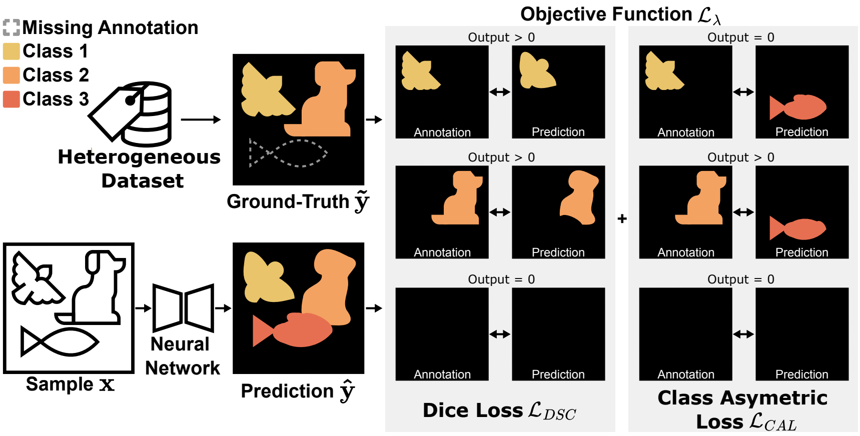

Missing Annotation
Class 1
Class 2
Class 3
Heterogeneous Dataset
Ground-Truth $\tilde{\mathbf{y}}$
Sample $\mathbf{x}$
Neural Network
Prediction $\hat{\mathbf{y}}$
Objective Function $\mathcal{L}_\lambda$
Output > 0
Output = 0
Annotation
Prediction
+
Dice Loss $\mathcal{L}_{DSC}$
Class Asymetric Loss $\mathcal{L}_{CAL}$

Figure 5.2: The combined objective function $\mathcal{L}_\lambda$ is visualized through a process where a sample $\mathbf{x}$ is input into the neural network. A training prediction $\hat{\boldsymbol{y}}$ is then produced, each class represented by a distinct color. The corresponding ground truth label $\tilde{\boldsymbol{y}}$ is given and misses Class 3, which is indicated by a dotted line where the label would be located. The calculation of $\mathcal{L}_\lambda$ is depicted on the right, showing the individual objective functions $\mathcal{L}_{CAL}$ and $\mathcal{L}_{DSC}$. For each class, the respective annotations and corresponding predictions are displayed. Due to prediction errors in both given classes, the Dice loss outputs a value $\mathcal{L}_{DSC} > 0$ for both. However, the unlabeled one outputs 0 as it cannot be determined without the ground truth mask. The class asymmetric loss only produces a value $\mathcal{L}_{CAL} > 0$ when an unlabeled class's predicted segmentation mask intersects with a given class's ground truth mask. In this case, a small portion of the unlabeled Class 3 was predicted as part of Class 2, resulting in $\mathcal{L}_{CAL} > 0$ for this class. No intersection occurred for the other labeled mask, leading to an output of 0. The unlabeled class, lacking a ground truth, is not evaluated by the class asymmetric loss, resulting in an output of 0. Adapted from [Sch22d].

where $\alpha \in [0, 1)$ is the weighting factor between the two individual losses. It should be noted that $\alpha = 1.0$ is not considered a valid value, as $\mathcal{L}_{CAL}$ only considers unlabeled data and hence cannot be used independently. This factor α is utilized to adjust the influence of the horizontal, channel-wise optimization of $\mathcal{L}_{DSC}$ and the vertical, probability distribution-wise optimization of $\mathcal{L}_{CAL}$.

The combined objective function $\mathcal{L}_\lambda$ is specifically designed to leverage heterogeneously labeled datasets. The calculation of $\mathcal{L}_\lambda$ is illustrated in Figure 5.2. It incorporates two individual loss functions that complement each other, each focusing on optimizing a distinct dimension of the output tensor, horizontal or vertical.

5.4 Results

5.4.1 Segmentation Mask Generation

The NeuBoot framework is assessed on two datasets: Droplet-DS and ISIC-DS (see Section 4.2). All evaluation metrics are computed using the ground truth annotations provided with each dataset. These annotations are not used during mask generation when applying the NeuBoot framework with thresholding methods (Otsu and k-means) or the NN U-Net, ensuring unsupervised segmentation.

Results for the Droplet-DS are displayed in Figure 5.3a. Using the U-Net improves the quality of segmentation masks across all metrics in all cases. Otsu's method and k-means produce similar pixel accuracy with a PA of 98% for both, indicating that both methods correctly classify a high percentage of pixels. However, the DSC and IoU metrics reveal that k-means slightly outperforms Otsu's method, with a DSC of 37% and IoU of 23% compared to Otsu's 34% and 21%. Interestingly, when the U-Net is trained on these initial segmentations, it shows a remarkable ability to improve upon them. The U-Net trained on Otsu's segmentations achieves higher scores with a DSC of 76%, IoU of 62%, and PA of 99%, compared to when trained on k-means segmentations, which yields a DSC of 71%, IoU of 56%, and PA of 99%. This suggests that the U-Net can learn more effectively from and correct the errors in Otsu's segmentations, despite their initially lower quality. The substantial increase in DSC and IoU for both cases indicates that the U-Net drastically enhances the segmentation compared to the ground truth.

Figure 5.3b presents the results for the ISIC-DS. The improvements here are less pronounced than with the Droplet-DS, likely due to the dataset's less homogeneous background (human skin) and RGB format, which pose greater challenges for

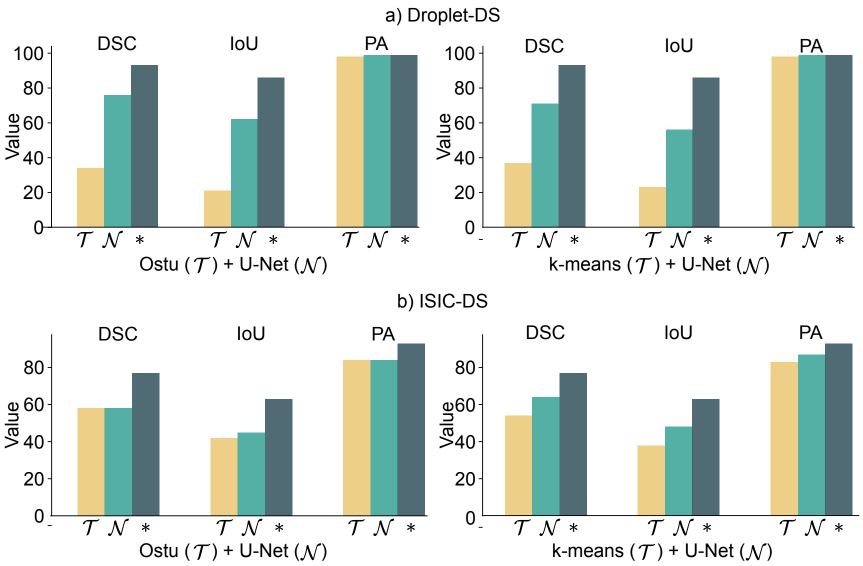


Figure 5.3: The evaluation results of segmentation mask generation using the NeuBoot framework concerning various metrics (DSC, IoU, and PA), using ground-truth masks as targets. All metrics are given as percentage values. $\mathcal{T}$ denotes the respective thresholding method (Otsu or k-means) and $\mathcal{N}$ the NN (U-Net) within the NeuBoot framework. The optimal achievable value is denoted as $*$, which is a U-Net trained with human annotations. Tabluated results are available in Appendix A.5.3.

segmentation. In this dataset, Otsu's method provides marginally better initial segmentations with a DSC of 58%, IoU of 42%, and PA of 84%, compared to k-means which achieves a DSC of 54%, IoU of 38%, and PA of 83%. However, the U-Net's ability to improve these segmentations differs between the two methods. Training the U-Net with Otsu-generated annotations yields minimal improvement, with the DSC remaining at 58% and only slight increases in IoU to 45% and PA to 84%. In contrast, k-means proves more beneficial for training, with notable improvements across all metrics: DSC increases to 64%, IoU to 48%, and PA to 87%. This suggests that for the ISIC-DS, the initial segmentations from k-means, despite being slightly less accurate, provide more informative features for the U-Net to learn from.

In Figure 5.4, the visual comparison of segmentation masks is shown. The U-Net remarkably refines initial segmentations for the Droplet-DS. In sample one,

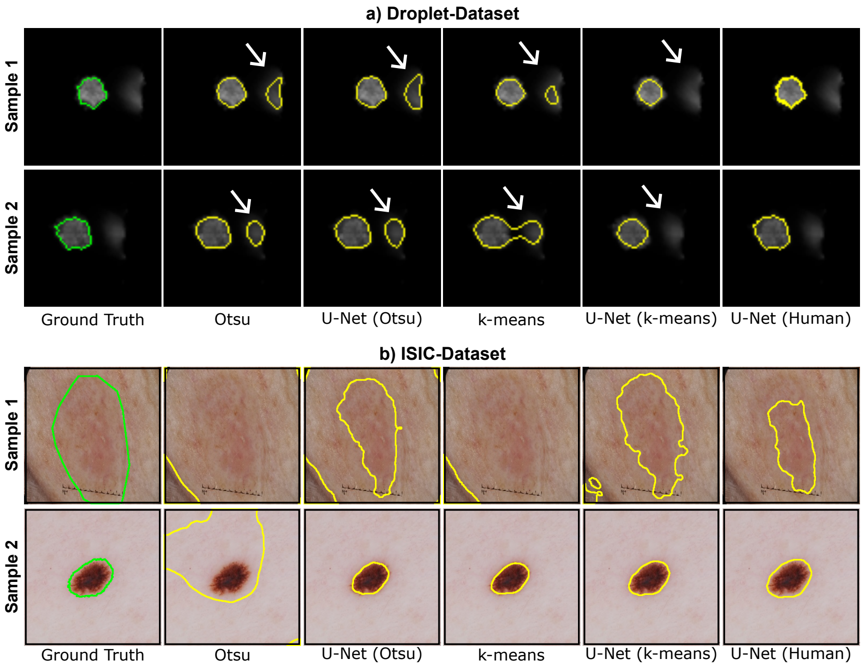


Figure 5.4: Two samples of both datasets with the learned segmentation masks. The annotations are given as contours over the respective sample. The ground truth is green, and the predicted masks are yellow. U-Net (*X*) denotes U-Net trained with annotations from method *X*. Adapted from [Ret22].

both the Otsu and k-means methods struggle with imaging reflections on the droplet's border, incorrectly including them in the segmentation. However, the U-Net, trained with k-means annotations, successfully recognizes and excludes these reflections, resulting in a more accurate boundary. The masks generated by Otsu do not appear to contain sufficient information for the U-Net to learn how to remove those erroneous reflections. Sample two of the Droplet-DS further highlights the U-Net's capabilities when combined with generated annotations. Again, reflections lead to wrong segmentation masks for both naïve approaches. The

U-Net trained with k-means annotations corrects this, producing a segmentation that closely matches the ground truth. Interestingly, the U-Net trained with Otsu annotations also improves the result, but fails to learn how to remove imaging reflections, as in sample one.

The improvements are equally noteworthy for the ISIC-DS but present different challenges. In sample one, both Otsu and k-means generate highly inaccurate initial segmentations, with neither method capturing the lesion. The U-Net, trained on either of these imperfect annotations, dramatically improves the segmentation, bringing it closer to the ground truth. This demonstrates the U-Net's ability to learn general features of skin lesions even from flawed initial segmentations. Sample two of the ISIC-DS showcases another interesting scenario. Here, Otsu's method largely fails to capture the lesion, while k-means provides a good initial segmentation. The U-Net trained on Otsu annotations manages to recover from this poor start, producing a segmentation that accurately captures the shape and location of the lesion. The U-Net trained on k-means annotations slightly refines the already decent initial segmentation, resulting in a mask that closely matches the ground truth. Across both datasets, it's striking how the U-Net consistently improves upon the initial segmentations, often correcting for systematic errors in the thresholding methods. This visual analysis corroborates the quantitative metrics and underscores the U-Net's potential in generating high-quality segmentation masks from imperfect, automatically generated annotations.

The proposed NeuBoot framework demonstrates remarkable effectiveness in producing high-quality masks from raw data, closely approaching the performance achievable with human-annotated training across both datasets. For the Droplet-DS, the U-Net trained with Otsu-generated annotations delivers robust and reliable results, establishing a strong foundation for segmentation tasks. Notably, the performance gap between generated and human annotations narrows further for the more complex ISIC-DS, highlighting the framework's adaptability to challenging real-world scenarios such as skin lesion segmentation. This underscores the potential of automated mask generation as a practical and scalable alternative to manual annotation for bootstrapping accurate segmentations in diverse domains. Integrating the NeuBoot framework with tools such as KaIDA

[Sch22c] can further streamline the annotation process, allowing users to refine or validate the generated masks efficiently and to leverage the full potential of algorithmic annotation support within established deep learning workflows.

5.4.2 Heterogenous Labeling

Four experiments are conducted to evaluate the effects of training with heterogeneous labels on the performance of multiclass semantic segmentation. In the first experiment, a general *ablation study* is performed by systematically removing labels across all classes. In the second experiment, expansion of a single-class dataset to include a second class is simulated. This setup is designed as a *transfer learning* task, where knowledge from the labels of the first class is used to aid learning for the new one. In the third experiment, the transfer learning scenario is modified by ensuring that each sample contains only one label mask at any time, directly simulating the process of *merging heterogeneously labeled datasets*. Finally, in the fourth experiment, the impact of the novel $\mathcal{L}_\lambda$ combined objective function's components is investigated by varying the weighting between the two loss terms, effectively *trading off horizontal with vertical loss*. All experiments are performed using the extended Heart-DS (see Section 4.3.1). Details about the training process are available in Appendix A.5.2.

Ablation Study: The first experiment assesses the ability of the developed methods to train on heterogeneously labeled data. Here, annotations from all classes are dropped equally with an increasing percentage. This general label ablation study does not specify a particular use case. However, it evaluates how dropped labels influence overall performance compared to the baseline of supervised training on the fully labeled dataset (dropped labels: 0%). The results displayed in Figure 5.5 show that there is no major decrease in performance up until about 60% dropped labels according to both the mean IoU and mean DSC over both classes, which means that over half of the annotations available do not provide notable additional knowledge for the NN in this case. The mean ψ_{PFR} metric also progresses similarly, indicating that 60% dropped labels provides

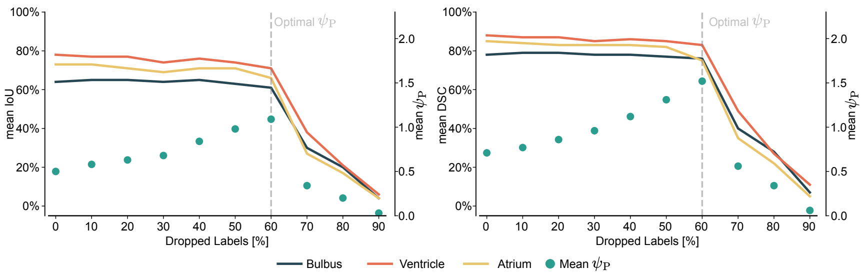


Figure 5.5: Performance results of the three classes in Heart-DS if increasing numbers of annotations are randomly removed from the training samples (Dropped Labels). The development of the mean IoU and mean DSC for both classes are plotted side by side, while the mean ψ_{PFR} is represented as scatter points. A vertical dotted line indicates the optimal mean ψ_{PFR}, where the trade-off between dropped labels and performance is optimal. The full tabulated results are detailed in Table A.3. Adapted from [Sch22d].

the optimal balance between performance and the number of annotated samples. After that, performance declines rapidly. At 70% removed labels, the network converges to a state where the background class is assumed to be the correct class for all pixels. It is believed that this behavior occurs because the background class is a composition of the other classes. Hence, it appears more frequently when few classes are labeled, which implicitly increases its importance during training. It is also apparent from the results that the *Bulbus* class is the hardest to segment, which is expected since the cardiac outflow tract is not as clearly delineated as the *Atrium* and *Ventricle*.

The robustness of the model trained with heterogeneous labels is further highlighted in Figure 5.6. Despite the noticeable reduction in labeled data in the *Ablation 60* row (60% of annotations removed), where only 40% of labels are available, the model still achieves a remarkably close segmentation performance to the fully labeled *Baseline* row. While convergence is slower and predictions are initially noisier, by *Epoch 50*, the segmentation masks produced by the *Ablation 60* training capture many of the key structural features present in the *Baseline* results. In detail, distinguishing between the *Ventricle* and *Atrium* classes is initially challenging. However, as training progresses, the model progressively improves segmentation accuracy, and by the final stages, the segmentation quality almost matches that of the *Baseline* method. This demonstrates that the model

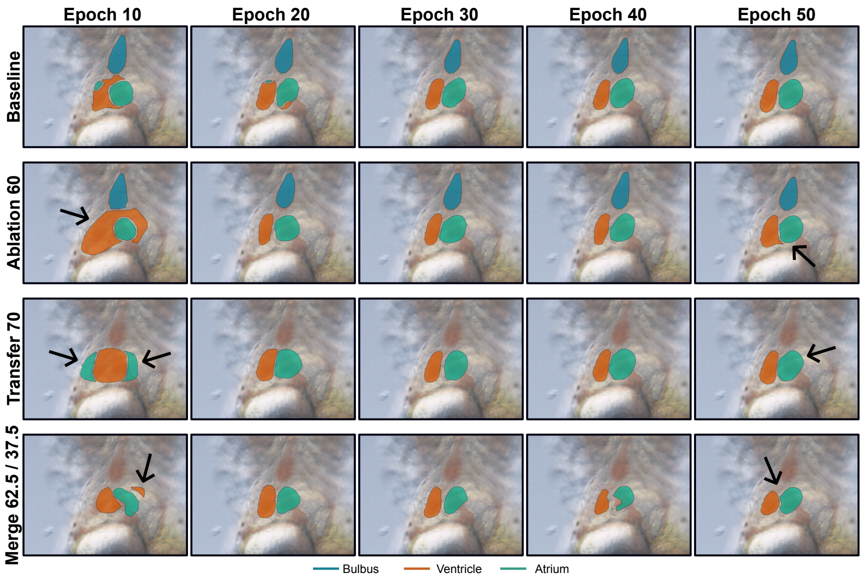


Figure 5.6: One sample of generated masks for heterogeneous training on the Heart-DS. The first row represents the baseline, where all annotations are provided for every sample. The second row illustrates the ablation experiment, where 60% of labels are randomly dropped, followed by the transfer learning experiment, which removes 70% of labels from the atrium class. The fourth row depicts the merge experiment, in which every sample contains only one class. In this case, 62.5% of samples contain the atrium class, while 37.5% contain the ventricle class. Adapted from [Sch22d].

generalizes effectively and produces meaningful segmentations even with substantially reduced labeled data. The results highlight the network's resilience to label sparsity, suggesting it can leverage partial supervision to achieve competitive performance. Furthermore, the findings demonstrate that the CoHeG framework, combined with the ψ_{PFR} metric, enable the determination of the optimal required annotation rate when using the novel combined objective function $\mathcal{L}_{\lambda}$, reducing annotation costs while maintaining high-quality segmentations. This approach is promising for real-world applications where fully labeled datasets may not always be feasible.

Transfer Learning: The second experiment considers a scenario where a dataset is initially fully labeled with a single class and is subsequently expanded to include a second one. Existing samples annotated with the first class are partially extended with the new one, creating a heterogeneously annotated dataset. Unlike the first experiment, one class is homogeneously labeled. In this case, the *Ventricle* class is consistently fully available, while the *Atrium* class contains varying annotations to simulate incomplete labeling processes. In this experiment, the dataset is limited to only two classes: *Atrium* and *Ventricle*. As seen in Figure 5.7, the performance of the *Ventricle* class remains relatively stable. However, the *Atrium* class exhibits a gradual decline in performance as the number of dropped labels increases. If 40% of the *Atrium* class labels are removed from the dataset, a performance reduction of 1% in DSC and 2% in IoU is observed. Interestingly, when only 60% of the *Atrium* samples are labeled, there is only a slight decrease in performance compared to labeling all samples.

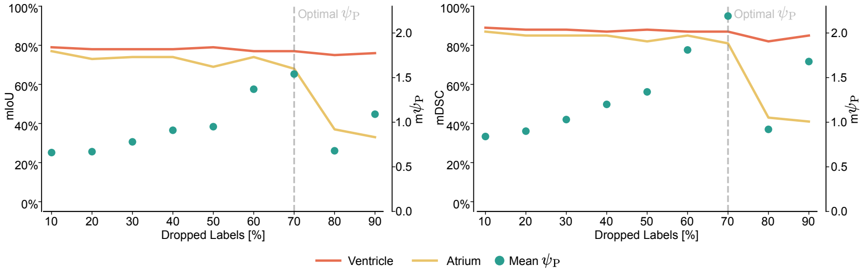


Figure 5.7: Transfer learning results for the ventricle class and atrium class of the Heart-DS. Here, the ventricle class is always fully labeled, and the atrium class is only contained in the number of samples specified by the Dropped Labels. The development of the mean IoU and mean DSC for both classes are plotted side by side, while the mean ψ_{PFR} is represented as scatter points. The extensive results are given in Table A.4. Adapted from [Sch22d].

The qualitative results in Figure 5.6 reveal that while the *Ventricle* segmentation remains largely intact, the *Atrium* segmentation deteriorates when 70% of its labels are removed. Compared to the *Baseline* row, where both classes are fully labeled, the *Atrium* segmentation in *Transfer 70* (70% of *Atrium* annotations removed) exhibits less defined boundaries and a diminished ability to capture the organ's shape precisely, especially in the early stages of training. Despite this degradation, the network can still identify the *Atrium*'s location and structural context. This suggests that the network benefits from the spatial and anatomical context provided by the fully annotated *Ventricle*, which helps compensate for the scarcity of *Atrium* labels. The results demonstrate that transfer learning with the CoHeG approach and the $\mathcal{L}_\lambda$ objective function is effective. The network uses the fully labeled *Ventricle* data to learn spatial relationships and anatomical priors, which enables accurate *Atrium* predictions despite limited labels and sparse direct supervision.

Merging Heterogeneous Labeled Datasets: The third experiment simulates a scenario in which two datasets are sampled from the same domain but feature differently labeled classes, like combining retinal scans where one segments blood vessels and another annotates optic discs into a unified dataset. In this setup, two datasets with annotations for distinct classes are merged without additional labeling. To achieve this, the initial dataset is divided into two separate ones: one containing only the *Ventricle* annotations and the other the *Atrium* ones. Only the *Ventricle* and *Atrium* classes are considered in this experiment. To evaluate the synergy effects of merging heterogeneous datasets, the dataset is split into various proportions (e.g., 62.5%/37.5%, where the *Ventricle* dataset contributes 62.5% of the total samples while the *Atrium* dataset contributes 37.5%). Although substantial differences in dataset sizes arise during this experiment, both datasets are sampled continuously in equal amounts. Oversampling is applied to achieve an equivalent sampling rate, meaning that samples from the minority class are randomly selected multiple times. Interestingly, Figure 5.8 reveals a discrepancy between the IoU and DSC metrics. While IoU demonstrates the expected behavior, peaking at the balanced 50%/50% split where both the *Atrium* and *Ventricle* have equal representation, DSC shows a less pronounced peak. This suggests that although the overall overlap (as measured by DSC) remains relatively consistent across different dataset ratios, the boundary accuracy (as measured by IoU) is particularly sensitive to a balanced representation of both classes. Furthermore, even when the overall overlap (DSC) seems sufficient, combining two distinct datasets drastically impacts the precise segmentation boundaries (IoU).

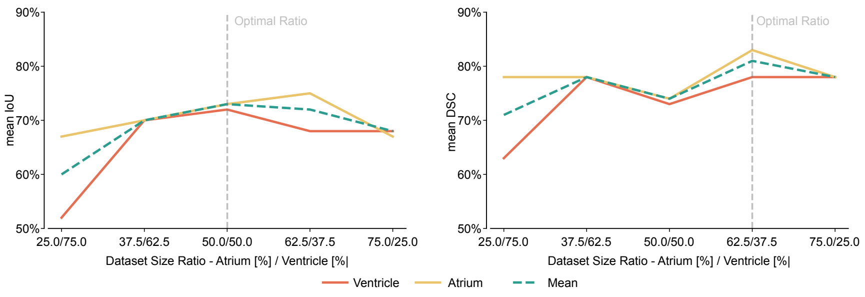


Figure 5.8: Results for merging heterogeneously labeled datasets. The ventricle and atrium classes of the Heart-DS are considered for this experiment. The development of the mean IoU and mean DSC for both individual classes and the mean for the two combined. The dataset size ratios are shown on the x-axis, and the optimal split rate, corresponding to the highest score for the respective metric, is represented by a vertical dashed line. The complete tabulated results results are provided in Table A.5. Adapted from [Sch22d].

The previous observations are visually confirmed in Figure 5.6 at the *Merge 62.5 / 37.5* row (62.5% *Ventricle* and 37.5% *Atrium* annotations), where the segmentations are somewhat less precise compared to the *Baseline* (where all labels are present), yet remain relatively well-preserved given that the network has never encountered both classes together during training. Especially at the beginning of the training, the two heart segments are segmented in a scattered and implausible way, which improves substantially as the training progresses. This indicates effective knowledge transfer within the network, utilizing the anatomical context to infer the presence of the unlabeled structure. This experiment demonstrates that merging two datasets with the CoHeG framework stemming from the same domain is feasible and yields reasonable segmentation results, even when only samples with one class each are available. This showcases the ability of the CoHeG framework and the novel $\mathcal{L}_\lambda$ objective function to learn from limited, differently labeled data. Interestingly, the performance metrics do not drop much, even when the imbalance is substantial.

Trading Off Horizontal With Vertical Loss: The fourth experiment examines the contributions of the two individual losses that create the combined $\mathcal{L}_\lambda$ objective function proposed in this work. The same test setup as described in the *transfer learning* experiment is used to ensure comparability. Thus, the dataset is restricted to only the *Ventricle* and *Atrium* classes, with intentionally removed labels for the *Atrium* class. The proportion of removed labels is fixed at 50%, meaning that half of the *Atrium* labels are removed. The weighting factor α is adjusted (as defined in Equation (5.6)) to explore how each loss function influences the learning process.

The results in Figure 5.9, evaluated using the mean IoU and mean DSC, indicate that performance remains constant as the weight assigned to $\mathcal{L}_{CAL}$ increases, with a small peak in the *Mean* value around $\alpha = 0.4$. Beyond this point, performance

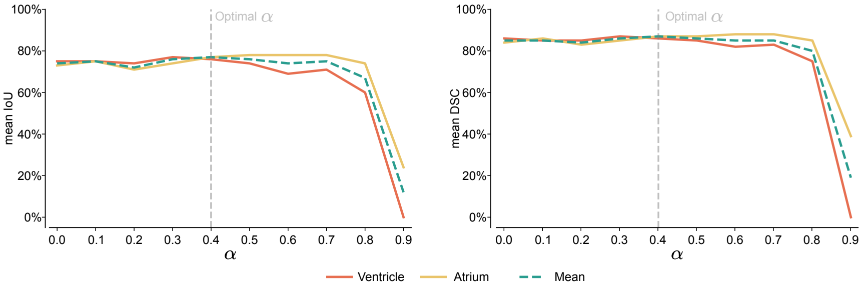


Figure 5.9: Results for estimating the optimal value for the weighting term α of the combined objective function defined in Equation (5.6). The development of the mean IoU and mean DSC for both individual classes and the mean for the two combined. Values for the α term are shown on the x-axis, and the optimal value is given as a vertical dashed line. The complete results are provided in Table A.6. Adapted from [Sch22d].

for both metrics begins to decline slightly, with a noticeable drop in performance after α values exceed 0.8. While performance remains relatively stable across a range of α values, careful selection still improves outcomes by optimally combining the complementary strengths of both loss functions. Specifically, the performance of the *Atrium* class, which is subject to label dropping, continues to improve slightly even after $\alpha = 0.4$. This implies that the $\mathcal{L}_{CAL}$ loss, which targets leveraging information from unlabeled regions, is especially beneficial for classes with incomplete annotations, enabling the network to better estimate the boundaries and spatial context of the *Atrium* despite the missing labels. However, after this threshold ($\alpha = 0.4$), the performance of the *Ventricle* class, which is fully labeled, begins to decrease. This decline indicates that an over-reliance on $\mathcal{L}_{CAL}$ can negatively impact the overall segmentation accuracy. This is likely because increasing α places greater emphasis on enforcing consistency in unlabeled regions, which may reduce the ability of the model to accurately fit the fully labeled *Ventricle* data. The drop in performance for the *Ventricle* class at higher α values also suggests possible conflicts between the learning signals provided by $\mathcal{L}_{DSC}$ and $\mathcal{L}_{CAL}$. When $\mathcal{L}_{CAL}$ is overly weighted, it leads the network toward globally consistent solutions across labeled and unlabeled regions but may not accurately segment the *Ventricle*. Optimal performance is observed when both loss functions are balanced and contribute approximately equally, around $\alpha = 0.4$. This balance allows the network to effectively utilize both the explicit supervision provided by labeled data (through $\mathcal{L}_{DSC}$) and the implicit information gleaned from unlabeled regions (through $\mathcal{L}_{CAL}$). These results show that while optimal performance is achieved near $\alpha = 0.4$, segmentation remains robust across a broad range of α values. Thus, although fine-tuning α provides performance benefits, the novel combined objective function $\mathcal{L}_{\lambda}$ is not overly sensitive to the precise weighting, simplifying practical deployment. Further examination of adaptive strategies for dynamically adjusting α during training could potentially yield even better results by enabling the network to automatically fine-tune the balance between $\mathcal{L}_{DSC}$ and $\mathcal{L}_{CAL}$ based on the specific characteristics of the training data and the current learning stage.

5.5 Discussion

The results in this chapter demonstrate the potential of algorithmic dataset enhancement techniques to address key challenges in DL for image segmentation, especially with scarce or heterogeneous annotations. The novel NeuBoot framework combines unsupervised thresholding with NNs for generating segmentation masks, thereby eliminating human involvement in the initial mask creation process and reducing annotation costs. This approach adapts well across datasets with varying characteristics, although its performance depends on the image complexity and the initial quality of the mask. Similarly, the proposed combined objective function $\mathcal{L}_\lambda$, integrated into the introduced CoHeG framework and evaluated using the novel ψ_{PFR} metric, robustly handles heterogeneous labeling. Missing annotations are addressed by leveraging fully labeled classes to enhance segmentation in partially labeled regions and facilitate dataset merging, thereby improving generalization without major performance trade-offs.

These findings have considerable implications for research and practical applications. In many domains, such as biomedicine, obtaining accurate segmentation masks is a substantial bottleneck because it requires highly trained experts, who are often unwilling or unavailable to dedicate time to tedious annotation tasks. Automating segmentation mask generation and enabling training on incomplete labels directly addresses this challenge, lowering barriers to entry for DL in resource-constrained settings. Training on heterogeneously labeled datasets further enhances model robustness by exposing it to diverse label distributions, improving generalization across tasks. These methods broaden the applicability of DL to domains like medical imaging or environmental monitoring, where high-quality annotations are often difficult and expensive to obtain. Moreover, integrating contextual information, spatial information from images, and anatomical information from labels drastically improves segmentation performance under suboptimal labeling conditions.

Some limitations remain: unsupervised methods for generating initial masks may struggle with highly complex or noisy datasets. The combined background class can cause issues in heterogeneous labeling when only a few labels are available for certain classes. Additionally, the reliance on synthetic heterogeneous datasets may not fully reflect real-world complexities, and evaluation is limited to a few datasets. Future work should prioritize advanced methods for generating initial masks, such as active learning or semi-supervised approaches that iteratively refine masks with minimal human input. Expanding evaluations to real-world heterogeneous datasets will provide more substantial evidence of generalizability. Adaptive strategies for dynamically adjusting the weighting factor during training could enhance performance without extensive hyperparameter tuning. Ultimately, testing these methods on larger and more diverse datasets will enable the evaluation of their scalability and robustness across different imaging modalities.

6 Deep Learning With Unannotated Data

Parts of this chapter are based on:

- L. Rettenberger, M. Schilling, S. Elser, M. Böhland, and M. Reischl. Self-supervised learning for annotation efficient biomedical image segmentation. *IEEE Transactions on Biomedical Engineering*, IEEE, 2023. DOI: 10.1109/TBME.2023.3252889.
- L. Rettenberger, N. J. Szymanski, A. Giunto, O. Dartsi, A. Jain, G. Ceder, V. Hagenmeyer, and M. Reischl. Leveraging unlabeled SEM datasets with self-supervised learning for enhanced particle segmentation. *npj Computational Materials*, volume 11, page 289, Nature Publishing Group UK London, 2025. DOI: 10.1038/s41524-025-01802-3.

6.1 Overview

Recent trends in ML emphasize the significance of large, carefully labeled datasets for achieving state-of-the-art results. The ImageNet dataset [Rus15], which contains over 14 million annotated images, exemplifies this, as systems pretrained on it consistently exhibit remarkable performance in solving tasks, even in domains unrelated to its content [Xie18, Mat21]. This transferability suggests that general task-solving models may be possible if extensive annotated training samples are available. However, the substantial effort required for annotating such datasets is impractical, especially in fields where labeling demands expert knowledge and considerable effort [Chi20, OMa]. Additionally, as automation advances in many domains, large volumes of data can often be generated automatically, leading to vast amounts of unused data, as most ML methods are supervised. To address this, new methods aim to reduce dependency on labeled data. One of the most promising approaches here is Self-SL, which identifies patterns in unannotated data to build a knowledge base, improving performance even with limited annotated samples. This approach offers several benefits. For example, it enhances the robustness of ML systems with task-agnostic knowledge and reduces the reliance on annotations [Zbo21]. Despite its great potential, Self-SL lacks applications and solutions tailored for small-scale datasets and real-world applicability. This chapter provides methodological frameworks for practitioners facing challenges when employing Self-SL, such as determining the number of annotated samples needed, selecting a suitable dataset and model size, and meeting hardware requirements. Furthermore, multiple novel metrics are introduced to reliably evaluate Self-SL methods in practical applications, enabling applicability and providing an all-encompassing overview of Self-SL approaches.

6.2 Related Works

Currently, computer vision tasks rely heavily on SL, which constrains the development of intelligent, generalist models due to the laborious nature of dataset annotation [LeC21]. This issue is particularly pronounced in segmentation tasks, where pixel-wise labeling is prone to errors and subjectivity [Xu19]. Moreover, the performance of ML systems scales sublinearly with dataset size, leading to diminishing returns [Sun17]. An ML system trained via SL to distinguish cats from dogs is unlikely to detect elephants, unlike a human child who can easily identify unknown animals. This is because the human brain develops a generalized world model for problem-solving [LeC21]. Self-SL is considered promising for establishing such background knowledge in ML, as it identifies inherent structures in samples rather than task-specific characteristics [Kol19]. Introductions to the general Self-SL paradigm and its role in DL are provided in Section 3.1.2 and Section 3.3.2.

Contrastive Learning: With several notable methods, the CL paradigm is a dominant approach in Self-SL literature. Momentum Contrast (MoCo) uses a query and momentum encoder, along with an encoding queue, to access diverse samples during training [He20]. Simple Contrastive Learning (SimCLR) incorporates Multi-Layer Perceptrons (MLPs) projection heads to map views into a lower-dimensional space, utilizing a shared encoder [Che20a]. Bootstrap Your Own Latent (BYOL) prevents model collapse by defining parameters as an exponential moving average, eliminating the need for dissimilar samples [Gri20]. Barlow Twins aims to align similar samples and decorrelate dissimilar ones by approximating the cross-correlation matrix to the identity matrix [Zbo21]. Maintaining semantic consistency is crucial for dense prediction tasks like semantic segmentation and object detection. Hence, Dense Contrastive Learning (DenseCL) and Detection Contrast (DetCo) extend MoCo to enhance dense predictions by utilizing multiple projection heads [Wan21b, Xie21]. DenseCL and DetCo have demonstrated effectiveness in dense prediction tasks, while SimCLR and MoCo have shown success in small-scale datasets [Wan21b, Xie21, Xu21, Zhe22]. Barlow Twins and BYOL have not been evaluated on dense predictions or small-scale datasets, and current studies rely on large datasets, typically assuming millions of samples are available, which is not feasible in real-world scenarios. Furthermore, the impact of emphasizing semantic consistency in these methods on their practical applicability remains largely unexplored. Also, there exist studies concluding that freezing the learned parameters in the pretext for downstream task training is a viable option, as this reduces training effort and the number of required annotated samples [Isi20], which is largely unexplored for most Self-SL approaches. Although many approaches have been developed, no universal framework exists for comparing Self-SL methods. This is a considerable challenge since Self-SL involves a complex, multi-step learning process that requires structured and in-depth comparisons if an optimal method for a challenge is sought. Evaluating the effectiveness of different methods is difficult for several reasons: the learned knowledge is not easily interpretable, errors introduced during individual steps are often hard to detect, and substantial computational resources are required to implement Self-SL effectively.

Masked Autoencoders: While CL has been widely adopted, Masked-AE-based pretraining has gained traction alongside the rise of ViTs [Dos20, He22]. The architectural compatibility between Masked-AEs and ViTs has led to promising results, whereas CNNs have faced challenges in implementing similar masking techniques effectively [Zha22a]. The introduction of ConvNeXtV2[1] has enabled Masked-AE-based Self-SL pretraining with CNNs, potentially offering advantages over other Self-SL methods. However, the evaluation of ConvNeXtV2 has been limited in scope, focusing primarily on classification performance on the ImageNet dataset and comparison with a single CL method [Woo23]. Despite the great potential of ConvNeXtV2, this narrow evaluation reveals a broader gap in Self-SL research: the lack of comprehensive frameworks for assessing and optimizing Self-SL methods across various tasks and datasets. Current Self-SL evaluations often use fixed configurations for large benchmark pretraining datasets, which overlooks the possibility that different methods may require varying amounts of data for optimal performance. Moreover, the availability of multiple model sizes for many Self-SL methods, which can be tailored to specific tasks and computational constraints, highlights the need for research into selecting the most appropriate variations. Existing studies prioritize benchmark performance metrics over operational parameters, such as optimal trainable parameters or pretraining data thresholds [Gui24, Ohr21]. The absence of adaptive evaluation protocols that can optimize essential configuration parameters while balancing model capacity and data requirements hinders the acceptance and integration of novel Self-SL methods.

In summary, there is an urgent need for systematic frameworks to evaluate and compare Self-SL approaches, which are crucial for ensuring fair, reproducible, and meaningful assessments, as well as for optimizing Self-SL methods for specific

[1] Details for the ConvNeXtV2 architecture are introduced in Section 3.3.3.

applications. To draw reliable conclusions about the effectiveness and practicality of different approaches, evaluation metrics must comprehensively account for real-world factors, such as dataset size, annotation budget, and hardware requirements, all within unified evaluation systems. Without this transparency and standardization, researchers and practitioners struggle to identify methods that meet real-world constraints, significantly hindering progress.

6.3 Methods

This section introduces novel metrics and frameworks for evaluating Self-SL methods in data-constrained real-world scenarios. It addresses critical gaps by introducing several metrics that quantify hardware and annotation requirements, which are integrated into the novel Self-SL Benchmarking Framework (SelfSLBench) for comparing the strengths and weaknesses of different approaches. This is demonstrated by identifying the most suitable CL method for biomedical instance segmentation, where data is limited and annotation costs are high. Building on this, an in-depth evaluation protocol called Self-SL Automatic Tuning Framework (SelfSLAutoTune) is also introduced to analyze performance across varying dataset sizes and annotation levels. The framework's effectiveness is demonstrated by comparing the best-performing CL method from Section 6.3.2 with the novel ConvNeXtV2 approach on challenging SEM data, thereby positioning the Masked-AE paradigm in relation to CL in resource-constrained domains. Together, these tools enable efficient selection and optimization of Self-SL methods for specific tasks.

6.3.1 Novel Evaluation Metrics

The introduction of three new metrics addresses the inadequacies of current evaluation techniques, which fail to sufficiently quantify Self-SL methods while considering the provided annotations.

Neighborhood Quality Criterion (ψ_{NQC}): The ψ_{NQC} metric evaluates how samples from the same class effectively cluster together for a Self-SL approach, quantifying sample neighborhood structure within a dataset. This novel metric is crucial because it offers insights into the organization of the latent space, essential for understanding how Self-SL methods learn.

Observing a vector $\boldsymbol{x}$ of a dataset $\mathcal{D}$, the nearest $D_{\text{n}}()$ and farthest $D_{\text{f}}()$ neighbors using the Euclidean distance $\mathrm{d}()$ are determined as:

$$D_{\text{n}}(\boldsymbol{x}) = \underset{\boldsymbol{k}\in\mathcal{D}}{\text{argmin}}\{\mathrm{d}(\boldsymbol{x},\boldsymbol{k})\},$$
$$D_{\text{f}}(\boldsymbol{x}) = \underset{\boldsymbol{k}\in\mathcal{D}}{\text{argmax}}\{\mathrm{d}(\boldsymbol{x},\boldsymbol{k})\}.$$

Samples from the same class are expected to cluster together and remain distant from dissimilar samples [Jai20], thereby providing valuable insights into the space spanned by $\mathcal{D}$. Neighborhood evaluation is particularly useful in the latent space representation of samples of ML models. Building on this, the ψ_{NQC} is introduced to summarize the quality of the neighborhood in a single value. For each sample $\boldsymbol{x}_i \in \mathcal{D}$, it is evaluated whether the nearest neighbor $D_{\text{n}}(\boldsymbol{x}_i)$ belongs to the same class as $\boldsymbol{x}_i$ or not. If $D_{\text{n}}(\boldsymbol{x}_i)$ and $\boldsymbol{x}_i$ are of the same class, 1 is yielded and 0 otherwise. ψ_{NQC} then sums over all entries in $\mathcal{D}$ and divides by the number of total samples:

$$\psi_{\text{NQC}} = \frac{1}{l}\sum_{i=1}^{l}\begin{cases}1, & \text{if } \boldsymbol{x}_i \text{ and } D_{\text{n}}(\boldsymbol{x}_i) \text{ are of same class}\\ 0, & \text{otherwise,}\end{cases}$$

where l represents the number of samples in $\mathcal{D}$. For ψ_{NQC} to be applicable, $\mathcal{D}$ needs to contain classification annotations. Further, since ψ_{NQC} is influenced by the number of classes in $\mathcal{D}$, it is suitable for comparing different representation spaces but not different datasets. Since feature spaces within DL are usually high-dimensional, a Principal Component Analysis (PCA) [Abd10] might be utilized before evaluating $D_{\text{n}}()$, which projects the representations into a low-dimensional feature subspace to mitigate the curse of dimensionality [Keo11]. The maximum number of dimensions for Euclidean distance to be applicable depends on the specific application and data characteristics.

Assuming a random representation distribution for a dataset $\mathcal{D}$ of length l with classification classes $\mathcal{C}$, the expected value $\mathbb{E}_{\text{random}}$ of ψ_{NQC} is computed as the sum of the probability P() multiplied by the number of samples for each class:

$$\mathbb{E}_{\text{random}}\left[\psi_{\text{NQC}}\right] = \frac{1}{l}\sum_{c\in\mathcal{C}} \mathrm{P}(c)|c|, \tag{6.1}$$

where c represents a class within the set of classes $\mathcal{C}$, and $|c|$ denotes the number of samples in class c.

Runtime Quality Criterion (ψ_{RQC}): The ψ_{RQC} measure systematically evaluates the temporal efficiency of Self-SL methods to address and compare the considerable computational resources required for Self-SL approaches. This metric is essential because it enables the comparison of different methods based on their training time, which is critical for real-world applications with limited computational resources.

The ψ_{RQC} quantifies the training duration by utilizing the point at which loss values stabilize (convergence) within the non-convex objective function. The time t() required for a method δ to reach convergence is evaluated in relation to the method that converges the quickest, denoted as t*:

$$\psi_{\text{RQC}} = \frac{\mathrm{t}(\delta)}{\mathrm{t}^*}. \tag{6.2}$$

Integrated Quality Criterion (ψ_{IQC}): The ψ_{IQC} metric assesses the performance of Self-SL methods across various labeling rates, addressing the challenge of evaluating their effectiveness under different annotation scenarios. This metric is crucial because it evaluates the performance of downstream tasks while considering the number of annotations available, providing a comprehensive view of how Self-SL methods perform with limited annotations.

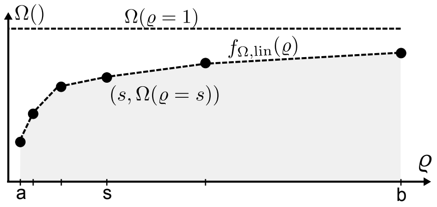


Figure 6.1: Illustration of the ψ_{IQC} metric. The interpolating function $f_{\Omega,\mathrm{lin}}(\varrho)$ is integrated over the boundaries a and b. This integral is normalized by the product of the interval length $(b - a)$ and the maximum quality value $\Omega(\varrho = 1)$. An example annotation rate s with $\Omega(\varrho = s)$ is displayed. Adapted from [Ret23b].

In the following, the quality criterion $\Omega()$ can be metrics such as the AJI$^+$ [Gra19, Kum19] or the DSC [Sud17]. The annotation rate is denoted by $0.0 \leq \varrho \leq 1.0$, where $\varrho = 0.0$ indicates no available annotations, and $\varrho = 1.0$ signifies full annotation. Let a and b represent the minimum and maximum annotation rates considered. All annotation rates are applied to $\Omega()$ to obtain the measurements $\mathrm{P} = \{\Omega(a), \ldots, \Omega(b)\}$. The values within P are then linearly interpolated to derive a continuous function $f_{\Omega,\mathrm{lin}}(\varrho)$. The area of $f_{\Omega,\mathrm{lin}}(\varrho)$ is then the ψ_{IQC}. The construction of $f_{\Omega,\mathrm{lin}}(\varrho)$ and the overall ψ_{IQC} are illustrated in Figure 6.1. Mathematically, to describe the overall quality concerning various annotation rates ϱ, the ψ_{IQC} formula is introduced as:

$$\psi_{\mathrm{IQC}} = \frac{1}{\Omega(\varrho = 1)(b - a)} \int_a^b f_{\Omega,\mathrm{lin}}(\varrho)\mathrm{d}\varrho.$$

Given that the quality achieved with a fully annotated dataset $\Omega(\varrho = 1)$ is assumed to be the maximum value, the integral is normalized by the product of $\Omega(\varrho = 1)$ and the interval length $b - a$ (the maximum possible area).

6.3.2 Self-SL Comparison Framework

The SelfSLBench framework, designed for comparing Self-SL methods in challenging, data-constrained tasks, is introduced to provide a structured and informed approach for comparing Self-SL methods. Generally, within Self-SL, experiments are structured in two stages: the first stage focuses on pretext tasks, while the second stage applies the learned representations to downstream tasks. Hence, a division between those two stages is established, and both stages are evaluated separately and then in combination. The proposed SelfSLBench framework is developed for EDN networks for segmentation tasks, but can be modified for any NN architecture.

Pretext Task: In the first step, a collection of pretext tasks $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3, \dots\}$ is combined with unannotated datasets $\mathcal{D}^{\mathrm{u}} = \{\mathcal{D}^{\mathrm{u}}_1, \mathcal{D}^{\mathrm{u}}_2, \mathcal{D}^{\mathrm{u}}_3, \dots\}$ to develop useful representations and parameterize the encoder f_θ of the EDN. The parameters θ that are learned are evaluated independently from the subsequent downstream task using various metrics. The SelfSLBench framework encompasses assessments of both hardware requirements and application overhead. This includes evaluating factors such as the number of hyperparameters needing adjustment, the implementation effort required, the necessary batch size, and the number of trainable parameters to be learned.

Downstream Task: In the second stage, the learned representations are utilized in downstream tasks $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \dots\}$ to assess performance on real-world segmentation challenges. For this purpose, $\mathcal{D}^{\mathrm{u}}$ is augmented with corresponding annotations $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \dots\}$, and the provided parameters f_θ are used as initialization for the encoder part of the EDN. The parameters of the decoder are randomly initialized. As Self-SL is particularly interesting for datasets with partial annotations, the performance is observed based on the number of annotated samples. Multiple metrics are used to evaluate the downstream tasks to assess performance on each dataset $\mathcal{D}^u_j$ with annotations $\mathcal{A}_j$, a pretext task $\mathcal{P}_i$, and a downstream task $\mathcal{D}_h$, depending on the number of available annotations $\mathcal{R} \subset \mathcal{A}_j$. The annotation rates are defined as $2^s/100$, where $s = 0, \dots, 6$, with powers of two used to focus on scenarios with few annotations.

To demonstrate the applicability of the SelfSLBench framework, a systematic comparison is conducted with the state-of-the-art in CL, which is the prevailing Self-SL paradigm with numerous different methods. This comparison employs a range of approaches and utilizes datasets of varying sizes, from small to medium-sized, for challenging segmentation tasks. It uses multiple metrics, including those introduced in this work, to assess the applicability of the most popular methods. *ImageNet* pretraining is used as a baseline for the experiments. Furthermore, a simple *Autoencoder*, representing the most basic form of representation learning, is trained for comparison to CL. For the CL methods, a combination of classical methods (*SimCLR*, *Barlow Twins*, *BYOL*, and *MoCo*) along with the latest approaches designed for dense tasks (*DenseCL* and *DetCo*) is employed to provide a comprehensive overview. For the conducted experiments, *No Pretraining* indicates that the parameters of the NN are randomly initialized [He15].

6.3.3 Self-SL Optimization Framework

The novel SelfSLAutoTune framework builds upon the SelfSLBench by providing investigations that offer decision support for practitioners regarding a specific Self-SL method. The emphasis is on the applicability and optimal implementation in challenging, real-world scenarios. The overall experimental setup is divided into two main parts.

Comparative Performance Evaluation: The SelfSLAutoTune framework assesses the optimal model size, specifically the number of trainable parameters required for a given dataset. This is crucial since Self-SL methods are often available at multiple scales. Hence, there are considerable differences in training speed and applicability.

Dataset Size Analysis: The dataset size analysis builds on the results of the comparative performance evaluation by estimating the optimal required dataset size for the optimized Self-SL method, calculating how much data is needed to utilize the architecture effectively and achieve the best possible results. These evaluations collectively reveal how a specific Self-SL method can be applied optimally and efficiently to balance resource requirements with performance. As the influence of label ratios is investigated with the SelfSLBench framework, it is possible to evaluate the potential of Self-SL under different sample volumes in the pretext task. Hence, an ablation study is conducted in which the number of overall samples within the inspected dataset is reduced to evaluate how the performance varies. As with the annotation rates, the powers of two are used to calculate the different number of samples within the reduced datasets, focusing on small datasets, and recognizing whether drastically more samples are associated with higher performance. In detail, the dataset size is calculated as $2^s/100$, where $s = 0, \ldots, 6$. With this, the findings of the SelfSLBench regarding the label ratios are enriched with a Self-SL-focused evaluation of the available samples for the pretraining task.

To demonstrate the applicability of the SelfSLAutoTune framework, the novel ConvNeXtV2 is compared with the best-performing CL method identified using the SelfSLBench. Further, ImageNet pretraining is employed as a comparative baseline. ConvNeXtV2 is available in multiple sizes ranging from roughly 3.7 million to about 660 million parameters and uses ConvNeXt [Liu22] as backbone, which is a considerable improvement in terms of performance compared to the ResNet and accounts for a large portion of the overall model size [Liu22][2]. Hence, the best-performing CL method and ImageNet pretraining are also employed with the different scales of the ConvNeXt backbone to enable a fair comparison. This provides insights into the best approach and how well CL and ImageNet can utilize the novel backbone. The comparison focuses on instance segmentation with semi-sized datasets, as this task is the most complex within detection. Furthermore, as the models range from 3.7 million parameters for the smallest model to 660 million parameters for the largest one, the experiments also provide a comprehensive estimation of the number of learnable parameters required for Self-SL methods using the novel ConvNeXt backbone and which approach can utilize it most effectively.

[2] In Appendix A.6.4, the different backbone and ConvNeXtV2 model sizes are listed, demonstrating that the backbone constitutes the largest portion of the ConvNeXtV2 model.

6.4 Results

In this section, the results of evaluations on extracting knowledge from unannotated data are presented. Initially, results obtained with the SelfSLBench framework, where CL methods are applied to small- and medium-sized datasets to generate segmentation masks, are presented. Building upon these findings, the SelfSLAutoTune framework is evaluated using an additional dataset from a different domain with a larger number of samples, allowing novel Masked-AE approaches within Self-SL to be assessed and compared against the prevailing CL paradigm. Details regarding NN training and hyperparameter tuning are provided in Appendix A.6.

6.4.1 Self-SL Comparison

6.4.1.1 Pretext Comparison

For the experiments involving the SelfSLBench framework, the ISIC-DS and MoNu-DS datasets are utilized, as they provide both segmentation masks and classification annotations, which are necessary for a comprehensive evaluation. Either a semantic segmentation (ISIC-DS) or an instance segmentation (MoNu-DS) task is solved. The U-Net [Ron15] architecture with a ResNet-50 [He16] backbone is used.

Neighborhood Evaluation: The SelfSLBench framework evaluates representation spaces by examining the nearest and farthest neighbors of reference images in the ISIC-DS and MoNu-DS datasets. This approach is crucial for understanding how effectively samples from the same class cluster together, which is essential for evaluating the performance of Self-SL methods.

Figure 6.2a shows the *Nearest* and *Farthest* neighbors for the ISIC-DS considering three *Reference* images. For the first image, taken from the *Seborrheic Keratosis* class, the *Nearest* neighbor is within the same class for four approaches. Qualitative similarities are observed, particularly for DenseCL and MoCo, where a round

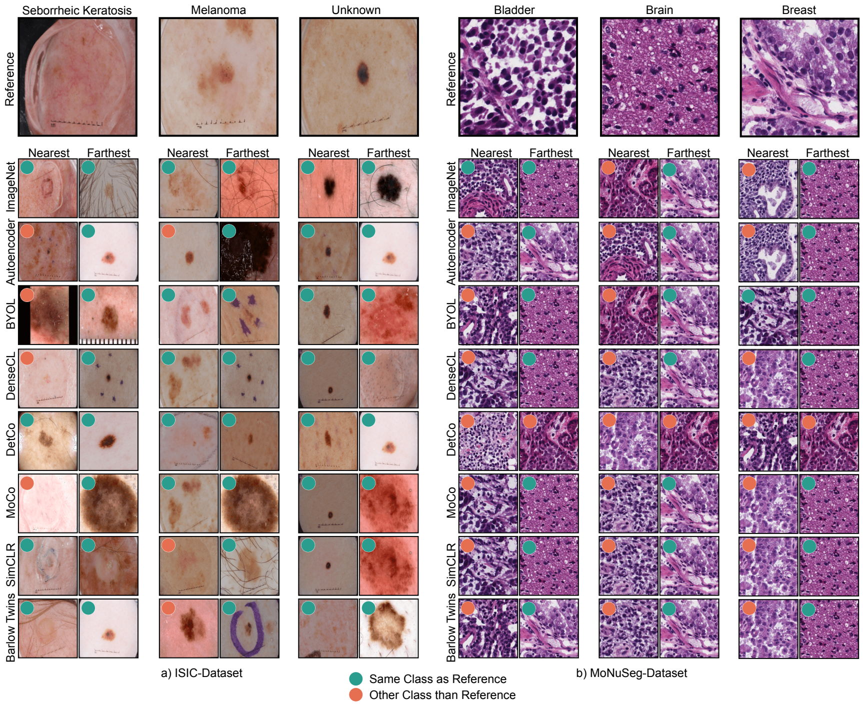


Figure 6.2: The nearest and farthest neighbors of three reference images for all considered CL methods, ImageNet weights, and the Autoencoder approach. The circle in the upper left corner is green if the class of the respective image is the same as the reference and red if it is not. The nearest and farthest neighbors are calculated as the Euclidean distance after a PCA with 10 output components. Adapted from [Ret23b].

border encircling the object of interest is evident in both the reference image and these methods. Qualitative similarities are also visible in the second sample taken from the *Melanoma* class. The *Nearest* neighbors for ImageNet, BYOL, DenseCL, and MoCo are not only within the same class but also exhibit similar visual characteristics (patchy, disseminated melanoma with indistinct borders). Although the *Nearest* neighbors for DetCo and SimCLR are not from the *Melanoma* class, visual similarities are still apparent. However, for the Autoencoder and Barlow Twins, the *Nearest* neighbors appear less similar. For the third sample (*Unknown* class), the *Nearest* neighbors for all methods are identified within the same class and appear visually similar, resembling small black melanomas with defined edges (except Barlow Twins). Across all reference images and methods, the *Farthest* neighbors are visually dissimilar. However, the methods do not always accurately cluster regarding the classification annotations, as sometimes the *Farthest* neighbors belong to the same class as the reference image. This inconsistency is likely attributed to substantial visual variations within individual classes.

The qualitative observations are supported by the Neighborhood Quality Criterion (ψ_{NQC}) metric, which quantifies the effectiveness of class-dependent clustering. Assuming a random mapping of samples into the representation space, the expected value is $\mathbb{E}_{\text{random}}[\psi_{\text{NQC}}] = 0.49$ (see Equation (6.1)). The Autoencoder, Barlow Twins, and DetCo do not surpass this value. All other methods are noticeably better than $\mathbb{E}_{\text{random}}$, which indicates that class-dependent clustering occurs during training of the Self-SL approaches, even though it is not flawless. DenseCL is striking with a value of 0.64. The full results are given in Table 6.1.

	$\psi_{\text{NQC}}\uparrow$	
Method	**ISIC-DS**	**MoNu-DS**
ImageNet	0.58	0.15
Autoencoder	0.48	0.07
BYOL [Gri20]	0.57	0.15
DenseCL [Wan21b]	**0.64**	**0.23**
DetCo [Xie21]	0.48	0.15
MoCo [He20]	0.56	0.15
SimCLR [Che20a]	0.56	0.15
Barlow Twins [Zbo21]	0.48	0.15

Table 6.1: Neighborhood Quality Criterion (ψ_{NQC}) for both datasets. $\uparrow$ means that large values are better. The best method is marked in bold.

Figure 6.2b shows the *Nearest* and *Farthest* neighbors for the MoNu-DS. The leftmost sample's *Nearest* neighbors are observed to be predominantly similar across most methods, characterized by intense, dark hues and contrasting red and white tones, with Autoencoder and DetCo standing out with slightly less similar results. For the middle sample, a less distinct pattern is observed. Similarities in the *Nearest* neighbors are identified for ImageNet, Autoencoder, and BYOL, while disparities are noted for other methods. The third reference image's neighborhood is similarly ambiguous, with four methods (DenseCL, MoCo, SimCLR, and BarlowTwins) sharing an identical *Nearest* neighbor. ImageNet and the Autoencoder exhibit a different, yet mutually consistent, *Nearest* neighbor. The correlation between class label clustering and representation quality is less noticeable for the MoNu-DS compared to ISIC-DS, which can be explained by its goal of instance segmentation rather than classification. While class labels are available in MoNu-DS, the dataset is not primarily designed for classification tasks.

The identified challenges above are reflected in the ψ_{NQC} calculation. A random representation mapping yields $\mathbb{E}_{\text{random}}[\psi_{\text{NQC}}] = 0.15$, a value that is not surpassed by most methods. The Autoencoder's performance even falls below this value. This is most likely because its pretraining parameters are more informative than Random but misaligned with the given class labels. This results in a representation space that is less effective in terms of class-based clustering compared to no pretraining. DenseCL stands out as the only method exceeding $\mathbb{E}_{\text{random}}$ with a value of 0.23 (see Table 6.1 for details).

The evaluation of the *Nearest* neighbors suggests that the methods construct representation spaces sharing visible qualitative characteristics. These observations are quantified by the novel ψ_{NQC} metric. Among the examined approaches, DenseCL emerges as the most promising, according to neighborhood observation, across both datasets. This is important because it highlights the practical implications of each method's performance in real-world scenarios. The developed evaluation methods offer comprehensive insights into the Self-SL approaches under consideration. They are particularly well-suited for evaluating methods in data-constrained environments, where the ability to cluster similar samples effectively is crucial. By quantifying neighborhood quality visually and quantitatively, these metrics offer insights into how the Self-SL methods organize their representation spaces. This indicates how well the methods generated generalizable knowledge from training.

The SelfSLBench framework demonstrates its utility by comprehensively assessing how different methods construct and organize their representation spaces. Quantifying neighborhood quality both visually and quantitatively provides insights into the strengths and weaknesses of each method in generating generalizable knowledge from training.

Hardware Requirements: Table 6.2 presents the relevant factors for assessing the hardware requirements and application overhead of the considered CL methods. ImageNet and Autoencoder contain no hyperparameters γ to be tuned ($\gamma = 0$). The classification results for ImageNet are accessible online, thereby reducing the implementation effort κ to a mere download and no additional trained parameters $\Delta\theta$. Removing skip connections in the U-Net is needed for the Autoencoder, increasing κ. Further, training of the expanding path is required additionally to the used backbone, resulting in an increase of parameters by $\Delta\theta = +11\text{M}$. In both cases, the required batch size b is approximately equivalent to SL.

Method	$\gamma\downarrow$	κ	$\Delta\theta\downarrow$	b	$\psi_{\text{RQC}}\downarrow$
ImageNet	**0**	**+++**	**+0M**	**+++**	-
Autoencoder	**0**	++	+11M	**+++**	**1.00**
BYOL [Gri20]	1	+	+46.6M	- -	4.23
DenseCL [Wan21b]	3	-	+41.4M	++	5.45
DetCo [Xie21]	2	- -	+946.5M	++	5.12
MoCo [He20]	2	-	+32.4M	++	4.93
SimCLR [Che20a]	1	+	+2.2M	- -	4.07
Barlow Twins [Zbo21]	1	+	+12.6M	- -	4.53

Table 6.2: Hardware and application requirements for the considered methods. The number of hyperparameters to be adjusted is denoted by γ. The implementation effort is represented by κ. The difference in parameter count relative to the ResNet-50 [He16] backbone is symbolized by $\Delta\theta$. $\mathbf{b}$ indicates the necessary batch size. Finally, ψ_{RQC} represents the relative duration until convergence compared to the *Autoencoder* averaged across both datasets.

Implementing SimCLR, Barlow Twins, and BYOL requires incorporating image pair generation, a contrastive loss function, and a projection head. One hyperparameter must be optimized for each of these methods, and a sufficiently large batch size b is essential to provide an adequate number of negative samples for approximating the similarity space. In terms of additional parameters, SimCLR has the fewest ($\Delta\theta$ = +2.2M), followed by Barlow Twins($\Delta\theta$ = +12.6M), and BYOL ($\Delta\theta$ = +46.6M).

Queue-based sampling for negative samples [He20] is employed by all MoCo-derived approaches (MoCo, DenseCL, and DetCo), which drastically reduces the batch size b while increasing the implementation effort κ. The tuning of two hyperparameters is required for MoCo and DetCo, whereas DenseCL necessitates three. Among all methods, DetCo has the highest parameter count due to its multiple projection heads ($\Delta\theta$ = +946.5M).

For the ψ_{RQC}, t* is set to Autoencoder, as it is the fastest method. The values are averaged over both datasets. The queue-based sampling methods MoCo, DenseCL, and DetCo need the longest training times. SimCLR($\psi_{\mathrm{RQC}} = 4.07$) is the fastest CL method, followed by Barlow Twins($\psi_{\mathrm{RQC}} = 4.53$) and BYOL ($\psi_{\mathrm{RQC}} = 4.23$).

For the running quality criterion ψ_{RQC}, the Autoencoder is the fastest converging method t*. The values presented are averaged across both datasets. The queue-based sampling methods MoCo, DenseCL, and DetCo exhibit the longest training durations. Among the CL methods, SimCLR demonstrates the highest efficiency ($\psi_{\mathrm{RQC}} = 4.07$), succeeded by BYOL ($\psi_{\mathrm{RQC}} = 4.23$) and Barlow Twins($\psi_{\mathrm{RQC}} = 4.53$).

The comparison of pretext methods reveals considerable differences in their hardware requirements. For instance, ImageNet and Autoencoder require minimal tuning and additional parameters, making implementing them straightforward. In contrast, methods like DetCo necessitate substantial computational resources due to their large number of parameters and complex architecture. This variation highlights the importance of considering hardware requirements when selecting a pretext method. The developed SelfSLBench framework is particularly adept at capturing these differences by assessing critical factors such as the number of hyperparameters to be tuned (γ), implementation effort (κ), and additional parameters ($\Delta\theta$). The SelfSLBench framework enables practitioners to select the most suitable methods based on their available resources by providing a comprehensive overview of these hardware-related factors. Moreover, SelfSLBench's structured approach ensures that the evaluation process is systematic and thorough, allowing for a fair comparison of pretext methods under various resource constraints. This capability is crucial for ensuring efficient deployment in real-world scenarios, where resource limitations can drastically impact the feasibility and effectiveness of Self-SL methods.

6.4.1.2 Downstream Application

Class Activation Maps: In Figure 6.3, three reference images are shown, together with their predicted segmentation masks, Class Activation Maps (CAMs), and corresponding quality metrics (DSC or AJI$^+$). The ISIC-DS was trained with 32 annotated samples, and the MoNu-DS with 10.

The SelfSLAutoTune framework provides a structured approach to assessing the performance of various pretext methods. For instance, with the ISIC-DS, acceptable performance is observed even with No Pretraining, as evidenced by the rough focus on the actual melanomas when observing the CAMs. The utilization of an Autoencoder as a pretext task (DSC = 60%) results in a diminished performance when compared to no-pretraining (DSC = 62%). This observation is further supported by the CAMs, which indicates that the Autoencoder pre-trained version of the U-Net focuses on implausible regions within the images. Pretraining with ImageNet yields good results (DSC = 70%), particularly considering that it is the leading implementation and training efficiency method. When evaluating the DSC score, BYOL, SimCLR, and Barlow Twins are found to be comparable, each achieving a score of 71%. This demonstrates the framework's ability to highlight subtle differences between methods.

In the case of the MoNu-DS, less straightforward results are observed, which is presumably attributed to its more challenging nature, as numerous individual instances require segmentation. Nevertheless, distinct variations among the different

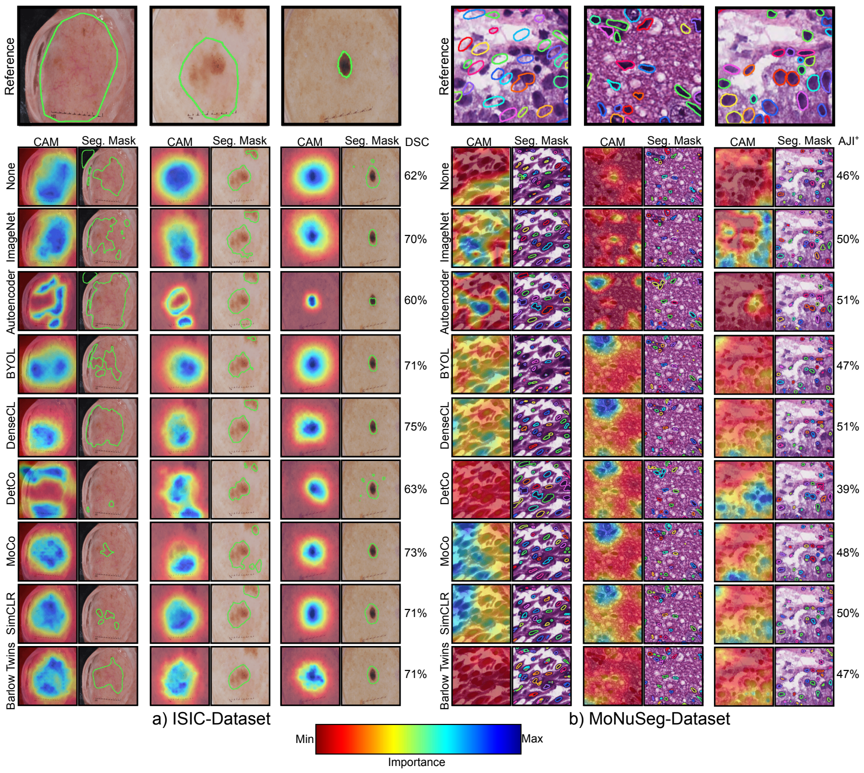


Figure 6.3: The Class Activation Maps (CAMs) for three reference images from the ISIC-DS and MoNu-DS. For each image, the ground-truth segmentation mask is overlaid. The CAM and predicted segmentation mask for every method are provided. The U-Net was trained using only 2% of labeled samples for each dataset, with the remaining 98% of unlabeled samples being excluded from the downstream task. The encoder of the U-Net was kept frozen during this process. The DSC and AJI$^+$ metrics are displayed and computed over the entire test split of the respective dataset. Adapted from [Ret23b].

methods are evident. DetCo exhibits the poorest performance (AJI$^+$ = 39%), even being surpassed by no pretraining (AJI$^+$ = 46%). ImageNet (AJI$^+$ = 50%) and the Autoencoder(AJI$^+$ = 51%) are determined to be reasonable options, as they outperform most of the Self-SL methods. DenseCL (AJI$^+$ = 51%) is comparable to the Autoencoder. The leftmost reference image shows that DenseCL concentrates on areas with high nuclei density, while the Autoencoder displays a less defined focus. The disparities between the various pretext methods are minimal, suggesting that limited information can be extracted from the MoNu-DS.

The ability of the SelfSLAutoTune framework to evaluate pretext methods under different conditions highlights the importance of task-specific optimizations. For instance, DenseCL yields the most fitting results for both datasets, as it is designed explicitly for dense predictions. This suggests that pretext methods optimized for the respective downstream task are beneficial and that general CL methods may not be optimal for complex tasks that focus on challenges beyond classification. SelfSLAutoTune thus offers valuable insights into how different methods perform under constrained conditions, facilitating the selection of the most appropriate approach for specific tasks.

Centered Kernel Alignments: The evaluation of CKAs enables a qualitative assessment of how different pretext methods perform in terms of focusing on relevant features. This is crucial because it highlights the ability of the SelfSLAutoTune framework to compare various methods under different conditions systematically.

In Figure 6.4, a comparison is presented regarding the CKA, contrasting various pretext methods and SL training on the entire datasets. For the ISIC-DS, correlations between CAMs and high CKA values are observed. It is noted that the most effective methods also exhibit representations most akin to SL. Given sufficient training data, this suggests that appropriate pretext tasks can learn parameters highly similar to those learned through fully supervised training. Additionally, these correlations indicate that the quantitative metric CKA may supplement or potentially replace the qualitative CAMs. ImageNet demonstrates superior representations in the upper layers (Conv1 and Conv2). While SimCLR does not achieve quite as strong representations in the upper layers, it exhibits less flattening of CKA values in the lower layers (Conv3, Conv4). DenseCL is observed to have the overall best and least diminishing CKA values. The mean CKA value across all layers further substantiates the previous observations. DenseCL demonstrates the highest mean value of 0.80, followed closely by SimCLR and ImageNet,both at 0.79, with a substantial gap to the next method BYOL, which

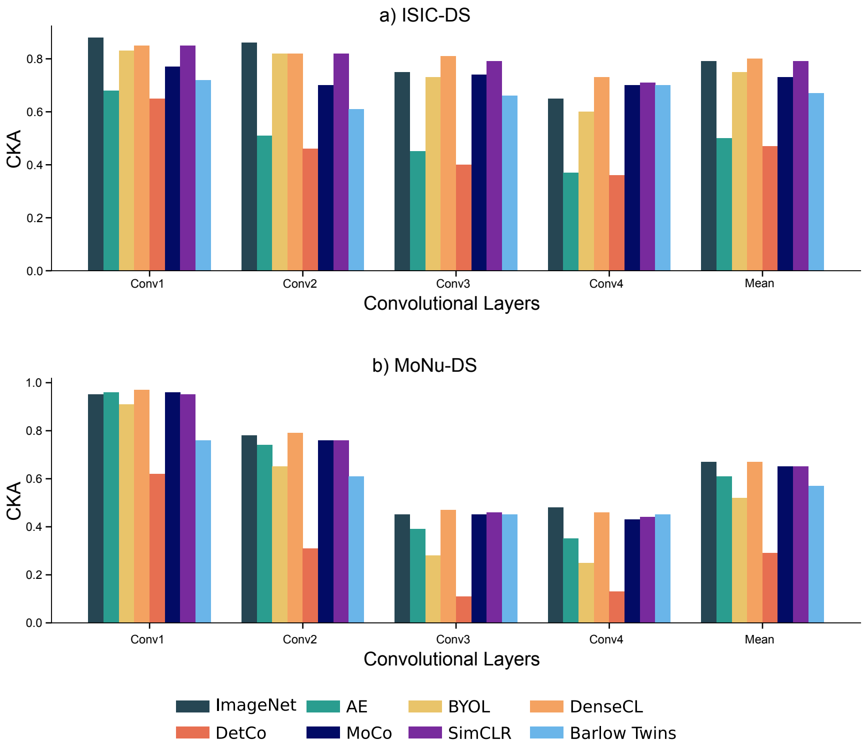


Figure 6.4: CKA similarity scores comparing the feature representations of various Self-SL methods to SL across the first four convolutional layers (*Conv1–Conv4*) of a ResNet-50 [He16] encoder. Higher CKA values indicate greater similarity to the SL baseline. For each method, the mean CKA value across all layers is also reported. This figure enables a direct comparison of how closely the Self-SL methods align with supervised representations at varying network depths. The tabulated values are provided in Table A.8.

has a mean value of 0.75. The findings support the effectiveness of transfer learning with ImageNet, as it provides generalized representations applicable to various challenges, although task-specific parameters must be relearned through fine-tuning. Conversely, Self-SL is found to learn useful features throughout the entire network, making it a better fit than mere transfer learning.

In the MoNu-DS, noticeably higher CKA values are observed in the initial layers compared to the deeper ones. This phenomenon is expected for ImageNet, given that the domain of histopathological data is considerably distinct from the content encompassed by the ImageNet dataset. When the Self-SL methods are examined, it is noted that the upper filters of the ResNet fit very well compared to SL, followed by a rapid deterioration. This observation suggests that insufficient information was provided during the training process to facilitate learning complex and specialized features in the network's lower layers. Despite this, it is found that DenseCL, with an average value of 0.67, provides the most optimal parameters across nearly the entire network, performing similarly to ImageNet with the same average value of 0.67.

This evaluation underscores the SelfSLBench's ability to assess the strengths and weaknesses of different methods under varying conditions. It provides insights into how they organize their representation spaces and focus on relevant features.

Integrated Quality Criterion: The quantitative assessment of the novel Integrated Quality Criterion (ψ_{IQC}) metric further substantiates the preceding observations. It summarizes the performance across the complete spectrum of label ratios into a single value within the SelfSLBench framework. Figure 6.5 shows the ψ_{IQC} for both observed datasets. Here, a distinction is made between *frozen* and *unfrozen* encoders to ascertain the practicality of fine-tuning the parameters provided by the pretext task. Consequently, it can be regarded as an estimation of a comprehensive perspective.

As in the previous observations, DenseCL is the predominant method, with ImageNet following closely. However, when the entire spectrum of available annotations is considered, the disparities in performance are much smaller.

An examination of the ISIC-DS reveals that ψ_{IQC} is notably higher for all Self-SL methods, except DetCo, when compared to the Random approach. In the case of *frozen* encoders, the most effective method, DenseCL, demonstrates an improvement in ψ_{IQC} of 8.5%. Each method's performance increases when the encoder is *unfrozen*. This observation suggests that the features acquired through Self-SL techniques are not inherently optimal for the downstream task and should, at minimum, be fine-tuned to achieve superior results. DenseCL maintains its superiority, showing a 4.4% enhancement compared to the Random approach.

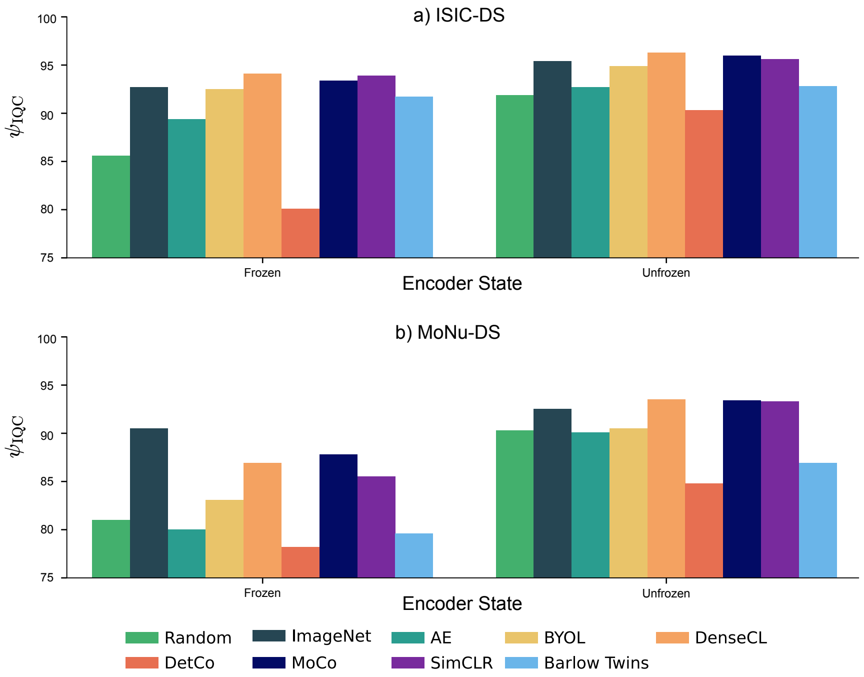


Figure 6.5: The ψ_{IQC} as a quantitative comparison of Self-SL methods for semantic segmentation performance on the ISIC-DS and MoNu-DS. Results are divided by frozen encoder (pretrained features fixed) and unfrozen encoder (features fine-tuned during downstream training). Segmentation quality is quantified using the DSC for ISIC-DS (skin lesion boundaries) and AJI^{+} for MoNu-DS (nuclear instance segmentation). Performance metrics are provided in percentages. The tabulated values are given in Table A.9.

Furthermore, all methods (excluding DetCo) exhibit at least a marginal improvement over the Random approach. This finding indicates that using a pretext task is justified, even in scenarios with varying annotation rates.

An examination of the MoNu-DS with *frozen* encoders reveals that most methods enhance downstream task performance compared to the Random approach. Exceptions are observed in the cases of Barlow Twins and DetCo, where performance worsens. The findings are consistent with the CKA similarities presented in Figure 6.4, where these two methods also exhibit the lowest values, drastically diverging from other approaches. Notably, while DetCo underperformed, Barlow Twins demonstrated efficacy in the ISIC-DS, potentially indicating that it is unsuitable for instance segmentation tasks. If the encoder is frozen, the best representations are provided by ImageNet, showcasing its capability to offer effective representation even if there are no apparent domain commonalities. This observation may be interpreted as either an indication of Self-SL's unsuitability for this particular downstream task or a reflection of insufficient dataset size. However, a very different pattern is observed when encoders are unfrozen. While ImageNet shows only marginal enhancement, substantial improvements are demonstrated by the Self-SL approaches. This suggests that, unlike ImageNet, Self-SL acquires useful representations throughout the entire network. It is hypothesized that considerable effort is required for their relearning when features deep within the network are unsuitable, as is likely the case with ImageNet-provided parameters.

The SelfSLBench framework's unique ability to evaluate the effectiveness of various Self-SL methods across different annotation rates highlights its exceptional value for real-world applications, where labeled data is often scarce. By delivering a comprehensive and detailed analysis of how various Self-SL methods perform under diverse conditions, SelfSLBench enables, for the first time, informed selection of the most suitable method for specific tasks. This substantially streamlines and enhances the deployment of Self-SL solutions in practical, real-world scenarios, marking a major advancement in the field of Self-SL.

6.4.2 Self-SL Optimization

For the evaluations focusing on the optimization of individual Self-SL methods using the SelfSLAutoTune framework, the SEM-DS and A-SEM-DS are used. The datasets consist of samples from materials science in which particles need to be identified. The SEM-DS has annotations that are provided as instance segmentations of the individual particles, and A-SEM-DS contains raw images of automated SEM scans. Details regarding the two datasets are provided in Section 4.3.

Comparative Performance Evaluation: As DenseCL has emerged as the most capable CL approach using the SelfSLBench framework, it is compared to ConvNeXtV2 and ImageNet fine-tuning to evaluate how the novel Masked-AE paradigm situates itself against the most promising of the established methods. The ConvNeXt [Liu22] encoder is employed for all models. DenseCL and ConvNeXtvV2 are pre-trained with the A-SEM-DS. As the downstream task, the instance segmentation challenge of the SEM-DS is used. The dataset is split by magnification into two distinct datasets, as it has been shown that doing this increases performance [Ret24c]. For this section, the uncertainty quantification of the SEM-DS is discarded. For training details of both the pretext and downstream tasks, see Appendix A.6.4.

Figure 6.6a shows SelfSLAutoTune's analysis for low magnification. The multi-

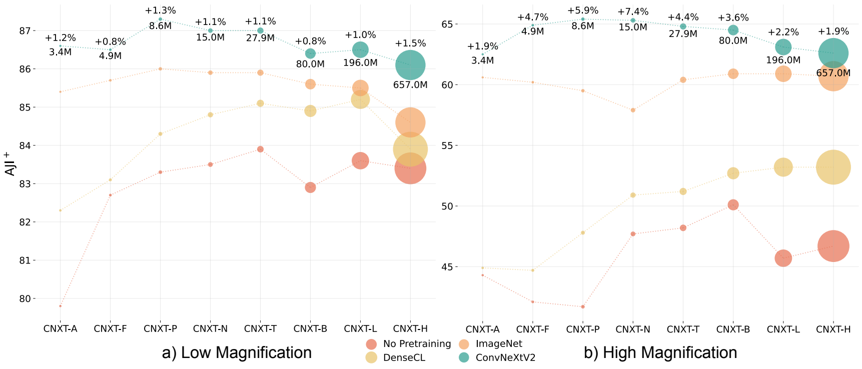


Figure 6.6: Comparison of No Pretraining, DenseCL, ImageNet, and ConvNeXtV2 with different backbone scales. Each circle represents one method's AJI$^+$ score with the respective backbone size. The size of the circles aligns with the number of parameters for the backbones. The number of parameters in millions and the difference between the best and second-best methods are provided at the circles of the ConvNeXtV2 model. Details about the model sizes are given in Appendix A.6.4, and the complete tabulated results are provided in Appendix A.6.5. Adapted from [Ret25b].

scale evaluation reveals ConvNeXtV2's consistent superiority across backbone sizes, particularly for smaller architectures. While performance gaps narrow with larger backbones, the evaluation identifies CNXT-P[3] as optimal (87% AJI$^+$) with diminishing returns beyond this scale. The Error Reduction Rate (ERR)[4] quantifies ConvNeXtV2's advantages: 24% error reduction vs No Pretraining, 19% over DenseCL, and 11% over ImageNet.

Figure 6.6b shows the high magnification analysis. The parametric evaluation reveals intensified performance differences, with ConvNeXtV2 achieving 7% higher AJI$^+$ than ImageNet for CNXT-N. The backbone scaling analysis reveals consistent peak performance at medium sizes (CNXT-P/N/T), with ERR improvements reaching 41% compared to No Pretraining. SelfSLAutoTune's multi-condition evaluation successfully captures modality-specific optimization requirements, showing ConvNeXtV2's Masked-AE approach is particularly effective for fuzzy high magnification features.

The results highlight the advantages of the Masked-AE paradigm implemented in ConvNeXtV2, showcasing considerable improvements compared to both traditional ImageNet pretraining and CL approaches, such as DenseCL. The consistency of ConvNeXtV2's superior performance across all backbone sizes underscores its effectiveness as an Self-SL method, while revealing a computational optimum at medium-sized backbones, indicating that larger models may not always yield better results. Generally, the SelfSLAutoTune framework enables practitioners to evaluate methods in terms of the number of trainable parameters, balancing computational effort against quality requirements.

The AJI$^+$ values for individual samples (Figure 6.7a) further demonstrate the clear superiority of ConvNeXtV2 over other methods while simultaneously validating SelfSLAutoTune's ability to capture method-specific performance characteristics. The per-sample analysis reveals ConvNeXtV2 consistently outperforms all other approaches across all samples, often with a substantial margin. As the other struggling methods indicate, this performance gap is particularly pronounced in challenging cases. It leads in five out of seven samples for high magnification cases, showcasing its versatility across different imaging conditions. However,

[3] The model identifiers of the ConvNeXt encoders are listed in Table A.10.
[4] The ERR is introduced in Section 3.4.1.

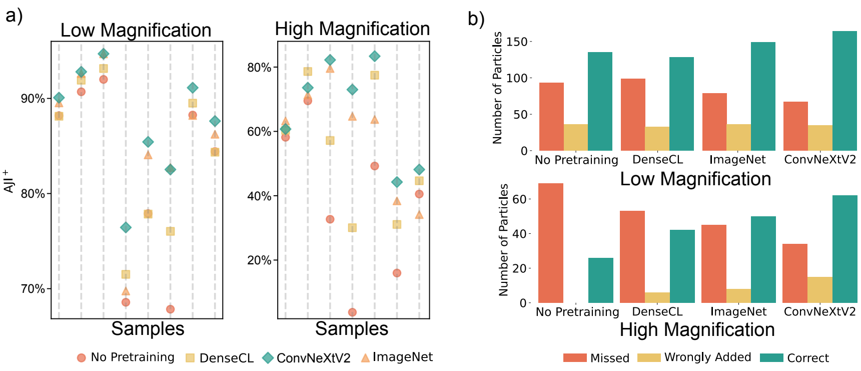


Figure 6.7: Details regarding the individual evaluated test samples. No Pretraining is training with random initialization of model weights, DenseCL / ConvNextV2 are the Self-SL methods pre-trained on the dataset of this work, and ImageNet is fine-tuning after training on the ImageNet classification challenge. a) The AJI^+ scores for all test set samples for the downstream task. b) The distribution of segmented particles for each method over all test set samples. A particle is correct if the IoU between the ground truth and an estimated particle is above 40%. It is marked as missed if there is no estimated particle with a IoU $\geq 40\%$ to a ground-truth particle. Wrongly added particles are those in which there is no ground-truth particle with a IoU $\geq 40\%$ to an estimated one. Adapted from [Ret25b].

the performance gap between ConvNeXtV2 and other methods is more consistent in low magnification samples. While ConvNeXtV2 generally leads in high magnification, there are instances where ImageNet and DenseCL come close or perform better. This indicates that the advantage of ConvNeXtV2 is to be more pronounced in lower resolutions, which are generally less fuzzy.

Figure 6.7b displays the distribution of correctly identified versus missed particles across all samples. ConvNeXtV2 excels in both magnifications, correctly identifying the highest number of particles. This advantage is particularly pronounced in high magnification scenarios. Moreover, ConvNeXtV2 misses the least number of particles in both magnification settings, indicating its robust detection capabilities. Interestingly, while the number of wrongly added particles is nearly identical for all methods in low magnification, ConvNeXtV2 shows a higher count in high magnification. This suggests that ConvNeXtV2 is slightly more sensitive than the other methods and might sometimes identify background noise as particles. For both magnifications, No Pretraining (random initialization) has many missed particles and few correctly identified particles. It is particularly striking at high magnification, with very few correctly identified particles and many missed particles. At low magnification, all methods identify more particles correctly than with high magnification, aligning well with the knowledge that high magnification SEM scans are more challenging, as seen by the overall reduction in correctly identified particles and an increase in missed particles.

Figure 6.8 shows visualizations of feature activations of the CNXT-P backbone for the different approaches. These visualizations provide critical insights into how

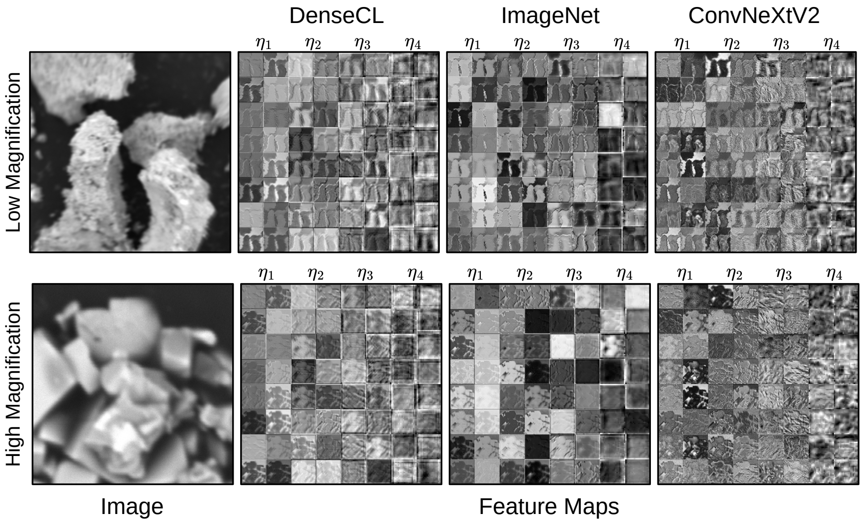


Figure 6.8: Visualization of feature activations from the CNXT-P backbone. For each of the four ConvNeXt backbone stages $\eta_i \in 1, \ldots 4$, 16 randomly selected feature activation maps are shown in two columns in small squares. The activation maps are drawn after the GRN layer, and the network weights are not modified after pre-training. Architectural details are given in [Liu22, Woo23]. Adapted from [Ret25b].

different pretraining paradigms shape feature representations at various network depths, particularly in layers prone to feature collapse [Woo23]. The feature maps are drawn from the dimension-expansion MLP layers after the GRN block within the ConvNeXt backbone. This network stage was chosen for visualization since this is where the feature collapse phenomenon was primarily observed [Woo23], which suggests that those features are particularly considerable. Randomly selected features are shown across all four stages η_1 to η_4.

For low magnification, DenseCL's activations exhibit repetitive, low-contrast patterns with limited spatial variation, particularly in early layers (η_1 and η_2). These patterns lack sharp edges and do not show a clear progression in complexity to later layers (η_3 and η_4), indicating limited hierarchical feature capture. ImageNet-pretrained activations reveal more defined structures, with noticeable edge detection in early layers and increasingly complex features in deeper layers, though some homogeneous regions suggest saturation. ConvNeXtV2 demonstrates the most refined patterns, characterized by sharp boundaries and diverse structures across all layers. Its early-layer activations (η_1) are comparable to ImageNet's but exhibit slightly higher definition, while its deeper-layer activations (η_4) maintain distinct and well-structured patterns.

For high magnification, these differences are amplified. DenseCL's activations remain diffuse with few discernible patterns. At the same time, ImageNet shows improved edge detection compared to DenseCL, though its precision diminishes at higher magnifications with increased saturated or dead features. ConvNeXtV2 consistently produces structured activations with clear particle boundaries and internal details across all stages. Notably, ConvNeXtV2 maintains a hierarchical progression of features from low-level edges in η_1 to complex structures in η_4, a property less evident in other methods, particularly at high magnification where DenseCL and ImageNet degrade drastically. While DenseCL and ImageNet often produce block-like structures in later stages, ConvNeXtV2 exhibits textured and detailed patterns indicative of richer representations. These visualizations align with numerical results, illustrating how ConvNeXtV2's Masked-AE-based approach fosters task-relevant feature representations while addressing challenges like feature collapse and scale adaptation. Furthermore, observing feature maps, especially to identify feature collapse, is an essential practice in general. This example demonstrates the importance of examining the features in NN and how seemingly similar approaches might differ if observed in detail.

Figure 6.9 presents one example from each magnification used for testing, along with ground-truth segmentations, estimations, and CAMs for all methods. This

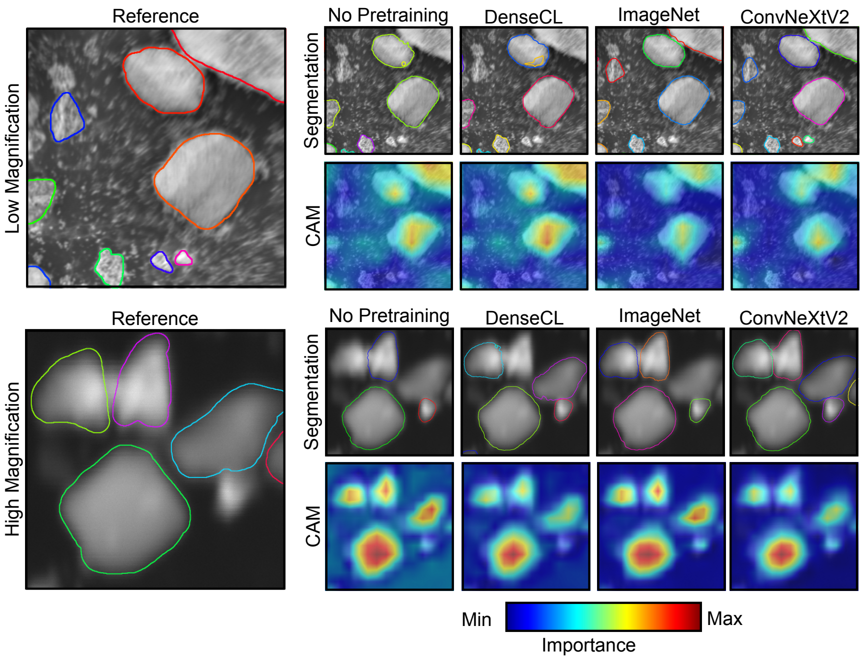


Figure 6.9: One sample for both magnifications with ground-truth labels (Reference). The colored curves are the domain experts' segmentations (ground truth) or the respective methods. Each curve is color-coded to indicate the individual instances. Further, the CAMs [Ram20] display which parts of the image the methods focus on the most. Adapted from [Ret25b].

evaluation leverages CAMs to analyze the focus regions of different approaches and their alignment with segmentation quality. Predicted segmentations vary noticeably across methods. No Pretraining shows poor segmentation quality, with particles often missing or inaccurately placed, especially at high magnifications. DenseCL improves over No Pretraining but struggles with precise segmentations in clustered scenarios. ImageNet achieves higher accuracy than DenseCL, capturing more precise boundaries and smaller, clustered particles. ConvNeXtV2 produces the most accurate segmentations overall, closely matching ground truth in both magnifications and outperforming ImageNet in challenging cases.

The CAMs provide valuable insights into the focus regions of each method to quantify differences in feature utilization. No Pretraining's CAM displays scattered focus poorly aligned with object boundaries. DenseCL focuses more on relevant regions, particularly at high magnification, but lacks sharpness and precision. ImageNet improves boundary focus and object center alignment compared to DenseCL. ConvNeXtV2 exhibits similar CAMs to ImageNet but places less emphasis on background noise while maintaining a strong focus on particles. This evaluation demonstrates how the proposed SelfSLAutoTune framework effectively uses CAMs to compare methods and identify subtle differences in their ability to focus on relevant features. These findings emphasize the importance of incorporating such evaluations when optimizing Self-SL methods for improved segmentation performance.

Dataset Size Analysis: Determining the data volumes needed for Self-SL is a critical challenge in data-constrained environments common to real-world applications. The SelfSLAutoTune framework enables systematic analysis of data efficiency through a controlled ablation study, shown by evaluating the impact of pretraining dataset size on ConvNeXtV2's performance.

Figure 6.10 illustrates the relationship between the number of samples used for Self-SL pretraining of ConvNeXtV2 and the resulting AJI$^+$ scores for both low and high magnifications. For low magnification (Figure 6.10a), the AJI$^+$ score improves steadily as the dataset size increases. Notably, ConvNeXtV2 surpasses ImageNet fine-tuning performance with only 1.000 samples (4% of the total dataset), highlighting its efficiency in learning from limited data. The performance continues to improve steadily and noticeably, with a considerable jump between the full dataset and 16.000 samples. This suggests that further increasing the dataset size beyond the current 25.000 samples could potentially yield even better results for low magnification tasks. This systematic evaluation identifies the potential for further gains beyond current dataset limits.

The impact of increased pretraining data is even more pronounced in the case of high magnification (Figure 6.10b). The evaluation reveals an initial performance

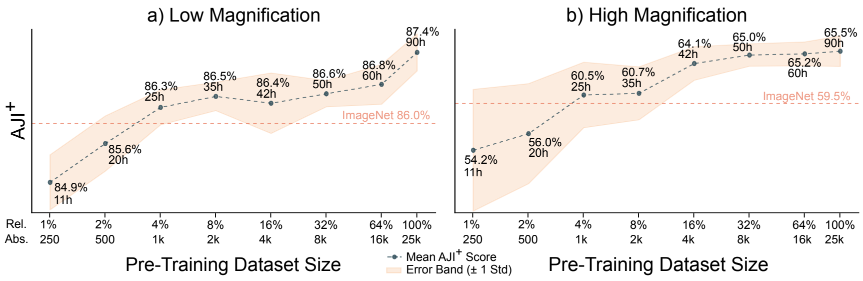


Figure 6.10: The relationship between AJI$^+$ and pretraining dataset size is illustrated. The log$_2$-scaled horizontal axis indicates the number of samples available for Self-SL training of ConvNeXtV2 CNXT-P (Pico) backbone, and the vertical axis displays the corresponding AJI$^+$ scores (in percentages) obtained after downstream training. Error bands representing the standard deviations are plotted along the scores in red, and the ImageNet fine-tuning AJI$^+$ score is included as a baseline reference. Training durations (in hours) for each dataset size are provided. Results represent 10 independent experimental repetitions. Complete tabulated data are provided in Table A.12. Adapted from [Ret25b].

plateau followed by steep quality increases, identifying 8.000 samples as the optimal cost-benefit value. Interestingly, at least 1.000 samples are needed to exceed the ImageNet fine-tuning baseline, which is the same data amount as the low magnification scenario. However, the performance gains for high magnification tasks appear to saturate as the dataset size approaches 100%. The increase AJI$^+$ score from 64% to 100% of the samples is relatively small, suggesting that further expanding the dataset may not yield substantial improvements. This evaluation highlights that the effectiveness of Self-SL pretraining varies depending on the data required for different tasks. In general, dataset-size-focused evaluations are essential for optimizing the application of Self-SL, as datasets differ drastically in the volume of data they need and the rate at which their performance saturates.

6.5 Discussion

The novel SelfSLBench and SelfSLAutoTune frameworks offer a systematic approach to evaluating Self-SL methods, facilitating both cross-method comparisons and detailed analysis.

SelfSLBench enables structured comparisons of methods based on metrics such as representation quality, runtime efficiency, and performance across annotation rates. DenseCL stands out for learning meaningful representations in challenging domains, excelling where annotated data is limited and spatial structure is complex, as in SEM particle segmentation. Queue-based methods, such as DenseCL and MoCo, enable training with smaller batch sizes, making them suitable for limited hardware resources, albeit at the cost of longer training times compared to batch-dependent methods. In contrast, SimCLR and Barlow Twins converge faster but require much larger batch sizes, which can be prohibitive with limited resources. This comparison highlights a tradeoff: queue-based methods are advantageous with restricted batch sizes, whereas batch-dependent methods are preferable where large batches are feasible.

The SelfSLAutoTune focuses on detailed evaluations of individual methods, emphasizing scalability and data efficiency. ConvNeXtV2 demonstrates superior performance across multiple backbone sizes, particularly at medium scales for low magnification SEM scans. This evaluation shows that larger models do not always yield better results, as excessive parameters can hinder performance. ConvNeXtV2 also outperforms DenseCL and ImageNet pretraining in both low and high magnification settings, reducing segmentation errors noticeably. Moreover, ConvNeXtV2 surpasses ImageNet fine-tuning with only a fraction of available data, demonstrating its efficiency in data-scarce scenarios.

These findings underscore the utility of the proposed frameworks for systematically evaluating Self-SL methods under real-world conditions. SelfSLBench facilitates cross-method benchmarking by providing insights into how different approaches perform across diverse tasks and resource constraints, while SelfSLAutoTune offers granular evaluations to guide practitioners in selecting appropriate model scales and dataset sizes. This chapter illustrates how Self-SL methods can reduce dependence on annotated data while maintaining high performance. The introduction of ConvNeXtV2 further expands the scope of Self-SL by offering an alternative paradigm that excels in dense prediction tasks compared to traditional CL approaches, providing valuable guidance for practitioners in resource-constrained environments.

7 Selection and Refinement of Neural Networks

Parts of this chapter are based on:

- L. Rettenberger, N. J. Szymanski, Y. Zeng, J. Schützke, S. Wang, G. Ceder, and M. Reischl. Uncertainty-aware particle segmentation for electron microscopy at varied length scales. *npj Computational Materials*, volume 10, page 124, Nature Publishing Group UK London, 2024. DOI: 10.1038/s41524-024-01302-w.
- L. Rettenberger, F. R. Münke, R. Bruch, and M. Reischl. Mask R-CNN outperforms U-Net in instance segmentation for overlapping cells. In *Current Directions in Biomedical Engineering*, volume 9, pages 335–338, Duisburg, Germany, 2023. De Gruyter. DOI: 10.1515/cdbme-2023-1084.

7.1 Overview

Despite the widespread adoption and numerous successes of DL across virtually all domains, its full potential often remains untapped due to insufficient refinement of ML algorithms and overlooked challenges inherent to real-world tasks. For example, an overarching issue is the lack of structured evaluation approaches that go beyond simple performance metrics. This deficiency is particularly noticeable in challenging scenarios, such as those involving overlapping objects. Another pressing issue is modeling annotation ambiguities in SL, which remain unaddressed at fine-grained levels, such as object-based uncertainty estimation. Consequently, existing DL solutions often lack the nuanced capabilities necessary for satisfactory application in scenarios involving challenges faced in real-world applications. Segmentation tasks, particularly instance segmentation, are affected by the shortcomings of current solutions due to inherent complexities such as object density, annotation quality, and overlap ambiguities, which current methods fail to address systematically.

To overcome these limitations, this chapter introduces novel overlap evaluation metrics to quantify dataset characteristics and an uncertainty-aware extension of Mask R-CNN that predicts instance-level confidence scores. The overlap metrics enable systematic evaluation, comparison, and selection of NN architectures based on dataset complexity. At the same time, the uncertainty head addresses annotation ambiguities by providing confidence estimates for individual object instances. Together, these contributions fill critical gaps in DL workflows by offering tools to account for dataset complexity, improve model selection, and enhance segmentation reliability through extended NN modeling capabilities.

7.2 Related Works

Architecture Selection: One of the most popular DL architectures for segmentation tasks is U-Net [Ron15], known for its simplicity and efficiency. Initially developed for biomedical applications, it has become the general go-to approach for semantic segmentation tasks. For example, it has wide applications in autonomous driving for road lane markings detection [Tra19], and scene segmentation [Sia18], and within materials science for vacancy and dopant detection in Transmission Electron Microscopys (TEMs) [Yan21], as well as particle detection in SEMs [Hor20, Gho21]. Moreover, the U-Net architecture is most widespread in its original domain of biomedicine, where it is used extensively for applications like the evaluation of the cardiac system medaka fish [Sch22d], retinal vessels [Xia18], skin lesions [Ibt20], and various other imaging applications [Xia18, Si23, Hou23]. Further, many extensions to the initial architecture exist, including adding residual mechanisms [Zhu19], using dilated convolutions [Tur19], introducing attention-based mechanisms [Zho18], and techniques for three-dimensional image segmentation [Bru22]. Despite its success and widespread adoption, U-Net encounters challenges in certain areas, such as learning instance segmentations. These challenges require considerable modifications and additional processing steps, as U-Net was not originally designed to address them. Several studies tackle these limitations in distinguishing individual objects by introducing additional processing steps [Zho18, Sch20, Wu22, Kir22, Wu22], modifying the architecture [Mah22], or combining U-Net with other NNs [Bai23]. This makes U-Net a popular choice for instance segmentation [Rah21, Zha22b].

An alternative approach to U-Net tailored for instance segmentation is Mask R-CNN [He17], which combines object detection and instance segmentation to produce precise masks around each object in an image. It has shown great potential, for instance, in segmentation tasks. For example, when segmenting cells [Fuj20], particles [Coh21, Pri21, Boy23], nanowires [Lin22], and cavities [Jac23, Che23] in electron microscopy. Furthermore, some early studies suggest that Mask R-CNN outperforms U-Net [Jac22, Rag23, Aga23]. However, only a limited number of studies directly compare the performance of Mask R-CNN and U-Net in semantic segmentation [Zha18, Alf19], and none that focus on instance segmentation.

Uncertainty Awareness: Quantifying the uncertainty of DL models is a highly relevant yet challenging endeavor [Jia18]. Many domains contain uncertainties, such as medical applications [Est17], autonomous driving [Der21], and microscopy imaging [Gho21]. Hence, many novel approaches exist that quantify uncertainty in DL. Here, most works focus on classification tasks and evaluate images as a whole to estimate uncertainties [Ova19, Abd21]. However, computing confidence across the entire image is not a practical approach in instance segmentation, as different instances may contain varying uncertainties. One approach estimates the epistemic uncertainty for bounding boxes and classifications to detect corner cases [Hei21]. Others cluster predictions to describe the uncertainty associated with each object in instance segmentations [Hei23, Smi23]. Additionally, some works focus on estimating uncertainty within objects to identify uncertain regions, such as blurred edges [Sid24].

Current uncertainty estimation approaches focus on either identifying out-of-distribution samples or distinguishing objects from the background. As a result, there is a significant emphasis on detection rather than finer differentiation between objects, such as assessing the complexity of an object to recognize.

7.3 Methods

The primary objective of this chapter is to provide tools for evaluating and selecting the most suitable approaches in segmentation tasks, thereby optimizing performance in real-world applications and enhancing existing approaches. Particular focus is placed on especially challenging instance segmentation tasks involving occluded, uncertain, and overall complex objects. To address the shortcomings of current methods that lack structured comparisons, a systematic approach for comparing DL models based on dataset difficulty is introduced. The presented framework is called the Architecture Selection Framework (ArchSelect) and uses multiple novel metrics to evaluate the suitability of architectures based on the complexity of datasets by introducing object-overlap measures. It provides decision support to aid practitioners in determining when to use specific methods, facilitating informed architecture selection instead of relying on trial and error. ArchSelect is exemplified by comparing U-Net and Mask R-CNN, two prevalent architectures within DL.

Following the strategies for improved architecture selection, a method is presented to enhance CNNs with uncertainty awareness, modeling ambiguous annotations, and eliminating conflicting signals during training to achieve more robust segmentations. The uncertainty-awareness concept is demonstrated by enhancing the popular Mask R-CNN [He17] architecture with uncertainty estimation capabilities for individual image instances, resulting in the extended framework called Uncertainty-Mask R-CNN (U-Mask R-CNN). This approach shifts the focus from mere detection to assessing object clarity, offering instance-level uncertainty measures that surpass existing methods, which are limited to whole-image or bounding-box confidences. These measures enhance the modeling capabilities of CNNs and provide actionable insights for identifying regions requiring expert review, ultimately optimizing instance segmentation pipelines for real-world scenarios involving overlap, occlusion, and annotation ambiguities.

7.3.1 Architecture Selection

The ArchSelect framework, a systematic approach for evaluating dataset complexity, is proposed to address the challenge of selecting appropriate NNs for complex instance segmentation tasks. ArchSelect aims to provide a nuanced understanding of how different architectures perform across varying levels of segmentation difficulty. By splitting a considered dataset into subsets based on complexity[1], the NNs can be trained on different levels of challenge, and performance metrics can be applied more granularly than on the whole dataset. This approach thoroughly examines how performance metrics evolve with increasing complexity, offering deeper insights than aggregating the score across all samples. In this way, practitioners are enabled to assess and compare DL architectures in a structured manner based on the specific challenges present in their datasets, providing an informed and efficient approach to architecture selection.

Consider a general dataset $\mathcal{D}$ comprising m samples to facilitate this in-depth analysis. This dataset is split based on complexity into n sub-datasets $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \dots \mathcal{S}_n\}$. Each sub-dataset in $\mathcal{S}$ contains the same number of samples to enable a fair comparison. This subdivision is utilized for NN training, as illustrated in Figure 7.1, and allows for a structured comparison of different methods, revealing performance patterns as complexities change within $\mathcal{D}$. For the considered instance segmentation dataset $\mathcal{D}$, assume that the ground truth instance

[1] Complexity within a dataset may refer to any identified difficulties. A common challenge in instance segmentation is the overlap of instances, which is regarded as the challenge factor to quantify in this work.

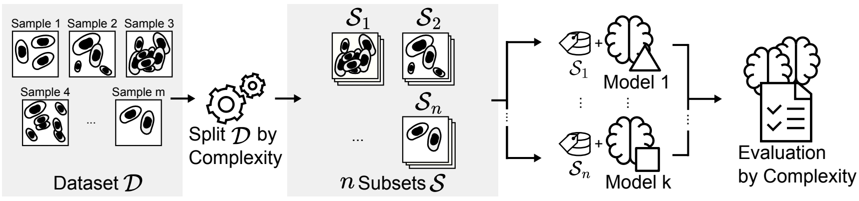


Figure 7.1: Overview of the ArchSelect framework. A dataset $\mathcal{D}$ containing m samples is split by *Complexity* into n equal-sized *Subsets* $\mathcal{S}$. Each *Subsets* is used by k considered DL models for optimization. After that, each model-subset combination is evaluated to obtain insights about performance considering the *Complexity*. Adapted from [Ret23a].

segmentation masks for each sample, denoted as $\mathcal{D}_i,\ i < m$, are represented as arrays of binary masks, each containing c instances and dimensions of $h \times w$ (height and width). Thus, the entire dataset is structured as $\mathrm{shape}(\mathcal{D}) = [m, c, h, w]$.

A novel measure called the Class Frequency Metric (ψ_{CFM}) is introduced to quantify instance overlap in samples. This metric explains how many different classes are overlaid at each pixel coordinate (u, v) in an image. The calculation of ψ_{CFM} for each pixel involves summing the binary masks of all instances at that pixel location:

$$\psi_{\text{CFM}}(\mathcal{D}, i, u, v) = \sum_{k=1}^{c} \mathcal{D}_{i,k,u,v}. \tag{7.1}$$

Here, $\mathcal{D}_{i,k,u,v}$ represents the binary mask value for the k-th instance at pixel (u, v) in the i-th image. This value is 1 if the instance k occupies the pixel (u, v) and 0 otherwise. By summing these values across all instances c for the sample i in dataset $\mathcal{D}$, the ψ_{CFM} metric counts the number of instances that overlap at the defined pixel.

To create the n equal-sized sub-datasets $\mathcal{S}$, the General Overlap Criterion (ψ_{GOC}) is introduced, which employs ψ_{CFM} to calculate how much general overlap is present in the images without specifying how many instances are overlaid over each other. The goal is to sort $\mathcal{D}$ based on the overlapping area and divide it into n equal-sized bins, each containing $\lceil |\mathcal{D}|/n \rceil$ samples. The percentage of multi-class pixels (overlaps) relative to the total masked area for a segmentation mask $\mathcal{D}_i$ is computed as

$$\psi_{\text{GOC}}(\mathcal{D}, i) = \frac{\sum_{u=1}^{h} \sum_{v=1}^{w} [\psi_{\text{CFM}}(\mathcal{D}, i, u, v) > 1]}{\sum_{u=1}^{h} \sum_{v=1}^{w} [\psi_{\text{CFM}}(\mathcal{D}, i, u, v) > 0]}, \tag{7.2}$$

where $[\cdot]$ is the Iverson bracket [Ive62] (1 if true, 0 otherwise). The numerator counts overlapping multi-class pixels, while the denominator calculates the total masked area. Using the ψ_{GOC} measure, $\mathcal{S}$ can be generated from $\mathcal{D}$, focusing on the general overlap area. The dataset samples are sorted by their ψ_{GOC} value for splitting by complexity and divided into equal-sized sub-datasets.

To further evaluate the overlap in more detail, the Overlap Frequency Criterion (ψ_{OFC}) is established, quantifying the overlap frequency, with values ranging from 0.0 (no overlap) to 1.0 (complete overlap). In contrast to the ψ_{GOC} metric, which is used to split the dataset based on the number of pixels that contain overlap, the ψ_{OFC} measure determines the intensity of overlap. To calculate the ψ_{OFC} metric, the maximum and minimum possible overlap frequencies must be defined to obtain the boundaries of values. The theoretical maximum overlap ${\psi_{\text{CFM}}}^{\uparrow}$ represents a scenario where all instances overlap at every position. Conversely, the theoretical minimum overlap ${\psi_{\text{CFM}}}^{\downarrow}$ defines the situation with no overlap. For a sample $\mathcal{D}_i$, the values of ${\psi_{\text{CFM}}}^{\uparrow}$ and ${\psi_{\text{CFM}}}^{\downarrow}$, are calculated as:

$$ {\psi_{\text{CFM}}}^{\uparrow}(\mathcal{D}, i) = \sum_{u=1}^{h} \sum_{v=1}^{w} [\psi_{\text{CFM}}(\mathcal{D}, i, u, v) > 0] * C, \tag{7.3}$$

$$ {\psi_{\text{CFM}}}^{\downarrow}(\mathcal{D}, i) = \sum_{u=1}^{h} \sum_{v=1}^{w} [\psi_{\text{CFM}}(\mathcal{D}, i, u, v) > 0]. \tag{7.4}$$

The theoretical maximum overlap ${\psi_{\text{CFM}}}^{\uparrow}$ assumes complete overlap of all C instances at every occupied pixel. ${\psi_{\text{CFM}}}^{\downarrow}$ represents no overlap, meaning each pixel belongs to exactly one instance.

With the ${\psi_{\text{CFM}}}^{\uparrow}$ and ${\psi_{\text{CFM}}}^{\downarrow}$ metrics established, the novel ψ_{OFC} metric can be introduced, with $\psi_{\text{OFC}} = 0$ indicating minimal overlap between instances, and

$\psi_{\text{OFC}} = 1$ suggesting maximum overlap. Specifically, the value of the ψ_{OFC} measure is calculated as:

$$\psi_{\text{OFC}}(\mathcal{D}, i) = \frac{\sum_{u=1}^{h} \sum_{v=1}^{w} \psi_{\text{CFM}}(\mathcal{D}, i, u, v) - \psi_{\text{CFM}}{}^{\downarrow}(\mathcal{D}, i)}{\psi_{\text{CFM}}{}^{\uparrow}(\mathcal{D}, i) - \psi_{\text{CFM}}{}^{\downarrow}(\mathcal{D}, i)}, \tag{7.5}$$

where the numerator calculates the actual overlap in the image by summing the class frequencies and subtracting the minimum possible overlap, and the denominator represents the range of possible overlap from minimum to maximum. This measure enables researchers and practitioners to quantify the complexity of segmentation tasks in a given dataset, compare the difficulty of different samples or datasets, and assess the performance of segmentation algorithms across varying levels of instance overlap.

To showcase the novel metrics for selecting the most fitting approach given a challenging segmentation dataset, two NN architectures are considered: U-Net and Mask R-CNN, as they have shown to be the most prevalent and promising within such tasks (see Section 7.2). As the initial U-Net architecture is not designed to address instance segmentation challenges, a popular extension is used to enable it to distinguish between individual objects[2]. The assessment evaluates segmentation performance, utilizing well-established metrics and visualizations, including the AJI^{+}. The evaluation delves deep into the nuances of these methods, using the novel methods introduced, particularly in scenarios involving challenging instance segmentations and object occlusions.

7.3.2 Uncertainty Awareness

This section introduces methods for enhancing neural networks with uncertainty awareness to model ambiguous annotations, with a focus on extending the Mask R-CNN framework to obtain the novel U-Mask R-CNN. This architecture is chosen for its promising results in instance segmentation and flexible framework. With a region proposal network and a separate mask generation branch, the Mask

[2] See Section 3.3.3 for details about the U-Net architecture and this extension.

R-CNN structure provides an ideal foundation for implementing uncertainty estimation. U-Net is not considered due to its architectural limitations in instance segmentation. The following sections detail the extension of Mask R-CNN with an uncertainty output (head), enabling confidence estimates for individual object instances. This enhancement provides instance-level uncertainty measures, surpassing existing methods limited to whole-image or bounding-box confidences.

Mask R-CNN Extension: The uncertainty estimation of the U-Mask R-CNN framework is achieved by adding a new output to the Mask R-CNN that predicts confidence levels for each detected instance. Unlike the objectness score, which is used as a preliminary filter to prioritize regions likely containing objects over background noise, this uncertainty quantification is conceptualized as an independent component. Its functionality is not integrated into the decision-making process or subsequent network operations but is intended to estimate the confidence of instances already identified as objects of interest. This separation enables a precise assessment of prediction reliability at the instance level, while preserving the integrity of the detection pipeline. The uncertainty output/head is implemented using a single-layer fully connected network, ensuring architectural simplicity. Decoupling uncertainty estimation from object selection enhances the model with nuanced confidence evaluations without compromising detection efficiency. This extension is visualized in Figure 7.2. An input $\boldsymbol{x}$ is fed through the network. First,

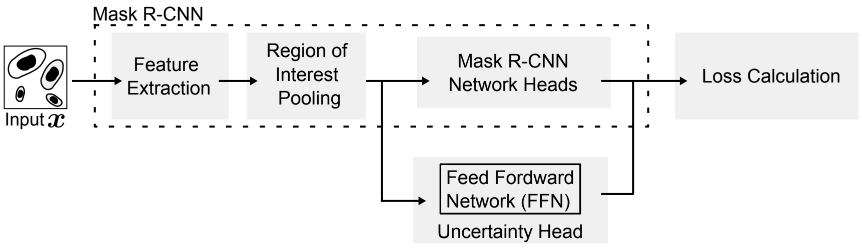


Figure 7.2: Visualization of the proposed extension to the Mask R-CNN architecture to obtain the novel U-Mask R-CNN. The input image $\boldsymbol{x}$ is processed by the default Mask R-CNN components. After Region of interest pooling, the aligned ROI are fed into the new FFN intended to estimate the uncertainty, and the output is added to the Mask R-CNN outputs.

the features are extracted, and region of interest pooling is applied to obtain the aligned ROIs[3]. After that, the aligned ROIs are processed by the default Mask R-CNN network heads and the novel uncertainty estimation. The uncertainty estimation may be implemented using any NN architecture, but it is realized through a single layer FFN in this work. After that, the outputs are combined for loss calculation. This novel U-Mask R-CNN architecture specifically quantifies the model's confidence in the precise boundaries and shape of each segmented object. This approach provides a nuanced measure of segmentation quality, particularly valuable in areas where factors like image blur, particle agglomeration, and complex morphologies can make accurate boundary delineation challenging.

The instance segmentation masks of a dataset provided by human annotators must be extended to enable the U-Mask R-CNN to learn the proposed uncertainty estimation. Hence, during labeling, each segmented object is given one of two distinct categories: uncertain objects ψ and certain objects ω. Having two categories for the human experts to set creates minimal additional annotation overhead while still obtaining information about certainty. However, estimating uncertainty may be difficult and prone to errors, as it exists on a spectrum rather than discrete values. Hence, the labels are transformed into continuous confidence scores. After this, Gaussian noise is added to reflect potential inaccuracies in the ground truth data. The transformation process is illustrated in Figure 7.3. The bounding boxes and segmentation masks, which classify objects as certain or uncertain, are employed to identify these categories. For certain objects, a value drawn from a distribution with a mean near one is utilized; in contrast, uncertain objects are sampled from a distribution with a mean near zero. The variability of this noise is controlled by the standard deviation σ. Integrating this noise enables a nuanced representation of object certainty based on the annotators' simple labels. It is intended that objects classified as uncertain should be associated with label values approximating 0, thus indicating a low level of confidence in their detection. Conversely, certain objects are expected to yield values close to 1, signifying high confidence in their

[3] For details regarding the Mask R-CNN components see Section 3.3.3.

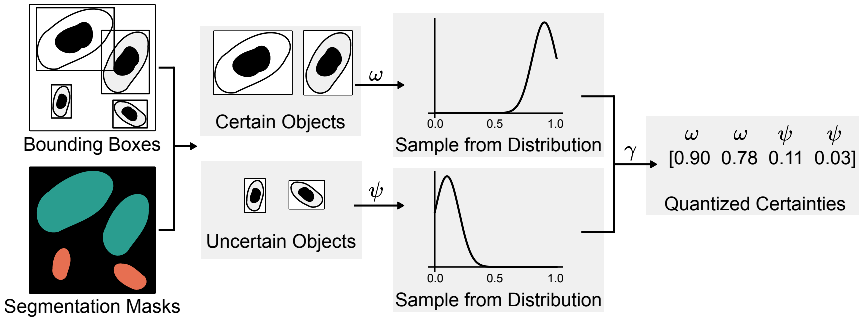


Figure 7.3: The transformation process from instance segmentation masks annotated with certain objects ω and uncertain objects ψ to quantized certainties γ. The bounding boxes and segmentation masks are used to identify the annotated objects, which are then transformed into continuous values by sampling from two distributions.

identification. The new labels are then used to generate a confidence value, denoted as γ. This value is constrained to ensure it falls within the range of 0 to 1. A normalization procedure is implemented wherein any γ values computed to be less than zero are automatically adjusted upward to 0, while those exceeding one are capped at this upper limit. This restrictive measure guarantees that the confidence value γ is consistently bounded within the closed interval [0, 1].

Label Uncertainty Loss: The Mask R-CNN loss[4] is enhanced to optimize the output of the U-Mask R-CNN using the quantized uncertainties provided by human annotators. The novel extension, denoted as $\mathcal{L}_c$, is designed to model uncertainty for individual objects through regression. An overview of the proposed uncertainty loss within the novel U-Mask R-CNN framework is shown in Figure 7.4. The quantized uncertainty values from ground-truth annotations form a bimodal distribution, with peaks near 0 (uncertain objects) and 1 (certain objects). Traditional loss functions such as Mean Absolute Error, Huber loss, and Euclidean loss are tailored for unimodal distributions and struggle with this complexity, requiring a specialized modeling approach. The multimodal regression problem is addressed by discretizing confidence scores into k bins [Mas16],

[4] The initial Mask R-CNN objective function is detailed in Section 3.3.3.

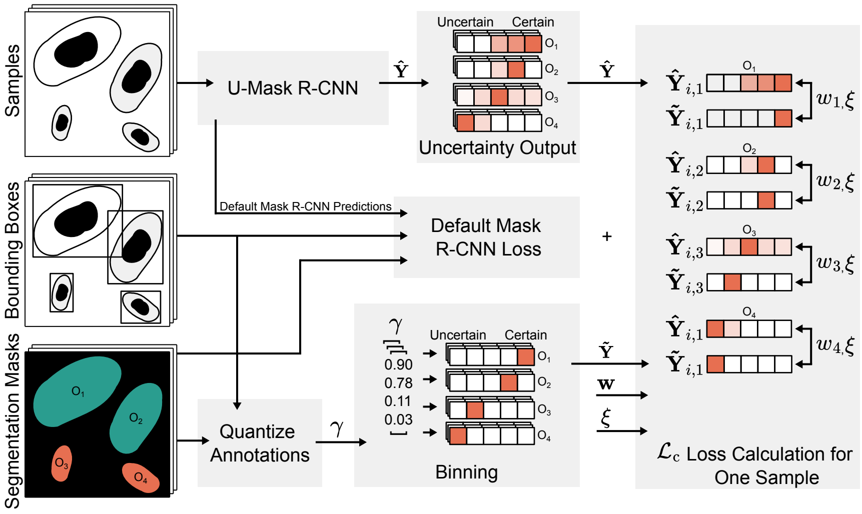


Figure 7.4: Visualization of the novel $\mathcal{L}_c$ loss calculation. First, the uncertainty predictions $\hat{\boldsymbol{Y}}$ and default Mask R-CNN loss values are calculated through the extended architecture. Simultaneously, the segmentation masks are quantized into their certainty values γ and then binned to discretize the uncertainties into the ground-truth matrix $\tilde{\boldsymbol{Y}}$. Then, the default Mask R-CNN loss and the novel $\mathcal{L}_c$ loss are calculated. $\tilde{\boldsymbol{Y}}$ and $\hat{\boldsymbol{Y}}$ are compared with a weight vector $\boldsymbol{w}$ and a similarity enforcing factor ξ. Here, the loss calulcation for one sample $\hat{\boldsymbol{Y}}_i$ / $\tilde{\boldsymbol{Y}}_i$ is shown. Adapted from [Ret24c].

converting it into a classification task. The confidence values γ (ranging from 0 to 1) are mapped to their corresponding bins, allowing the model to capture the bimodal uncertainty distribution without assuming an explicit parametric form.

The mapping from a continuous value to a classification is achieved through the function $\lfloor (k+1)\gamma \rfloor$. For training, a cross-entropy loss is employed to formulate the confidence estimation function, extended with a distance regularization term. This additional term imposes smaller penalties for smaller deviations from the ground truth. The loss function is defined as:

$$\mathcal{L}_c(\hat{\boldsymbol{Y}}, \tilde{\boldsymbol{Y}}, \xi, \boldsymbol{w}) = -\frac{1}{n} \sum_{i=1}^{n} \sum_{p=1}^{c} w_p \log \frac{\exp(\hat{Y}_{i,p})}{\sum_{j=1}^{c} \exp(\hat{Y}_{i,j})} \exp\left(\frac{-|\tilde{Y}_i - p|}{\xi} \right), \tag{7.6}$$

where $\hat{\boldsymbol{Y}} \in \mathbb{R}^{n \times c}$ are the predicted logits for all bins and $\tilde{\boldsymbol{Y}} \in \mathbb{R}^{n \times 1}$ contains the ground-truth bin indices for the batch, n is the batch size, $\boldsymbol{w}$ is a weight vector specifying the importance for each class, and ξ controls the steepness of the distance penalty: smaller ξ increases penalties for distant bins. $|\tilde{Y}_i - p|$ is the absolute difference between the ground-truth bin index $\tilde{Y}_i$ and the bin index p. The calculation of $\mathcal{L}_\mathrm{c}$ comprises two main parts: the cross-entropy term and the distance regularization term. The cross-entropy term encourages the model to predict the correct class, while the distance regularization term applies a smaller penalty for predictions close to the target class. This approach enables the mapping of confidence scores to bins, allowing the U-Mask R-CNN model to effectively handle multimodal distributions and apply varying degrees of penalization to uncertainties. The loss encourages high softmax scores near the true bin, not just exactly at it.

This novel objective function is integrated into the overall loss by simple addition to the $\mathcal{L}_2$[5]:

$$\mathcal{L}_2 = \mathcal{L}_\mathrm{s} + \mathcal{L}_{\mathrm{b}_2} + \mathcal{L}_\mathrm{c}, \tag{7.7}$$

where $\mathcal{L}_\mathrm{s}$ and $\mathcal{L}_{\mathrm{b}_2}$ are the original components of the Mask R-CNN loss function. These enhancements enable the U-Mask R-CNN architecture to estimate uncertainty at the object level in instance segmentation tasks while optimizing efficiency and minimizing human effort.

Combination of masks: The output of U-Mask R-CNN consists of m independent masks, each corresponding to a detected object. Several post-processing steps are applied to create a unified segmentation mask. To isolate certain objects ω, the m masks are filtered to retain only those where the index of the maximum value of the k bins of the label uncertainty loss is at least $k/2$. Conversely, to identify uncertain particles ψ, only masks where the maximum confidence is less than $k/2$ are considered. If no distinction is required, no filtering is applied to m. To prioritize the most confident predictions and facilitate the removal of

[5] See Appendix A.3.3 for details about the overall Mask R-CNN objective function.

duplicates, the retained masks are first sorted in descending order according to their predicted objectness scores $\hat{o}$. This sorting ensures that masks with higher confidence are processed first. Masks with $\hat{o}$ below the thresholds θ_ω for certain masks ω and θ_ψ for uncertain masks ψ are discarded, as they are considered less reliable. Afterward, to eliminate duplicates, the remaining masks are iterated through sequentially. Each mask is compared against all subsequent masks, and if the IoU score between two masks exceeds a predefined threshold $\theta_\cap$, the second mask is removed. Next, the retained masks are sorted in descending order based on their predicted objectness scores $\hat{o}$. Masks with $\hat{o}$ below the thresholds θ_ω for ω masks and θ_ψ for ψ masks are then discarded. The remaining masks are iterated through, and each mask is compared to all subsequent masks. If the IoU score between two masks exceeds a threshold $\theta_\cap$, the second mask is removed to eliminate duplicates.

In cases where the final prediction should include both ω and ψ masks, an additional post-processing step is performed. The IoU score is computed for each pair of masks from the ω and ψ categories. If the IoU score for a given pair exceeds a threshold θ_η, the ψ mask is removed to prevent overlapping information where an object is classified as both certain and uncertain. Finally, the segmentation masks are binarized using an activation threshold θ_Σ.

7.4 Results

This section presents the evaluations of experiments conducted to assess the novel evaluation metrics, the ArchSelect framework, and the proposed U-Mask R-CNN architecture. Details regarding NN training and hyperparameter tuning are available in Appendix A.7.1 and Appendix A.7.2.

7.4.1 Architecture Selection

Cyto-DS Splitting: The Cyto-DS[6] consists of synthetic grayscale cervical cytology images with controlled cell overlap levels, created to benchmark instance segmentation performance in challenging biological imaging scenarios. It is split into equal-sized training bins based on the general overlap criterion ψ_{GOC} for the evaluation of the ArchSelect framework to ensure that each subset contains the same amount of training data, allowing for a fair comparison. The resulting splits $\mathcal{S}$ with the ψ_{OFC} metric are shown in Table 7.1, and Figure 7.5 illustrates samples with varying overlap levels.

Subset	ψ_{OFC}	#Train	#Test
$\mathcal{S}_1$	0%	131	24
$\mathcal{S}_2$	1%	131	30
$\mathcal{S}_3$	12%	131	28
$\mathcal{S}_4$	19%	131	28
$\mathcal{S}_5$	26%	131	27
$\mathcal{S}_6$	37%	125	28

Table 7.1: The *subsets* of the Cyto-DS, split by the overlapping mesaure ψ_{GOC} into equal-sized bins with the ψ_{OFC} metric listed. *#Train* and *#Test* are the number of train and test samples.

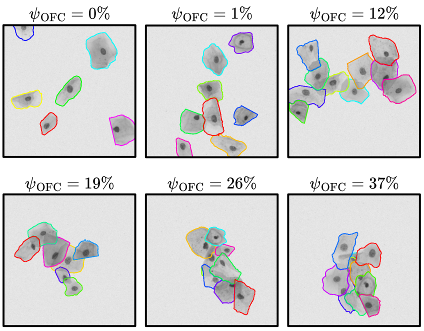

$\psi_{\text{OFC}} = 0\%$
$\psi_{\text{OFC}} = 1\%$
$\psi_{\text{OFC}} = 12\%$
$\psi_{\text{OFC}} = 19\%$
$\psi_{\text{OFC}} = 26\%$
$\psi_{\text{OFC}} = 37\%$

Figure 7.5: Six samples of the Cyto-DS for varying values of the overlapping measure ψ_{OFC}. The ground-truth masks are provided as overlays with unique colors per instance. Adapted from [Ret23a].

Quantitative Analysis: The AJI$^+$ scores for both U-Net and Mask R-CNN across all six subsets ($\mathcal{S}_1$-$\mathcal{S}_6$), along with their relative performance difference Δ (Mask R-CNN vs. U-Net), are presented in Table 7.2. U-Net performs well on the sub-dataset $\mathcal{S}_1$ with minimal overlap (AJI$^+$ = 95%) and even outperforms Mask

[6] See Section 4.2 for details regarding the Cyto-DS.

		$\mathbf{AJI^+}$		
Subset	ψ_{OFC}	**U-Net**	**Mask R-CNN**	$\mathbf{\Delta}$
$\mathcal{S}_1$	0%	95% ± 1%	90% ± 1%	−5%
$\mathcal{S}_2$	1%	84% ± 3%	87% ± 1%	+3%
$\mathcal{S}_3$	12%	78% ± 2%	81% ± 1%	+4%
$\mathcal{S}_4$	19%	64% ± 2%	72% ± 1%	+13%
$\mathcal{S}_5$	26%	54% ± 2%	63% ± 1%	+17%
$\mathcal{S}_6$	37%	41% ± 4%	55% ± 1%	+34%

Table 7.2: Results for each subset with the overlap amount ψ_{OFC}, the AJI$^+$s for both the U-Net and Mask R-CNN, and their performance difference Δ = Mask R-CNN - U-Net.

R-CNN by 5% (AJI$^+$ = 90%). However, even with slightly occluded instances to be segmented, a drastic decrease in the performance of U-Net is noticeable. This is apparent in subset $\mathcal{S}_2$, where U-Net drops sharply in performance (AJI$^+$ = 84%) compared to $\mathcal{S}_1$ and hence situates itself 3% below Mask R-CNN (AJI$^+$ = 87%), which also performs worse than in $\mathcal{S}_1$ but to a much smaller degree. The performance gap between U-Net and Mask R-CNN widens considerably as the degree of overlap increases across the subsets. For $\mathcal{S}_3$ with 12% overlap, Mask R-CNN (AJI$^+$ = 81%) outperforms U-Net (AJI$^+$ = 78%) by 3%. This advantage grows to 13% for $\mathcal{S}_4$ (19% overlap), 17% for $\mathcal{S}_5$ (26% overlap), and reaches a substantial 34% for the most challenging $\mathcal{S}_6$ subset with 37% overlap. Notably, the standard deviations for the U-Net tend to be larger than those for Mask R-CNN across all subsets apart from $\mathcal{S}_1$. This suggests that U-Net's segmentation results may be less consistent and more sensitive to variations in the input data, particularly as the degree of overlap increases. In contrast, Mask R-CNN demonstrates more stable performance, with smaller standard deviations, even on the more challenging subsets. Furthermore, the relative performance gap Δ between the two models increases as the degree of overlap grows. This suggests that Mask R-CNN's ability to handle overlapping instances improves more effectively than U-Net's as the segmentation task becomes more challenging. The substantial 34% performance difference on $\mathcal{S}_6$ underscores Mask R-CNN's superior capability to address heavily overlapped cell instances.

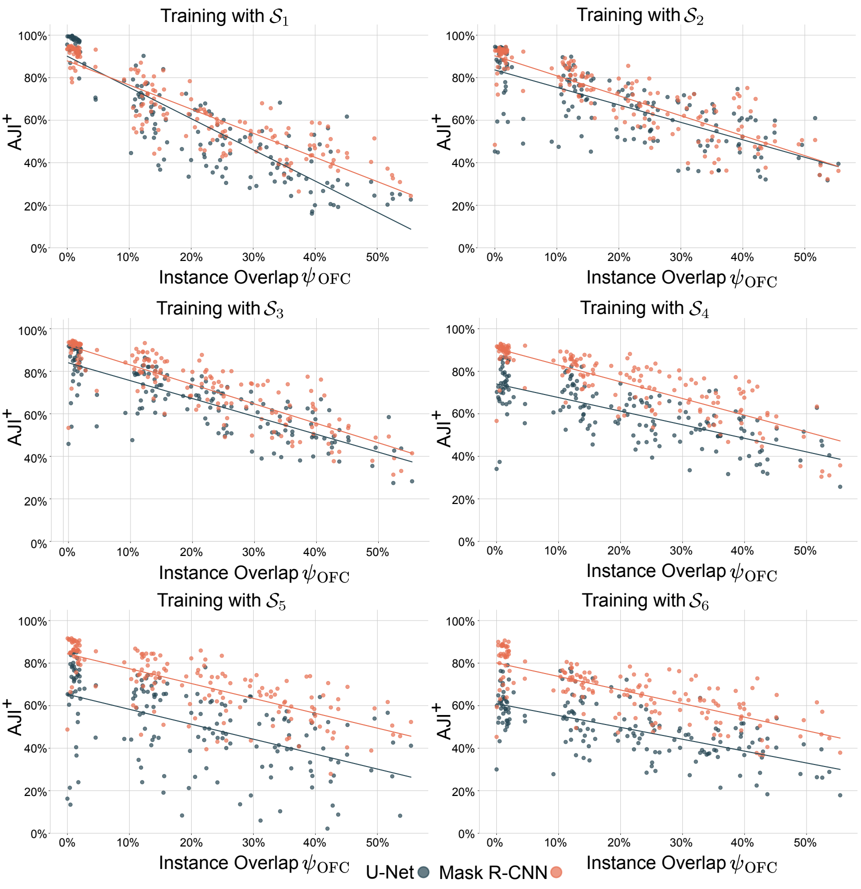


Figure 7.6: Scatter plot with the ψ_{OFC} metric on the x-axis and the AJI^{+} metric on the y-axis for U-Net and Mask R-CNN trained on each sub-dataset. The best model for training with the respective sub-dataset is shown. Each point is one sample from the combined test data, where all samples from all dataset splits are merged. A linear regression line is shown over the scatter points for both methods. Adapted from [Ret23a].

Figure 7.6 presents scatter plots of model performance for the best Mask R-CNN and U-Net models trained on each sub-dataset split. All test sets are merged, and each sample is plotted individually to show how performance changes with increasing overlap (ψ_{OFC}) between instances. Linear regression lines for both models in each subset highlight overall trends. The figure demonstrates that Mask R-CNN's advantage over U-Net becomes more pronounced as the training ψ_{OFC} increases, with greater separation along the AJI^{+} axis for more challenging, overlapping subsets. This confirms that Mask R-CNN is better suited for handling overlapping instances.

Even when trained on the non-overlapping subset $\mathcal{S}_1$, Mask R-CNN outperforms U-Net on samples with high ψ_{OFC}, indicating an ability to separate overlapping instances without explicit training on such cases. For $\mathcal{S}_2$ and $\mathcal{S}_3$, where overlaps are slight, both models perform similarly, though Mask R-CNN maintains a small but consistent lead that becomes clearer in $\mathcal{S}_3$. The difference becomes substantial in $\mathcal{S}_4$, where Mask R-CNN segments nearly all test samples better than U-Net. This trend continues and strengthens in $\mathcal{S}_5$ and $\mathcal{S}_6$, where Mask R-CNN maintains high performance while U-Net's results drop further. Notably, Mask R-CNN remains robust even on samples with little or no overlap, regardless of the overlap level in the training data. In contrast, U-Net's ability to generalize from high-overlap training to low-overlap samples is much more limited.

Qualitative Analysis: A qualitative analysis of the segmentation results complements the quantitative metrics, providing an intuitive understanding of the model's performance. Figure 7.7 shows a sample from each sub-dataset $\mathcal{S}_1$-$\mathcal{S}_6$, displaying the ground truth segmentation masks alongside those generated by U-Net and Mask R-CNN.

For the $\mathcal{S}_1$ sample, both U-Net and Mask R-CNN produce accurate masks that align well with the ground truth, confirming their effectiveness on non-overlapping cells. In $\mathcal{S}_2$, both models still perform well, but U-Net starts to show minor errors, such as splitting a single cell into two. This trend continues in $\mathcal{S}_3$, where U-Net's accuracy decreases slightly, though both models remain generally adequate. In $\mathcal{S}_4$, differences become more apparent. U-Net begins to merge distinct cells or split single cells incorrectly, while Mask R-CNN, although missing some instances, provides more plausible segmentations and demonstrates better generalization. With $\mathcal{S}_5$, Mask R-CNN's strengths are clear: it reliably detects cells and provides

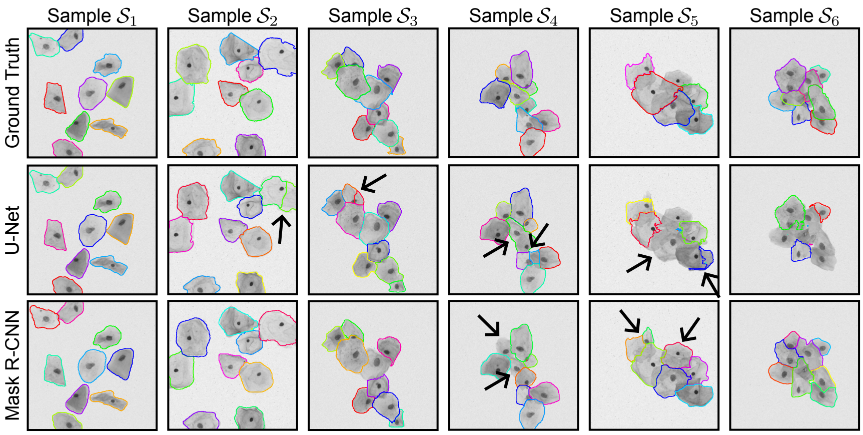


Figure 7.7: One sample from the test data of each sub-dataset $\mathcal{S}_1$-$\mathcal{S}_6$ with the ground truth instance segmentation masks, U-Net predictions, and Mask R-CNN predictions. The ground-truth masks are provided as overlays with unique colors per instance. Adapted from [Ret23a].

precise boundaries, even outperforming the ground truth in ambiguous cases. U-Net, by contrast, misses many cells and struggles with central regions. The most challenging sample, $\mathcal{S}_6$, shows the most noticeable gap: U-Net's segmentations degrade further, while Mask R-CNN continues to provide plausible results.

Overall, U-Net's performance is stable in simpler scenarios ($\mathcal{S}_1$–$\mathcal{S}_4$) but declines as complexity and cell density increase, often merging or missing cells in denser samples. Mask R-CNN consistently segments cells of varying sizes and maintains high boundary precision, especially in complex cases ($\mathcal{S}_5$ and $\mathcal{S}_6$), making it more suitable for crowded or challenging environments. U-Net remains a viable option for less complex datasets or when computational resources are limited.

7.4.2 Uncertainty Estimation

Dataset Combination: The SEM-DS[7] comprises hand-annotated SEM images of powder particles, including explicit uncertainty labels for each segmented

[7] Details about the dataset are provided in Section 4.3.2.

instance. The SEM-DS is used to evaluate the uncertainty estimation of the novel U-Mask R-CNN architecture as it contains fitting annotations. As SEM scans vary drastically at different magnifications, the SEM-DS is divided based on magnification rates. However, splitting the samples into distinct datasets reduces the number of samples available for the CNN, so it needs to be analyzed whether the distinction based on magnification benefits training. To assess the impact of this division, two experiments are conducted. First, the high magnification and low magnification splits are evaluated separately to measure their particle segmentation performance when trained in isolation. Second, both splits are combined into a single dataset containing all samples.

Observing the results in Table 7.3, it shows that training exclusively on low magnification data achieves the best performance on low magnification test images ($\text{AJI}^+ = 81\%$) but performs poorly on high magnification test images ($\text{AJI}^+ = 39\%$). This outcome is expected since the NN was not exposed to high magnification data during training. Conversely, training on high magnification data yields moderate performance across both test sets, with AJI^+ scores of 43% (low magnification) and 51% (high magnification), suggesting that features learned from high magnification data may generalize better than those from low magnification data, albeit with a trade-off in overall accuracy.

Training Data	**Test Data**	
	Low Magnification	**High Magnification**
Low Magnification	81% ± 2%	39% ± 8%
High Magnification	43% ± 3%	51% ± 3%
Combined	45% ± 2%	43% ± 2%

Table 7.3: The AJI^+ metric for the observed training data combinations. *Training data* refers to the data used for training the U-Mask R-CNN, while *Testing Data* is the data used for evaluation.

When both datasets are combined for training, performance on both test sets declines compared to training on individual datasets. Specifically, the low magnification test performance drops to 45%, representing a 36% decrease compared to training solely on low magnification data. Similarly, the high magnification test performance decreases to 43%, an 8% reduction relative to training exclusively on high magnification data. These findings suggest that combining datasets with distinct characteristics introduces domain conflicts that degrade segmentation performance. Hence, low and high magnifications are handled separately during CNN training to achieve optimal results.

Segmentation Results: Figure 7.8 provides a comprehensive comparison of the AJI$^+$ scores for U-Net and U-Mask R-CNN models on the SEM-DS, for both low and high magnification splits. The low magnification results reveal a striking

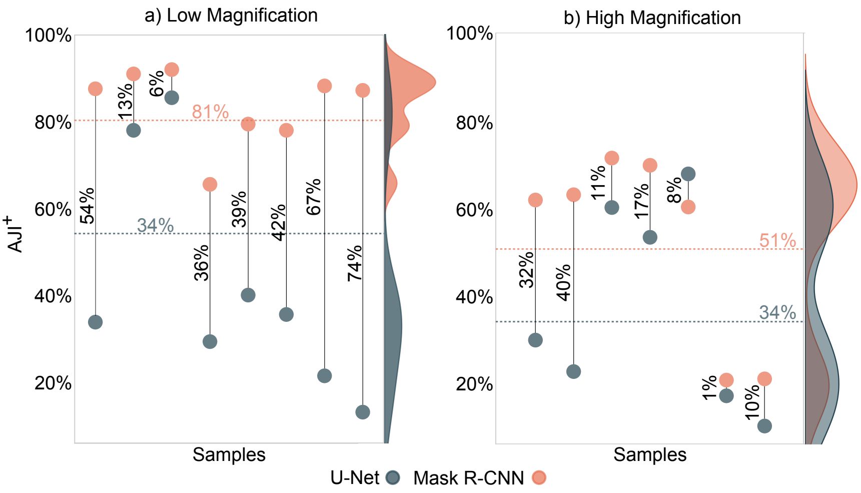


Figure 7.8: The AJI$^+$ for all test samples from both splits of the SEM-DS. Orange and gray circles represent AJI$^+$ values for U-Mask R-CNN and U-Net, respectively. Vertical lines connect results from the same image, highlighting the differences. Dashed horizontal lines indicate average AJI$^+$ values. Density distribution curves for both methods are displayed on the right. U-Mask R-CNN outperforms U-Net in all but one sample. Adapted from [Ret24c].

superiority of U-Mask R-CNN over U-Net. With an average AJI$^+$ score of 81% compared to U-Net's 55%, U-Mask R-CNN demonstrates drastically enhanced performance in handling low magnification images. The trend is consistent across almost all samples, with a single exception. The high magnification results show a similar trend, albeit with a narrower performance gap. U-Mask R-CNN achieves an average AJI+ score of 51%. In contrast, U-Net provides a much lower score of 55%, indicating U-Mask R-CNN's superior capability in managing the increased complexity inherent in high magnification images. Beyond the average scores, the results highlight the consistency of U-Mask R-CNN across different samples. In contrast, U-Net exhibits higher variability in its performance, particularly evident in high magnification images, where it struggles considerably. This inconsistency in U-Net's segmentation capabilities raises concerns about its reliability in complex imaging scenarios. The magnification's impact on the models' performance is noteworthy, especially in low magnification images. While this gap narrows slightly in high magnification images, U-Mask R-CNN maintains a clear advantage. The sophisticated U-Mask R-CNN architecture appears to be better equipped for delineating complex boundaries and overlapping instances often present in high magnification images. Interestingly, while U-Net performed better in scenarios with minimal overlap in previous results, it underperforms in this dataset at both magnifications. This discrepancy can be attributed to several factors. The SEM-DS likely contains more complex particle arrangements and morphologies than the previously used dataset, challenging U-Net's ability to distinguish individual instances. SEM images often have less distinct boundaries between particles, which may confuse the U-Net. The range of particle sizes in SEM images is diverse, favoring U-Mask R-CNN's region-based approach over U-Net's fixed receptive field.

These findings offer valuable insights for engineers and researchers working with SEM or similar data. The consistent superiority of U-Mask R-CNN across magnifications indicates it should be the preferred choice for complex and ambiguous image segmentation tasks, particularly when dealing with complex or overlapping structures. Given its better performance, allocating more computational resources to train and deploy U-Mask R-CNN may be justified, despite its higher complexity compared to U-Net. The results emphasize the importance of curating diverse datasets within a clearly defined domain and that more data does not automatically mean better performance when developing robust segmentation models. They also emphasize the importance of carefully considering specific data characteristics when selecting and implementing instance segmentation models. U-Mask R-CNN's superior performance suggests that its region-based approach is particularly well-suited for addressing the challenges presented by SEM images. It offers engineers a powerful tool for accurate object analysis within complex domains.

To obtain further insights into the superior performance of U-Mask R-CNN over U-Net, the segmentation results from two test samples across the two datasets are visualized in Figure 7.9. The first sample, captured at low magnification, reveals

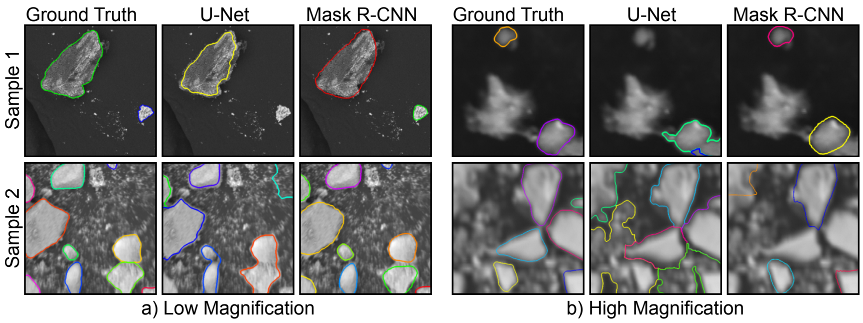


Figure 7.9: SEM image samples at low and high magnification with segmentation results. Expert-annotated ground truth overlays are compared to U-Net and U-Mask R-CNN model predictions. Overlay colors are arbitrarily assigned for each image to distinguish segments. Adapted from [Ret24c].

a distinct particle that U-Mask R-CNN segments accurately, while U-Net fails to detect it, likely due to its small size and irregular form. U-Net's performance is also subpar in the second sample, where it appears to rely on brightness thresholds to identify particles rather than actual structures. This results in small particles being incorrectly merged into a single large segment, presumably because the differences in brightness between the particles are too small. Additionally, U-Net erroneously segments the background in the top right corner of the image as being a particle, even though it is merely background noise. Conversely, U-Mask R-CNN successfully segments all individual particles, even those close to each other. Overall, U-Mask R-CNN performs superiorly in segmenting particles of various shapes and sizes at low magnifications, demonstrating its ability to distinguish complex structures and separate closely clustered particles.

The performance gap between U-Net and U-Mask R-CNN becomes even more evident when these models are applied to high magnification images, which typically exhibit increased blur, higher complexity, and fuzziness. In the first high magnification sample, U-Net inaccurately segments two particles with imprecise borders in the lower right corner. In contrast, U-Mask R-CNN delineates borders that align well with expert labels. In the second sample, U-Net segments large areas without detecting many individual particles due to considerable blur and low contrast. U-Mask R-CNN avoids segmenting the background and correctly identifies several larger particles, though it misses two smaller ones. Despite occasional false positives, U-Mask R-CNN excels in detecting individual particles in high magnification images, outperforming U-Net by effectively isolating objects in blurry, low-contrast backgrounds. The occasional discrepancies between ground truth and U-Mask R-CNN are mainly due to inherent labeling uncertainties from limited resolutions and complex morphologies. U-Mask R-CNN reliably identifies particle-like regions, while U-Net often misclassifies noise, underscoring U-Mask R-CNN's strength in adapting to challenging image conditions and positioning it as a promising framework for precise object delineation in complex visual environments.

Uncertainty Estimation: The ability of the U-Mask R-CNN model to predict confidences for individual instances using the novel uncertainty estimation is examined now as the next step. In segmentation, distinguishing between certain and uncertain instances is key to separating well-defined objects from ambiguous ones. This distinction is crucial for evaluating the reliability of segmentation results and pinpointing areas that may necessitate additional expert review.

Figure 7.10 illustrates the predicted uncertainties in comparison to the ground-truth annotations provided by human experts. The figure presents kernel density plots showing the predicted uncertainty distributions for low and high magnification images. The distributions clearly distinguish between certain and uncertain objects in low magnification images (Figure 7.10a). The green curve (certain objects) skews toward higher uncertainty scores (approaching 1). In comparison, the red curve (uncertain objects) concentrates at lower scores (approaching 0), demonstrating the model's ability to match the annotations by experts. The high

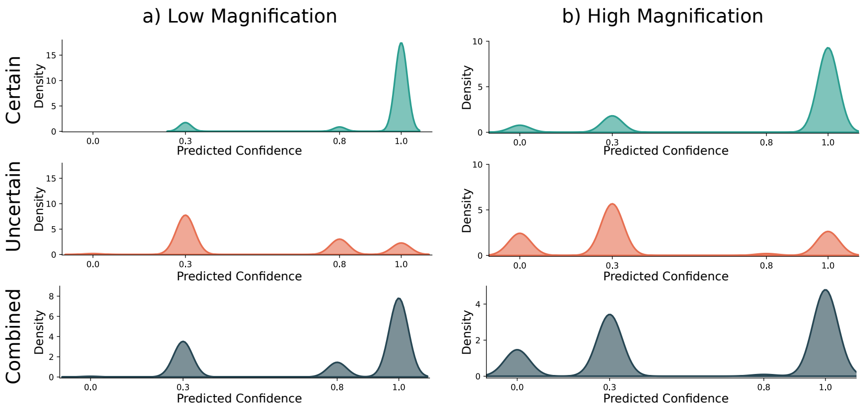


Figure 7.10: Kernel density plots showing the predicted uncertainty distributions of the U-Mask R-CNN model. The x-axis represents the predicted uncertainty scores (ranging from 0 to 1), while the y-axis shows the density values. The green curves represent objects classified as certain by domain experts, the red curve shows objects marked as uncertain, and the dark cyan curve displays the combined distributions of all objects without splitting by expert certainty. The clustering around the values 0.3, 0.8, and 1.0 is due to the discretized output of the U-Mask R-CNN. Adapted from [Ret24c].

magnification results (Figure 7.10b) exhibit similar trends but with slightly greater overlap between distributions, likely due to increased complexity at higher scales. Despite this overlap, the model maintains effective discrimination, assigning high confidence to certain objects and low confidence to uncertain ones. Both magnification levels display bimodality in the combined distribution (dark cyan curve), showing that the learning approach developed can model such distributions. The uncertainty estimation capabilities are robust and generalize well across the two magnification scales. Overall, the uncertainty estimation is successfully incorporated into the U-Mask R-CNN model, with good alignment to expert annotations maintained and effectiveness preserved across different magnification levels. This capability enhances the utility in real-world applications, where understanding the confidence of segmentation results is crucial for decision-making and further analysis.

To evaluate the model's accuracy across difficulty levels and gain behavioral insights, the AJI$^+$ scores are separately compared for certain and uncertain particle predictions in Table 7.4. This separation clarifies how segmentation performance

Magnification	**Uncertain**	**Certain**	**Combined**
Low	$23\% \pm 5\%$	$81\% \pm 2\%$	$81\% \pm 2\%$
High	$23\% \pm 2\%$	$51\% \pm 3\%$	$51\% \pm 3\%$

Table 7.4: Comparison of AJI$^+$ scores for the *certain* particles only, only *uncertain* ones, and if all objects are *combined*, so there is no distinction between certainties.

varies with the certainty of annotations. A noticeable performance gap exists between certain and uncertain predictions at both magnifications. At low magnification, the AJI$^+$ for certain particles is substantially higher than for uncertain ones. Similarly, at high magnification, uncertain labels underperform compared to certain ones. Grouping all particles (without certainty distinctions) yields AJI$^+$ scores identical to those of certain labels. The consistent performance on uncertain labels across magnifications suggests the model's uncertainty handling is magnification-independent. However, the pronounced drop in certain-label performance from low to high magnification indicates that high magnification images present greater challenges for confident predictions.

To confirm the previous results and gain visual insights into the segmentation performance with uncertainty estimation, Figure 7.11 displays four samples from both magnifications, along with the predictions of the U-Mask R-CNN and the ground truth annotations. In *Sample 1* of low magnification, the model accurately identifies and segments both the certain and uncertain particles correctly, even though they differ noticeably in appearance. *Sample 2* shows how the model can segment and estimate the certainty even if the particles are difficult to distinguish due to noise. Further, in *Sample 3*, the segmentations are mostly accurate, but a small area at the edge of a particle is incorrectly divided and marked as uncertain.

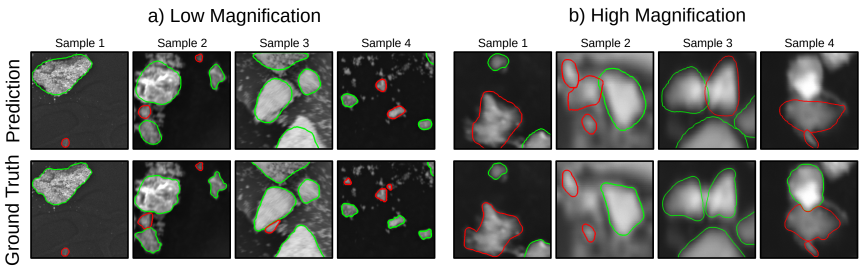


Figure 7.11: Two sample images for low and high magnification with segmentation predictions by the U-Mask R-CNN. For comparison, the ground truth annotations as shown. Green and red overlays represent certain and uncertain labels, respectively. The predictions of the U-Mask R-CNN model are considered uncertain if the confidence of particles is $\leq 50\%$. Adapted from [Ret24c].

In *Sample 4*, a particularly difficult case is shown in which the prediction of the U-Mask R-CNN and the ground truth annotations do not match exactly. Two small uncertain particles are missed, and there is some confusion about the certainty of the detected instances.

The four high magnification samples further confirm the capability of the model. *Sample 1* shows two detections of each certainty level that are correctly segmented and classified. In *Sample 2*, the certain particle is segmented adequately. Still, there is some confusion about the uncertain predictions, which is probably due to the large amounts of noise in the image. *Sample 3* shows four correct segmentations but one misclassification, and in *Sample 4*, the two uncertain particles are detected correctly, but the certain one is missed.

Comparison to Human Experts: The U-Mask R-CNN displays convincing results; however, it sometimes disagrees with the ground truth annotations and provides results that appear visually plausible. Hence, additional validation by multiple human experts is pivotal for gaining further insights and ensuring practical applicability. Further, aligning with human expertise and being universally applicable to diverse challenges is essential. Since the LGS-DS contains annotations from three experts for each sample and a chemical compound not included in the SEM-DS, it is well-suited for evaluating both aspects.

Trained solely on the low magnification split of the SEM-DS, the U-Mask R-CNN model predicts an average particle area of 37 µm^2 when evaluated on the LGS-DS. This calculates an average diameter of 6.7 µm when assuming spherical particles, a common assumption within inorganic SEM analysis [Bal23]. This value agrees well with the expected particle size of $LiCoO_2$ powders, which typically range from 5-10 µm. Further, the model's prediction closely matches the average particle area of 31 µm^2 derived from the expert annotations. Figure 7.12 displays the distribution of sizes by the U-Mask R-CNN model and the ones provided by three domain experts. The average particle area, as determined by the three

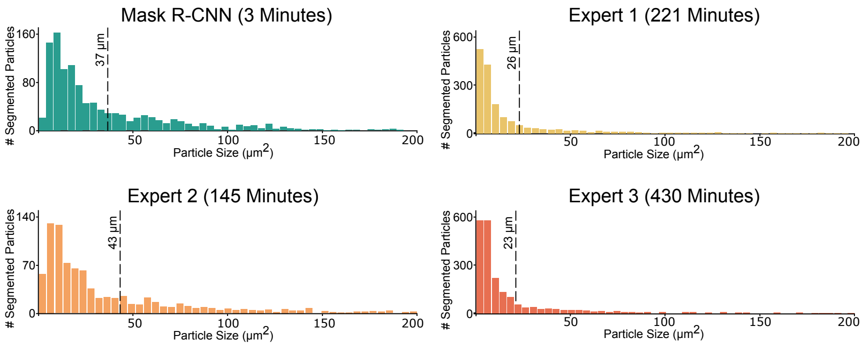


Figure 7.12: Comparison of the detected particle size distributions for the U-Mask R-CNN model and three domain experts on the LGS-DS. The U-Mask R-CNN model is trained on low magnifications. The required time for each annotation method is listed in minutes. The vertical dashed lines denote the average particle size. Adapted from [Ret24c].

domain experts, ranged from 23 to 43 µm, highlighting the subjective nature of manual particle analysis. This discrepancy highlights the challenges inherent in particle segmentation, particularly when dealing with smaller particles. The particle size distributions generated by both the U-Mask R-CNN model and the human experts exhibit similar qualitative characteristics. A peak in particle count is visible at around 10 µm^2, accompanied by a long tail extending to particles as large as 200 µm^2. However, noticeable differences were noted in the detection of smaller particles. *Experts 1* and *3* identified several hundred particles smaller than 10 µm^2, while *Expert 2* labeled fewer than 100 in this size range. The

U-Mask R-CNN model tends to underestimate the number of very small particles compared to two of the experts. This discrepancy between the human experts is further highlighted when calculating the mean AJI^+ across all samples among the experts and our model as the agreement. The agreement between *Expert 1* and *Expert 2* is 65 ± 20, between *Expert 1* and *Expert 3* is 58 ± 23, and between *Expert 2* and *Expert 3* is 56 ± 21. Comparatively, the agreement between U-Mask R-CNN and *Expert 1* is 61 ± 21, with *Expert 2* is 64 ± 21, and with *Expert 3* is 60 ± 23. These results indicate that, on average, U-Mask R-CNN's predictions align well with the experts' labels despite the considerable variations among the annotations. Furthermore, this demonstrates that U-Mask R-CNN is well-equipped to learn from noisy and inaccurate annotations, as it can detect particles from previously unseen compounds. One further great advantage of the U-Mask R-CNN is its efficiency. The model completed the analysis of the entire dataset in just 3 minutes on a desktop CPU, requiring no manual intervention. In contrast, the time invested by human experts ranged from 145 to 430 minutes. The model produced results that qualitatively matched the manual labels, which required 10 to 100 times longer to create. Another critical factor is the minimum size of instances that U-Mask R-CNN can detect. The particle sizes identified by the three experts span from 0.01% to 9.41% of the entire image, whereas U-Mask R-CNN predicts particle sizes between 0.08% and 9.41%. Despite U-Mask R-CNN's ability to detect particles as small as 0.08% of the image, it struggles with particles as tiny as 0.01%, which the experts annotated. This indicates a challenge for U-Mask R-CNN in identifying particularly small instances.

7.5 Discussion

The ArchSelect framework introduces a systematic method for evaluating dataset complexity and choosing suitable NN architectures. By quantifying instance overlap through the novel metrics ψ_{CFM}, ψ_{GOC}, and ψ_{OFC}, it becomes possible to objectively assess the difficulty of segmentation tasks. This method is particularly beneficial for tasks with overlapping instances, facilitating informed decision-making during architecture selection, which can save substantial time and computational resources. The comparison between U-Net and Mask R-CNN on the Cyto-DS demonstrates the practical applicability of this technique. While U-Net excels in simple, non-overlapping scenarios, Mask R-CNN increasingly surpasses it as the complexity of overlap increases. This trend demonstrates that the choice of architecture becomes increasingly critical as dataset complexity increases and that no single architecture universally suits all scenarios.

The novel uncertainty-aware U-Mask R-CNN architecture, introduced with the extended label uncertainty loss $\mathcal{L}_c$, addresses a critical gap in instance segmentation tasks where ambiguity exists regarding the segmentation of individual objects. By providing instance-level confidence scores, this method enables researchers to identify challenging or ambiguous objects that may require further expert analysis. The evaluation on the SEM-DS and LGS-DS demonstrates the practical applicability of this method. The U-Mask R-CNN outperforms U-Net in complex segmentation tasks, yielding results comparable to those of human experts in a fraction of the time. The uncertainty estimation results reveal clear distinctions between certain and uncertain objects. This capability enables a nuanced analysis of segmentation results, potentially reducing the need for manual review of entire images. While integrated into the U-Mask R-CNN architecture in this work, the main idea of this developed uncertainty-estimation approach applies to all NN architectures that provide feature extraction.

The methods developed address key challenges in instance segmentation workflows. The ArchSelect framework enables data-driven optimization of model choices, and the U-Mask R-CNN architecture allows users to identify and prioritize ambiguous predictions, substantially reducing manual review while maintaining expert-level accuracy. Together, these approaches form a powerful toolkit, enabling practitioners to select the optimal architecture for their dataset and utilize uncertainty scores to identify critical instances. This integrated workflow allows efficient resource allocation, reliable automation, scalable analysis, and easy adaptation to new conditions. By bridging theory and practice, these methods provide robust tools for real-world segmentation, from optimizing computation to streamlining quality control in high-throughput pipelines.

8 Conclusion & Outlook

This work addressed a critical bottleneck in modern deep learning: the limited availability of high-quality, annotated data, particularly in specialized domains such as biomedicine and materials science. The motivation stems from the understanding that the full potential of deep learning remains untapped due to the laborious and costly nature of data annotation, which hinders the efficient use of vast amounts of raw data generated by automated processes. This work enhances the robustness and efficiency of deep learning by developing novel methodologies that minimize the need for extensive manual annotation and maximize the utilization of available data.

By addressing key challenges at multiple stages of the deep learning pipeline, from data preparation and optimized utilization to model architecture enhancement and training, this work provides a comprehensive strategy for improving the robustness and efficiency of modern deep learning. First, in *algorithmic dataset enhancements*, automated techniques are developed to generate high-quality segmentation masks and to effectively utilize heterogeneously labeled data, supported by new objective functions and evaluation metrics. Second, in *deep learning with unannotated data*, comprehensive frameworks are introduced to systematically evaluate and optimize approaches that extract valuable information from raw data, enabling improved model performance in situations with limited labeled data. Third, the section on *selection and refinement of neural networks* presents methods for assessing dataset complexity, selecting suitable neural network architectures, and incorporating uncertainty estimation to improve model reliability. Collectively, these contributions reduce the dependency on manual annotation, enhance robustness to real-world data ambiguities, and increase resource efficiency. This work provides a comprehensive framework for applying deep learning in resource-constrained scientific and industrial domains. The methodologies demonstrate consistent performance improvements across biomedical imaging and materials science applications, highlighting their generalizability beyond conventional benchmark datasets. Summarized, the main contributions of this work are as follows:

1. The Self-SL Benchmarking Framework (SelfSLBench) gives a structured and fair way to compare different methods that learn valuable information from images without needing expert-provided labels.
2. The Self-SL Automatic Tuning Framework (SelfSLAutoTune) automatically finds the optimal settings for methods that learn from unlabeled data, thereby improving performance without requiring manual trial and error.
3. The novel Uncertainty-Mask R-CNN (U-Mask R-CNN) incorporates uncertainty in expert labels, yielding reliable results in challenging situations.
4. The label uncertainty objective function $\mathcal{L}_{\mathrm{c}}$ considers expert confidence, enabling the U-Mask R-CNN to be learn ambiguous expert information.
5. The introduced combined objective function $\mathcal{L}_{\lambda}$ allows deep learning models to be trained effectively on datasets where not every image has the same type or amount of expert input.
6. The Controlled Heterogeneity Generator (CoHeG) automatically generates datasets with partial labels from a fully annotated source, enabling the evaluation of algorithms that learn from inconsistent or incomplete expert information.
7. The Architecture Selection Framework (ArchSelect) provides a systematic approach for choosing deep learning models in complex, real-world tasks containing overlapping objects.
8. The Neural Bootstrapping Framework (NeuBoot) utilizes basic algorithmic procedures to create initial outlines of objects in images in combination with deep learning models to generate precise pixel-wise annotations that require little manual effort.

9. The Performance Frugality Ratio (ψ_{PFR}) measures how efficiently available expert information is used in datasets with incomplete annotations.
10. The novel Class Frequency Metric (ψ_{CFM}), General Overlap Criterion (ψ_{GOC}), and Overlap Frequency Criterion (ψ_{OFC}) collectively evaluate deep learning methods based on dataset complexity.
11. The novel imaging datasets Heart-DS, SEM-DS, LGS-DS, and A-SEM-DS offer new multi-domain image collections with diverse expert annotations, varying completeness, and uncertainty estimates, enabling comprehensive testing and validation of all developed methods in realistic and challenging scenarios.

Looking ahead, several promising research directions emerge. One area is enhancing the detection of initial object boundaries in raw data using the NeuBoot framework. Although current methods are effective, integrating more advanced algorithms could further enhance dataset improvement, especially in complex scenarios. Another critical step is to test the robustness of the developed approaches with a broader range of data types and qualities, especially in particularly challenging domains. This would further demonstrate the effectiveness of the methods in even more real-world settings and potentially reveal new ways to enhance their strength. As self-supervised learning continues to develop quickly, using the SelfSLBench and SelfSLAutoTune frameworks on novel approaches, yet to be developed, enables quick and easy comparison, integration, and optimization of such new methods. Furthermore, collaborative efforts to integrate and extend the developed methods across various fields of research can accelerate progress and help further explore the potential of deep learning in data-scarce environments. Pursuing these directions will not only refine and validate the developed methods but also pave the way for innovative applications of deep learning in data-scarce environments. Ultimately, this dissertation comprehensively advances the state of deep learning by systematically overcoming key barriers in data annotation, utilization, and model robustness, thereby enabling more reliable and efficient deployment in domains where the volume and complexity of generated data far exceed what can be processed by human experts.

A Appendix

A.1 Software Packages

The main `python` software packages utilized are provided in Table A.1. For package management, `conda`[1] is employed.

Software Package	Version
`numpy` [Har20]	1.24.3
`pytorch` [Pas19]	1.11.0
`pytorch lightning` [Fal19]	1.2.4
`torchvision` [Mai16]	0.12.0
`tifffile` [Goh16]	2021.11.2
`scikit-image` [VdW14]	0.18.3
`scikit-learn` [Ped11]	1.0.1
`scipy` [Vir20]	1.7.1
`albumentations` [Bus18]	0.5.2
`pytorch gradcam` [Gil21]	0.2.1
`torch cka` [Sub21]	0.21
`opencv-python` [Bra00]	4.5.4.60
`wandb` [Bie20]	0.12.6
`minkowski engine` [Cho19]	0.5.4

Table A.1: The main `python` packages used in this work along with their versions.

[1] https://docs.anaconda.com/miniconda/

A.2 Implementation Details

All NN architectures are implemented in PyTorch Lightning [Fal19]. The Albumentations [Bus18] library is used to implement image augmentations. For visualizing the CAMs, the PyTorch Grad-CAM library [Gil21] is employed. To calculate and display CKAs, the PyTorch Model Compare [Sub21] library is utilized. For sparse convolutions, the Minkowski Engine [Cho19] is applied.

Small-scale experiments are conducted on a local machine equipped with an AMD Ryzen 9 5950X 16-core CPU and an NVIDIA GeForce GTX 3090 GPU. Large-scale experiments are performed on the Hochleistungsrechner Karlsruhe (HoreKa) supercomputer or the Helmholtz AI Computing Resources (HAICORE) partition. HoreKa contains 813 nodes (individual servers) and HAICORE 16 nodes, each equipped with an Intel Xeon Platinum 8368 CPU (2 sockets, 76 cores per socket) and four NVIDIA A100 Tensor Core GPUs. Training HoreKa and HAICORE is conducted using Simple Linux Utility for Resource Management (SLURM) [Yoo03] for job scheduling and PyTorch distributed [Li20] for training on multiple nodes simultaneously.

A.3 Technical Foundations

A.3.1 ResNet Details

A default neuron in a NN with the activation function ϕ, weights ω, and inputs $\boldsymbol{x}$, learns the mapping $f(\boldsymbol{x}) = \phi(g(\omega, \boldsymbol{x}))$. In contrast, a standard layer in a ResNet, also called a residual block, can be represented as $f(\boldsymbol{x}) = \phi(g(\omega, \boldsymbol{x}) + \boldsymbol{x})$. This addition of the input to the output enables the network to learn residual functions related to the layer inputs, rather than learning functions that are not referenced. This enables the network to set the residual to zero and effectively skip the layer, making it easier to learn identity functions and facilitating the successful training of much deeper networks [He16].

The differences between ResNet variants lie in their depth and the structure of their residual blocks. Shallower networks like ResNet-18 and ResNet-34 use simpler blocks with two 3×3 convolutional layers, while deeper networks employ bottleneck blocks with 1×1, 3×3, and 1×1 convolutions to reduce parameters and computation. Pretrained weights for various tasks for the ResNet architecture are readily available online [Pas19]. Further details can be found in the literature [Goo16, He16].

A.3.2 U-Net Details

Observing the U-Net architecture in Figure 3.11, the encoder includes layers, each with two 3×3 convolutional layers followed by a ReLU activation and 2×2 max-pooling, which halves the feature map size. The number of feature maps doubles with each layer, starting at 64. The standard U-Net encoder has four layers and two additional 3×3 convolutional layers without max-pooling. During encoding, the maximum values and their indices from the max-pooling operations are stored for use in upsampling.

The decoder mirrors the encoder's structure in reverse. During decoding, max-unpooling is used, where all non-maximal values are set to zero. A transposed convolution operator then processes the upsampled feature maps to match the encoder's depth. Skip connections link the pooling and unpooling operations by concatenating feature maps from the encoder to the decoder, preserving high-resolution information from the input. Padding operations ensure that feature maps maintain their size after convolution.

A.3.3 Mask R-CNN Details

Feature Pyramid Network: The FPN leverages multiple intermediate layers from the ResNet backbone to build hierarchical features. Starting from lower levels, lateral connections propagate higher-resolution features downward, preserving fine-grained information across scales. Features are upsampled and merged smoothly to create unified multi-scale representations.

Region Proposal Network: The RPN slides a small CNN window over feature maps, predicting rectangular regions (proposals) with associated objectness scores. Non-maximum suppression subsequently filters overlapping proposals.

Region of Interest Align: ROI align crops and resizes proposed regions from original images to a fixed size (7×7 pixels), ensuring spatial consistency for subsequent processing.

Mask R-CNN Loss Function: The loss function of the Mask R-CNN is closely linked to the architecture and rarely exchanged (unlike with other ANN architectures). It comprises two main components, $\mathcal{L}_1$ and $\mathcal{L}_2$, each addressing specific aspects of the model's performance. The first component, $\mathcal{L}_1$, includes the objectness classification $\mathcal{L}_\mathrm{o}$ and the bounding box regression $\mathcal{L}_{\mathrm{b}_1}$. The purpose of $\mathcal{L}_\mathrm{o}$ is to differentiate between foreground and background areas within a predicted bounding box using a softmax cross-entropy loss. The aim of $\mathcal{L}_{\mathrm{b}_1}$ is to train the model to predict bounding box offsets accurately for proposed regions, calculated using the Huber loss. The overall $\mathcal{L}_1$ is a weighted sum of $\mathcal{L}_\mathrm{o}$ and $\mathcal{L}_{\mathrm{b}_1}$:

$$\mathcal{L}_1 = \mathcal{L}_\mathrm{o} + \lambda_{\mathrm{b}_1}\mathcal{L}_{\mathrm{b}_1}, \tag{A.1}$$

where λ_{b_1} is a parameter that determines the significance of the bounding box regression in the total loss. In this work λ_{b_1} is always set to 1.

The second component, $\mathcal{L}_2$, is composed of three elements: the segmentation loss $\mathcal{L}_\mathrm{s}$, the refined bounding box regression loss $\mathcal{L}_{\mathrm{b}_2}$, and the confidence loss $\mathcal{L}_\mathrm{c}$. The segmentation loss $\mathcal{L}_\mathrm{s}$ is designed to produce object masks for each ROI. Unlike traditional segmentation architectures, such as U-Net, there is no competition among classes when generating masks. Each ROI is handled independently, and the Binary Cross-Entropy Loss is computed for each detected object. Building on $\mathcal{L}_{\mathrm{b}_1}$, $\mathcal{L}_{\mathrm{b}_2}$ further refines the bounding box coordinates for the ROIs to better match the ground truth bounding boxes, using the Huber Loss. The total $\mathcal{L}_2$ is the sum of its individual parts:

$$\mathcal{L}_2 = \mathcal{L}_\mathrm{s} + \mathcal{L}_{\mathrm{b}_2}. \tag{A.2}$$

Finally, the overall loss is the sum of the two components

$$\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2. \tag{A.3}$$

A.3.4 ConvNeXt Details

The macro and micro design changes of the ConvNeXt architecture collectively contribute to ConvNeXt's improved performance, allowing it to compete with and sometimes outperform transformer-based models while retaining the simplicity and efficiency of CNNs [Woo23]. ConvNeXt is available in various scales, ranging from ConvNeXt Atto with 3.4 million parameters to ConvNeXt Huge with 657 million parameters, offering a range of models to suit different computational budgets and performance requirements.

Macro Design Changes: The macro design changes in ConvNeXt focus on high-level structural decisions that affect the overall architecture. ConvNeXt adopts the distribution of computational blocks across the network's stages, following the successful transformer architectures [Liu21b]. A stem block at the beginning of the network acts as a simplified patchify layer, consisting of 4×4 non-overlapping convolutions with stride 4, followed by layer normalization. This efficiently downsamples the input image and divides it into distinct patches. ConvNeXt also incorporates depthwise convolutions, reducing the number of floating-point operations while widening the network's capacity. Inspired by transformer blocks and MobileNetV2 [San18], ConvNeXt adopts an inverted bottleneck structure, expanding the hidden dimension in each block to four times the input dimension. Larger kernel sizes (up to 7x7) in depthwise convolutions increase the receptive field.

Micro Design Changes: Micro design changes in ConvNeXt involve lower-level modifications to individual layers and components. The ReLU activation is replaced with the GELU, which performs better in various tasks and is commonly used in transformer models. Layer Normalization is used instead of Batch Normalization, aligning more closely with transformer architectures and contributing to improved performance. ConvNeXt introduces separate downsampling layers with 2x2 convolutions and normalization, similar to the approach used in Swin Transformers [Liu21b], helping to maintain spatial information during downsampling operations.

A.3.5 ConvNeXtV2 Details

Fully Convolutional Masked Autoencoder: FCMAE treats individual patches of the input image after the stem block as a set of 2D sparse arrays of pixels. It uses sparse convolutions to process only the visible parts of the input image, preventing mask pattern dissipation in traditional CNNs during hierarchical convolution [Cho19, Pei24]. The input image is masked by removing a predefined number of patches after the stem block. The masked input is then processed through the ConvNeXt backbone, modified with sparse convolutions. This step allows the network to extract features from the partially obscured image, forcing it to develop a more generalized understanding of image structures and patterns. After that, a lightweight decoder is used to reconstruct the original image from the encoded representation. This reconstruction task encourages the network to learn meaningful and comprehensive features that can capture the essence of the input image, even with missing information.

Global Response Normalization: The GRN enhances inter-channel feature competition and addresses the issue of feature collapse observed when training ConvNeXt directly on masked input. It operates by first computing global statistics across spatial dimensions and a subset of channels, then normalizing feature responses, and finally applying a learnable affine transformation. This process helps maintain diverse and informative feature representations throughout the network, which is crucial for both Self-SL and supervised learning tasks. The ConvNeXtV2 architecture incorporates the GRN layer into its block structure, placing it between two convolutions to enhance inter-channel feature competition and promote feature diversity throughout the network.

A.4 Datasets

A.4.1 Download Details for Novel Datasets

Table A.2 provides the download links for all novel datasets introduced in this work. Additionally, details and code for processing the SEM-DS dataset are available at: https://github.com/lrettenberger/Uncertainty-Aware-Particle-Segmentation-for-SEM-Data.

Dataset Name	**Download Link**
Extended Heart-DS	https://osf.io/uyk79/
SEM-DS	https://osf.io/f2y8w/
LGS-DS	https://osf.io/f2y8w/
A-SEM-DS	https://osf.io/yzrfv/

Table A.2: Download links for all novel datasets developed in this work.

A.4.2 LGS-DS Pre-Processing

For the LGS-DS, a low-pass filter is applied on the images before resizing to smooth high-frequency artifacts in the downscaled scans (see Figure A.1). The image resized without a low-pass filter shows considerable high-frequency noise and artifacts due to the lack of pre-filtering before resizing. The filtered image is smoother and cleaner, with reduced noise and fewer artifacts, demonstrating the effectiveness of the low-pass filter for SEM images.

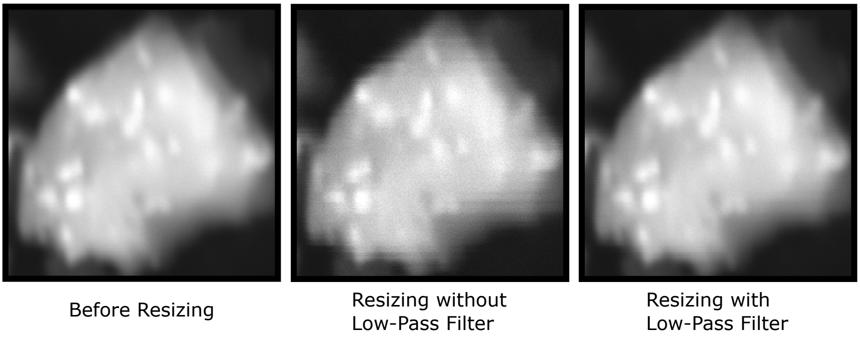

Before Resizing
Resizing without Low-Pass Filter
Resizing with Low-Pass Filter

Figure A.1: Particle before and after resizing. On the left is the original image, in the middle is the downscaled image without a low-pass filter, and on the right is the image after resizing with a low-pass filter. Adapted from [Ret24c].

A.5 Training Details for Algorithmic Dataset Enhancement

A.5.1 Segmentation Mask Generation

The Dice Loss [Sud17] is employed as the objective function, and the Adam [Kin14] optimizer with a learning rate of 0.001 is utilized in all experiments. All samples are resized to 256×256 pixels for both the ISIC-DS and the Droplet-DS. Various data augmentation techniques are applied during training, including horizontal and vertical flipping, blurring, Gaussian noise, rotations, scaling, brightness variations, and contrast jittering. Detailed information about the data augmentation process can be found in the project repository[2]. The data is randomly divided into an 80%, 20%, and 20% split for training, validation, and testing, respectively. The entire training dataset comprises the segmentation masks generated by $\mathcal{T}$. The actual annotations are only made available for evaluation during test time. All samples are normalized to fall within the range of 0-1. The U-Net architecture

[2] https://github.com/Heterogeneous-Semantic-Segmentation/Reducing-Annotation-Efforts-by-DNN-Bootstrapping

[Ron15] is implemented using PyTorch Lightning, with a minor modification to accept RGB images while maintaining its original structure. The k-means algorithm is implemented using the OpenCV [Bra00] package, while Otsu thresholding is performed using scikit-image [VdW14]. For the Otsu algorithm, ISIC-DS images are converted to grayscale, as it requires mono-channel input. All experiments are repeated four times with different random seeds to mitigate initialization effects, and mean metrics with standard deviations are reported. The evaluation is consistently performed using a test split from the respective dataset.

A.5.2 Heterogenous Labeling

For training, a set of data augmentations is utilized on the extended Heart-DS. Each sample and the corresponding segmentation mask are always augmented in the same way:

- Rotations in range: [0,0.3] degrees,
- Width shift in range: [-0.05,0.05],
- Height shifts in range: [-0.05,0.05],
- Shear angles in range: [0,0.05] degrees (counter-clockwise direction),
- Zoom in range: [0,0.5],
- Horizontal flips with 50% probability,
- Fill mode: Nearest.

Additionally, all images and their corresponding annotation masks are scaled to a size of 256×256 pixels and pixel intensities in the range of $[0, 1]$. The dataset is divided into training, validation, and test sets using a 60% / 20% / 20% split, respectively. As the extended Heart-DS contains sequences of heartbeats, all dataset splits (including the test set) are divided exclusively between complete heartbeat sequences, preventing partial sequences from appearing in multiple subsets. This approach preserves temporal integrity across training, validation, and testing data and avoids leakage of information between sub-datasets. Evaluations are always conducted on the test split of the dataset. The U-Net model is trained until there are no improvements in the DSC metric on the validation data. A batch size of 11 is used. The Adam optimizer [Kin14] with a learning rate of 10^{-3} is used. All experiments are repeated ten times.

In the general ablation study, results remain consistent for up to 60% of dropped labels, with a standard deviation not exceeding 0.04 for the DSC metric. At 60%, the mean DSC is $78\% \pm 3\%$ (see Table A.3). The fully labeled ventricle class demonstrates expected stability in the transfer learning scenario, with a single exception at 80% dropped labels due to an outlier in one experiment. For the atrium class at 60% dropped labels, the mean DSC is $81\% \pm 5\%$ (see Table A.4). When combining two datasets, the results for each configuration remain approximately constant across all runs, regardless of how the combined dataset is divided. An even split of the dataset between both classes (50% / 50% share) yields a mean DSC of $74\% \pm 5\%$ (see Table A.5). In the trade-off between horizontal and vertical loss, minimal variation is observed across all runs. At $\alpha = 0.4$, the mean DSC is found to be $87\% \pm 1\%$ (see Table A.6).

Dropped	**mDSC**				$\psi_{\mathbf{PFR}}$ (Mean)
Labels [%]	Atrium	Bulbus	Ventricle	Mean	
0	**85% ± 2%**	**78% ± 2%**	**88% ± 1%**	**84% ± 2%**	**0.71**
10	84% ± 3%	79% ± 2%	87% ± 2%	83% ± 2%	0.77
20	83% ± 1%	79% ± 1%	87% ± 1%	83% ± 1%	0.86
30	83% ± 2%	78% ± 2%	85% ± 2%	82% ± 2%	0.96
40	83% ± 2%	78% ± 3%	86% ± 1%	82% ± 2%	1.12
50	82% ± 3%	77% ± 2%	85% ± 1%	81% ± 2%	1.31
60	**75% ± 4%**	**76% ± 3%**	**83% ± 3%**	**78% ± 3%**	**1.52**
70	35% ± 1%	40% ± 18%	49% ± 2%	41% ± 17%	0.56
80	22% ± 3%	28% ± 27%	27% ± 3%	26% ± 30%	0.34
90	5% ± 1%	7% ± 4%	11% ± 7%	8% ± 4%	0.06

Dropped	**mIoU**				$\psi_{\mathbf{PFR}}$ (Mean)
Labels [%]	Atrium	Bulbus	Ventricle	Mean	
0	**73% ± 2%**	**64 ± 3%**	**78% ± 2%**	**71% ± 6%**	**0.50**
10	73% ± 4%	65% ± 3%	77% ± 2%	72% ± 3%	0.58
20	71% ± 2%	65% ± 1%	77% ± 1%	71% ± 1%	0.63
30	69% ± 3%	64% ± 2%	74% ± 2%	69% ± 2%	0.68
40	71% ± 2%	65% ± 4%	76% ± 2%	71% ± 3%	0.84
50	71% ± 4%	63% ± 2%	74% ± 1%	70% ± 2%	0.98
60	**66% ± 0.04**	**61% ± 4%**	**71% ± 5%**	**66% ± 4%**	**1.09**
70	27% ± 13%	30% ± 15%	38% ± 20%	32% ± 16%	0.34
80	17% ± 24%	20% ± 21%	21% ± 29%	20% ± 25%	0.20
90	4% ± 1%	4% ± 2%	6% ± 3%	5% ± 2%	0.03

Table A.3: Results of the ablation study, where labels are systematically removed from all classes. The results are organized based on the evaluation metric employed and the proportion of labels dropped. Each value reported represents an average derived from ten runs with the corresponding standard deviation (±). The baseline and the optimal ψ_{PFR} are marked in bold.

Dropped	**mDSC**			$\psi_{\mathbf{PFR}}$ (Atrium)
Labels [%]	Ventricle	Atrium	Mean	
10	**89% ± 2%**	**87% ± 4%**	**88% ± 3%**	**0.84**
20	88% ± 2%	85% ± 3%	87% ± 3%	0.90
30	88% ± 1%	85% ± 7%	87% ± 2%	1.03
40	87% ± 3%	85% ± 5%	86% ± 4%	1.20
50	88% ± 2%	82% ± 4%	85% ± 3%	1.34
60	87% ± 3%	85% ± 5%	86% ± 4%	1.81
70	**87% ± 2%**	**81% ± 15%**	**84% ± 9%**	**2.19**
80	82% ± 12%	43% ± 8%	62% ± 10%	0.92
90	85% ± 7%	41% ± 6%	63% ± 7%	1.68

Dropped	**mIoU**			$\psi_{\mathbf{PFR}}$ (Atrium)
Labels [%]	Ventricle	Atrium	Mean	
10	**79% ± 3%**	**77% ± 4%**	**77% ± 7%**	**0.66**
20	78% ± 3%	73% ± 4%	76% ± 4%	0.67
30	78% ± 4%	74% ± 7%	76% ± 6%	0.78
40	78% ± 4%	74% ± 4%	76% ± 4%	0.91
50	79% ± 4%	69% ± 4%	74% ± 4%	0.95
60	77% ± 3%	74% ± 6%	76% ± 5%	1.37
70	**77% ± 4%**	**68% ± 4%**	**73% ± 4%**	**1.54**
80	75% ± 13%	37% ± 8%	56% ± 11%	0.68
90	76% ± 6%	33% ± 5%	55% ± 6%	1.09

Table A.4: Results of the transfer learning experiment, which extends the dataset with an additional class. The results are organized based on the quantity of discarded atrium label masks with the mIoU and mDSC metrics reported. All values are the average of ten successive iterations, with their respective standard deviations (±). The baseline and the highest value for the ψ_{PFR} are given in bold.

Dataset Size Ratio		**mDSC**	
Atrium [%] / Ventricle [%]	Ventricle	Atrium	Mean
25.0 / 75.0	63% ± 8%	78% ± 2%	71% ± 5%
37.5 / 62.5	78% ± 3%	78% ± 5%	78% ± 4%
50.0 / 50.0	**73% ± 5%**	**74% ± 4%**	**74% ± 5%**
62.5 / 37.5	**78% ± 9%**	**83% ± 3%**	**81% ± 6%**
75.0 / 25.0	78% ± 3%	78% ± 6%	78% ± 5%

Dataset Size Ratio		**mIoU**	
Atrium [%] / Ventricle [%]	Ventricle	Atrium	Mean
25.0 / 75.0	52% ± 2%	67% ± 5%	60% ± 4%
37.5 / 62.5	70% ± 4%	70% ± 3%	70% ± 4%
50.0 / 50.0	**72% ± 8%**	**73% ± 3%**	**73% ± 5%**
62.5 / 37.5	**68% ± 2%**	**75% ± 4%**	**72% ± 3%**
75.0 / 25.0	68% ± 4%	67% ± 9%	68% ± 6%

Table A.5: The evaluation results of the experiment exploring transfer learning through merging two datasets. The results are organized based on the proportion between the two datasets. For instance, a ratio of 25.0% / 75.0% signifies that 25% of the total samples are labeled as atrium class, while 75% is the ventricle class. All values are average across ten successive runs, with the standard deviation provided (±). The baseline condition (50% / 50% label distribution) and the dataset size ratio that yielded optimal performance are emphasized in bold.

α	mDSC		
	Ventricle	Atrium	Mean
0.0	**86% ± 2%**	**84% ± 1%**	**85% ± 1%**
0.1	85% ± 1%	86% ± 1%	85% ± 1%
0.2	85% ± 2%	83% ± 4%	84% ± 3%
0.3	87% ± 1%	85% ± 1%	86% ± 1%
0.4	**86% ± 0%**	**87% ± 1%**	**87% ± 1%**
0.5	85% ± 3%	87% ± 0%	86% ± 3%
0.6	82% ± 4%	88% ± 1%	85% ± 3%
0.7	83% ± 1%	88% ± 0%	85% ± 2%
0.8	75% ± 2%	85% ± 1%	80% ± 1%
0.9	0% ± 1%	39% ± 10%	19% ± 20%

α	mIoU		
	Ventricle	Atrium	Mean
0.0	**75% ± 4%**	**73% ± 1%**	**74% ± 3%**
0.1	75% ± 2%	75% ± 1%	75% ± 2%
0.2	74% ± 3%	71% ± 6%	72% ± 5%
0.3	77% ± 2%	74% ± 1%	76% ± 3%
0.4	**76% ± 0%**	**77% ± 1%**	**77% ± 1%**
0.5	74% ± 5%	78% ± 0%	76% ± 4%
0.6	69% ± 1%	78% ± 1%	74% ± 4%
0.7	71% ± 1%	78% ± 1%	75% ± 3%
0.8	60% ± 3%	74% ± 4%	67% ± 8%
0.9	0% ± 1%	24% ± 8%	12% ± 13%

Table A.6: Evaluation results of the experiment that explores the trade-off between horizontal and vertical loss. These results have been organized based on the mIoU and mDSC metrics and then over the weighting factor α (see Equation (5.6)). All values are averages from ten runs, given with the standard deviations (±). The baseline performance (using only Dice loss) and the optimal performance are marked in bold.

A.5.3 Detailed CoHeG Framework Results

a) Droplet-DS

	Otsu	U-Net (Otsu)	k-means	U-Net (k-means)	U-Net (Human)
DSC	34%	76% ± 1%	37%	71% ± 1%	93% ± 1%
IoU	21%	62% ± 0%	23%	56% ± 1%	86% ± 2%
PA	98%	99% ± 0%	98%	99% ± 0%	99% ± 0%

b) ISIC-DS

	Otsu	U-Net (Otsu)	k-means	U-Net (k-means)	U-Net (Human)
DSC	58%	58% ± 3%	54%	64% ± 3%	77% ± 1%
IoU	42%	45% ± 3%	38%	48% ± 4%	63% ± 2%
PA	84%	84% ± 0%	83%	87% ± 0%	93% ± 0%

Table A.7: The evaluation results of segmentation mask generation using the CoHeG framework concerning various metrics, using ground-truth masks as targets. All metrics are given as percentage values. U-Net (*X*) denotes U-Net trained with annotations from method *X*. Tables a) and b) show results for the Droplet-DS and the ISIC-DS, respectively. The standard deviation over four runs with different random seeds is given as ±. The best performance possible is given as U-Net (Human), which is trained with ground truth annotations.

A.6 Training Details for Deep Learning With Unannotated Data

A.6.1 Self-SL Comparison Framework

A.6.1.1 Preprocessing, Augmentations, and Hyperparameters

The samples in the MoNu-DS are relatively large (1000 × 1000 pixels). To maintain content integrity while ensuring processability, each image is subdivided into multiple 256×256 pixel crops. For the ISIC-DS, a resizing operation to 256×256 pixels is performed on the samples.

In the context of CL, appropriate image augmentations are essential. For the ISIC-DS, augmentations commonly utilized in most Self-SL frameworks are selected [Wan21b]. However, alternative augmentations are necessary due to the histopathological nature of the MoNu-DS. As a standardized approach for this data type does not exist, an extensive hyperparameter search identifies a set of suitable augmentation parameters. For all augmentations, excluding *Gaussian blurring*, the *Range* value is the relative percentage change ranges. In the case of *Gaussian blurring*, the *Range* value denotes the standard deviation. The probability of application for each augmentation is specified by P. The following augmentations are found to be optimal:

- Random Cropping. Range $[0.2,\ 1.0]$ and P=100%.
- Brightness modifications. Range: $[0.4,\ 1.6]$ and P=80%.
- Contrast modifications. Range: $[0.2,\ 1.8]$ and P=80%.
- Saturation modifications. Range: $[0.2,\ 1.8]$ and P=80%.
- Brightness modifications. Range: $[0.8,\ 1.2]$ and P=80%.
- Gaussian blurring. Range: $[0.1,\ 2.0]$ and P=80%.
- Horizontal and vertical flipping. P=50%.

For all CL pretext tasks, training is conducted using the Stochastic Gradient Descent (SGD) optimizer. The parameters are set as follows: weight decay of 1x10^{-4}, momentum of 0.9, and learning rate of 1×10^{-3}. Additionally, Cosine Annealing [Los16] is applied to the learning rate. In the case of semantic segmentation, the Dice Loss [Sud17] is employed, while, for instance segmentation, the Smooth L1 Loss [Gir15a] is utilized. Both losses are combined with the Adam [Kin14] optimizer, using a learning rate of 1×10^{-3}. For instance segmentation, a subsequent seed-based watershed post-processing step is implemented [Sch20].

A.6.2 Detailed Centered Kernel Alignment Values

a) ISIC-DS

Method	Conv1↑	Conv2↑	Conv3↑	Conv4↑	μ↑
ImageNet	**0.88**	**0.86**	0.75	0.65	0.79
Autoencoder	0.68	0.51	0.45	0.37	0.50
BYOL [Gri20]	0.83	0.82	0.73	0.60	0.75
DenseCL [Wan21b]	0.85	0.82	**0.81**	**0.73**	**0.80**
DetCo [Xie21]	0.65	0.46	0.40	0.36	0.47
MoCo [He20]	0.77	0.70	0.74	0.70	0.73
SimCLR [Che20a]	0.85	0.82	0.79	0.71	0.79
Barlow Twins [Zbo21]	0.72	0.61	0.66	0.70	0.67

b) MoNu-DS

Method	Conv1↑	Conv2↑	Conv3↑	Conv4↑	μ↑
ImageNet	0.95	0.78	0.45	**0.48**	**0.67**
Autoencoder	0.96	0.74	0.39	0.35	0.61
BYOL [Gri20]	0.91	0.65	0.28	0.25	0.52
DenseCL [Wan21b]	**0.97**	**0.79**	**0.47**	0.46	**0.67**
DetCo [Xie21]	0.62	0.31	0.11	0.13	0.29
MoCo [He20]	0.96	0.76	0.45	0.43	0.65
SimCLR [Che20a]	0.95	0.76	0.46	0.44	0.65
Barlow Twins [Zbo21]	0.76	0.61	0.45	0.45	0.57

Table A.8: CKA comparing Self-SL methods and SL. *Conv1-Conv4* represents the CKA similarity for the respective layer of the ResNet-50 [He16] encoder. The mean value across all layers is denoted by μ.

A.6.3 Detailed Integrated Quality Criterion Values

a) ISIC-DS

Method	Frozen	Unfrozen
Random	86 (-)	92 (-)
ImageNet	93 (0.000)	95 (0.000)
Autoencoder	90 (0.000)	93 (0.000)
BYOL [Gri20]	93 (0.000)	95 (0.000)
DenseCL [Wan21b]	**94** (0.000)	**96** (0.000)
DetCo [Xie21]	80 (0.999)	90 (0.991)
MoCo [He20]	93 (0.000)	96 (0.000)
SimCLR [Che20a]	94 (0.000)	96 (0.000)
Barlow Twins [Zbo21]	92 (0.000)	93 (0.019)

b) MoNu-DS

Method	Frozen	Unfrozen
Random	81 (-)	90 (-)
ImageNet	**90.5** (0.000)	93 (0.021)
Autoencoder	80 (0.76)	90 (0.000)
BYOL [Gri20]	83 (0.048)	91 (0.410)
DenseCL [Wan21b]	87 (0.001)	**94** (0.003)
DetCo [Xie21]	78 (0.997)	85 (0.992)
MoCo [He20]	88 (0.000)	93 (0.003)
SimCLR [Che20a]	86 (0.002)	93 (0.016)
Barlow Twins [Zbo21]	80 (0.869)	87 (0.999)

Table A.9: The ψ_{IQC} as a quantitative comparison of Self-SL methods in the context of segmentation tasks for both the ISIC-DS and MoNu-DS. The metric is calculated with the DSC (ISIC-DS) or the AJI^{+} (MoNu-DS). The p values of the t-test are given in brackets. It is discriminated between *Frozen* and *Unfrozen* encoders. Performance metrics are provided in percentages.

A.6.4 Self-SL Optimization Framework

A.6.4.1 ConvNeXtV2 Model Sizes

		#Parameters	
Name	**Short Name**	**Backbone**	**ConvNeXtV2**
ConvNeXt Atto	CNXT-A	3.40×10^6	3.70×10^6
ConvNeXt Femto	CNXT-F	4.90×10^6	5.20×10^6
ConvNeXt Pico	CNXT-P	8.60×10^6	9.10×10^6
ConvNeXt Nano	CNXT-N	1.50×10^7	1.60×10^7
ConvNeXt Tiny	CNXT-T	2.79×10^7	2.86×10^7
ConvNeXt Base	CNXT-B	8.80×10^7	8.90×10^7
ConvNeXt Large	CNXT-L	1.96×10^8	1.98×10^8
ConvNeXt Huge	CNXT-H	6.57×10^8	6.60×10^8

Table A.10: All ConvNeXt backbones and ConvNeXtV2 model sizes. *Name* is the full name of the respective backbone, *Short Name* the abbreviation used in this work, and *#Parameters* the number of parameters for the *bacbone* and the whole ConvNeXtV2 model respective.

A.6.4.2 Training Details

For Self-SL pretraining, the A-SEM-DS is divided into an 80% training set and a 20% evaluation set, with no test split being utilized during this phase. For the implementation of DenseCL, the configuration recommended by the method's authors is adopted. This includes a learning rate of 0.3, linear scaling with batch size, step-wise decay scheduling, and linear warm-up, in conjunction with SGD optimization and mixed-precision training [Wan21b]. The data augmentation techniques applied for DenseCL are those determined to be optimal before [Ret23b]. For ConvNeXtV2, the creators' suggestions are followed, employing a cosine decay learning rate schedule with a base rate of 0.0008, an AdamW [Los17] optimizer with weight decay and layer-wise learning rate decay [Woo23]. Augmentations are modified for SEM image analysis. To ensure the visibility of fine-grained details and small particles, images are not scaled but randomly cropped to 640×640 pixels and horizontally flipped with a 50% probability. The patch size for the Masked-AE masking is set at 32×32 pixels. A prior hyperparameter study optimized the number of removed patches to 50%.

Both Self-SL methods are trained for 1,500 epochs without early stopping. The epoch with the lowest validation loss is subsequently used for downstream training. ImageNet weights are obtained from the official ConvNeXtV2 GitHub page[3]. Downstream training is conducted uniformly across all methods. The Mask R-CNN implementation and configuration found to be optimal is utilized [Ret24c], although the uncertainty head is not employed. To ensure reliable metrics, the downstream experiments are repeated 10 times. The Self-SL pretraining is performed on the HoreKa supercomputer using distributed training across 20 nodes, each equipped with 4 GPUs, resulting in a total of 80 GPUs.

[3] https://github.com/facebookresearch/ConvNeXt-V2

A.6.5 Detailed Masked Autoencoders Results

Table A.11 lists the detailed results, and Table A.12 shows the full tabulated results of the ablation study for the model comparison in Section 6.4.2.

a) Low Magnification

Backbone	**None**	**DenseCL**	**ImageNet**	**ConvNeXtV2**
CNXT-A	$80\% \pm 3\%$	$82\% \pm 2\%$	$85\% \pm 1\%$	$87\% \pm 0\%$
CNXT-F	$83\% \pm 1\%$	$83\% \pm 1\%$	$86\% \pm 1\%$	$87\% \pm 1\%$
CNXT-P	$83\% \pm 1\%$	$84\% \pm 1\%$	$86\% \pm 1\%$	$87\% \pm 0\%$
CNXT-N	$84\% \pm 1\%$	$85\% \pm 1\%$	$86\% \pm 1\%$	$87\% \pm 0\%$
CNXT-T	$84\% \pm 1\%$	$85\% \pm 1\%$	$86\% \pm 1\%$	$87\% \pm 0\%$
CNXT-B	$83\% \pm 1\%$	$85\% \pm 1\%$	$86\% \pm 1\%$	$86\% \pm 1\%$
CNXT-L	$84\% \pm 1\%$	$85\% \pm 1\%$	$86\% \pm 1\%$	$87\% \pm 1\%$
CNXT-H	$83\% \pm 1\%$	$84\% \pm 1\%$	$85\% \pm 1\%$	$86\% \pm 1\%$

b) High Magnification

Backbone	**None**	**DenseCL**	**ImageNet**	**ConvNeXtV2**
CNXT-A	$44\% \pm 6\%$	$45\% \pm 6\%$	$61\% \pm 2\%$	$62\% \pm 4\%$
CNXT-F	$42\% \pm 6\%$	$45\% \pm 6\%$	$60\% \pm 4\%$	$65\% \pm 2\%$
CNXT-P	$42\% \pm 9\%$	$48\% \pm 7\%$	$60\% \pm 4\%$	$65\% \pm 2\%$
CNXT-N	$48\% \pm 5\%$	$51\% \pm 3\%$	$58\% \pm 3\%$	$65\% \pm 1\%$
CNXT-T	$48\% \pm 8\%$	$51\% \pm 5\%$	$60\% \pm 2\%$	$65\% \pm 3\%$
CNXT-B	$50\% \pm 6\%$	$53\% \pm 5\%$	$61\% \pm 3\%$	$64\% \pm 2\%$
CNXT-L	$46\% \pm 9\%$	$53\% \pm 6\%$	$61\% \pm 6\%$	$63\% \pm 2\%$
CNXT-H	$47\% \pm 10\%$	$53\% \pm 4\%$	$61\% \pm 6\%$	$63\% \pm 2\%$

Table A.11: Comparison of random initialization of the ConvNeXt backbone (*None*), and pretraining with *DenseCL*, *ImageNet*, and the *ConvNeXtV2* approach. The methods are evaluated with the AJI$^+$ metric and all scales of the ConvNeXt backbone for the high and low magnification dataset splits of the SEM-DS. *DenseCL* and *ConvNeXtV2*: A-SEM-DS pretraining; *ImageNet*: transfer learning; *None*: random initialization. All configurations are repeated 10x with random seeds; the standard deviation is listed along the AJI$^+$.

Dataset Size	Low Magnification	High Magnification
1%	$85\% \pm 0\%$	$54\% \pm 7\%$
2%	$86\% \pm 0\%$	$56\% \pm 6\%$
4%	$86\% \pm 0\%$	$60\% \pm 4\%$
8%	$86\% \pm 0\%$	$61\% \pm 3\%$
16%	$86\% \pm 1\%$	$64\% \pm 2\%$
32%	$87\% \pm 0\%$	$65\% \pm 1\%$
64%	$87\% \pm 0\%$	$65\% \pm 1\%$
100%	$87\% \pm 0\%$	$66\% \pm 2\%$

Table A.12: Comparison of various dataset sizes relevant to the ConvNeXt CNXT-P (Pico) backbone. Each experiment is conducted 10 times using different random seeds, and the standard deviation is provided alongside the AJI^{+}.

A.7 Training Details for Selection and Refinement of Neural Networks

A.7.1 Architecture Selection

The Dice Loss [Sud17] is selected as the objective function for the architecture selection. The Adam optimizer [Kin14], a popular choice for deep learning tasks, was employed with learning rates of 0.002 for the U-Net model and 0.0001 for the Mask R-CNN architecture. Data augmentation techniques were deliberately omitted from the training process. The Cyto-DS dataset is partitioned, with 80% of the samples allocated for training purposes and the remaining 20% reserved for testing. This split is implemented randomly across all sub-datasets to ensure unbiased evaluation. Before processing, the samples are scaled to ensure all values fall between 0 and 1, thereby improving model convergence. Early stopping mechanisms and learning rate scheduling are incorporated into the training to mitigate overfitting and optimize training efficiency. Each experiment is repeated four times to account for the stochastic nature of neural network training and ensure the robustness of the reported results. Different random seeds are used for each repetition to initialize the network parameters. The final results are presented as the mean values across these four runs with their corresponding standard deviations, providing a measure of the variability in performance. For evaluation, a test split of the corresponding sub-dataset is used. Further details regarding the NN architectures are given in Section 3.3.3.

A.7.2 Uncertainty Estimation

For the uncertainty evaluation, multiple hyperparameter searches were conducted. The original U-Net architecture is enhanced by a ResNet-50 [He16] backbone, allowing for the same encoder to be used for both models, thereby increasing comparability. Hence, the U-Net implementation comprises 38.5 million in trainable parameters, and the Mask R-CNN model encompasses 44.0 million, of which $6,150$ is attributed to the novel uncertainty head. For the optimization process, different approaches are adopted. The U-Net model is optimized using Adam citekingma2014adam, with a learning rate set at 0.005. Conversely, the Mask R-CNN is optimized using AdamW [Los17], employing a learning rate of 0.0001. A normalization procedure is applied to all samples, constraining them within the range of 0-1. Early stopping and learning rate scheduling techniques are employed to prevent overfitting and enhance model performance. All experiments are repeated four times to ensure the reliability of the metrics and mitigate initialization effects. The SEM scans are not resized during training to preserve all identifiable details in the images. Due to the substantial image size, a mini-batch size of 1 is employed for both the U-Net and Mask R-CNN models. For the evaluation regarding uncertainty estimation, the ground truth bounding boxes are provided to the Mask R-CNN. This approach is used to evaluate the confidence predictions exclusively. Given that the model outputs confidence as a decimal value, a threshold is established. Predictions with confidence levels exceeding 50% are classified as certain, while those with confidence levels of 50% or below are assigned as being uncertain. Each dataset is split into 80% train- and 20% test-data. All experiments are evaluated on the test split.

A.7.2.1 Network Configurations

Bayesian optimization[4] search for 24 hours is conducted for both NNs hyperparameters to estimate and fine-tune the optimal configurations for instance segmentation of SEM images in high- and low magnifications.

U-Net: Table A.13 lists the details regarding the search for the parameters $\mathbf{P}_{\text{cell}}$ and $\mathbf{P}_{\text{seed}}$ for low and high magnifications. For details regarding the parameters, see Section 3.3.3. The hyperparameter optimization reveals that $\mathbf{P}_{\text{cell}}$ values differ drastically between magnifications, with a lower threshold (0.04) needed for high magnification images compared to low magnification (0.12), indicating more precise cell boundary detection is required at higher resolutions. In contrast, $\mathbf{P}_{\text{seed}}$ values remain relatively consistent (0.37 for low and 0.34 for high magnification), suggesting that seed extraction is more robust across different magnifications.

Magnification	**Parameter**	**Minimum**	**Maximum**	**Optimal**
Low	$\mathbf{P}_{\text{cell}}$	0.01	0.30	**0.12**
	$\mathbf{P}_{\text{seed}}$	0.10	0.99	**0.37**
High	$\mathbf{P}_{\text{cell}}$	0.01	0.30	**0.04**
	$\mathbf{P}_{\text{seed}}$	0.10	0.99	**0.34**

Table A.13: Results for the hyperparameter search regarding the U-Net network configuration. The validation loss determines the optimal value.

Mask R-CNN: For the Mask R-CNN network, the optimal values for bin size (k), noise on ground-truth labels (σ), similarity enforcement factor (ξ), and all post-processing parameters for both magnifications are determined and listed in

[4] Bayesian hyperparameter optimization is a probabilistic model-based approach that builds a surrogate model of the objective function and uses it to select the most promising hyperparameters for evaluation [Sno12].

Table A.14. Details regarding the parameters are given in Section 7.3.2. The hyperparameter search revealed that a medium-sized bin size of 6 works best for both magnifications. Interestingly, both magnification levels performed better with slight noise in the ground truth labels (σ). However, a noticeable difference can be observed in how the enforcement is affected between the two magnifications. High magnification benefits from minimal enforced similarity, while low magnification requires stronger enforcement through ξ. This distinction aligns with the characteristics of the magnification levels, where low magnification provides clearer samples, and high magnification includes blurrier and less accurate ones. Regarding the post-processing parameters, it was found that certain trends were consistent for both magnification levels, as depicted in. Notably, the parameter θ_Σ exhibited a very small value for high magnification, possibly indicating higher uncertainty in these images. Additionally, it can be observed that the threshold values, including θ_ω, θ_ψ, and θ_η, were generally smaller for high magnification, likely due to the complexity and uncertainty inherent in those SEM scans.

An automated approach is employed to derive bounding boxes from the hand-labeled segmentation masks since the Mask R-CNN architecture requires bounding boxes. The approach ensures a close fit around the particle edges. This close fit, while enhancing precision, can occasionally pose challenges for accurate segmentation. Therefore, an additional investigation was conducted on whether it would be advantageous to enlarge the boxes slightly with padding. This search identifies an optimal 33-pixel padding, as shown in Figure A.2. Examples of different bounding box sizes are shown in Figure A.3.

a) Pre-Processing

Magnification	**Parameter**	**Minimum**	**Maximum**	**Optimal**
Low	**k**	2	10	**6**
	σ	0.000	1.100	**0.245**
	ξ	0.001	3.000	**0.412**
High	**k**	2	10	**6**
	σ	0.000	1.100	**0.189**
	ξ	0.001	3.000	**2.127**

b) Post-Processing

Magnification	**Parameter**	**Minimum**	**Maximum**	**Optimal**
Low	θ_ω	0.00	0.99	**0.75**
	θ_ψ	0.00	0.99	**0.69**
	$\theta_\cap$	0.00	0.80	**0.70**
	θ_η	0.00	0.80	**0.49**
	θ_Σ	0.00	0.80	**0.41**
High	θ_ω	0.00	0.99	**0.71**
	θ_ψ	0.00	0.99	**0.51**
	$\theta_\cap$	0.00	0.80	**0.46**
	θ_η	0.00	0.80	**0.48**
	θ_Σ	0.00	0.80	**0.14**

Table A.14: Results for the hyperparameter search regarding the Mask R-CNN network configuration, split by pre- and post-processing. The validation loss determines the optimal value.

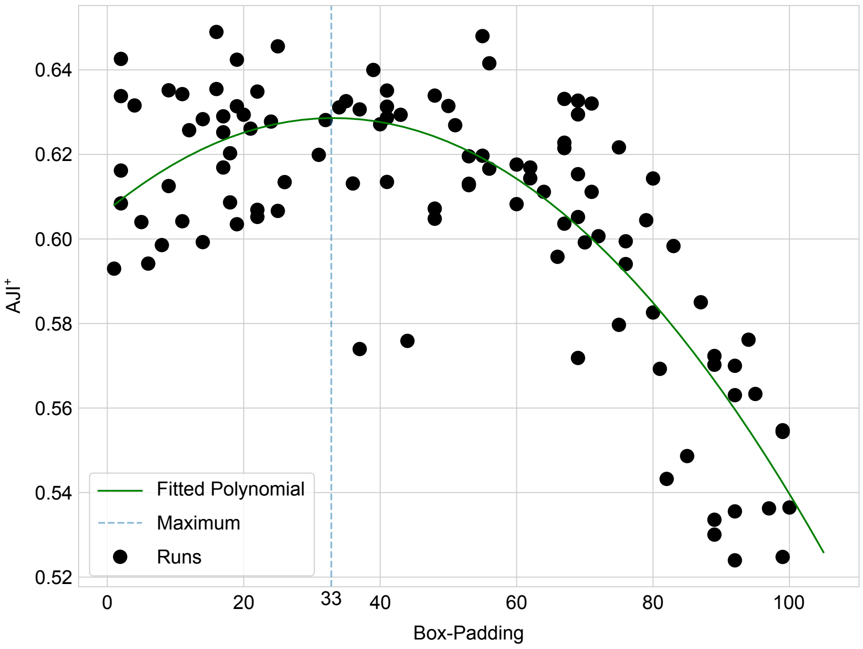


Figure A.2: The average AJI^{+} for each run plotted against the paddings. A polynomial is fitted over the runs, and the maximum of 33 pixels is determined as the optimal padding value. Adapted from [Ret24c].

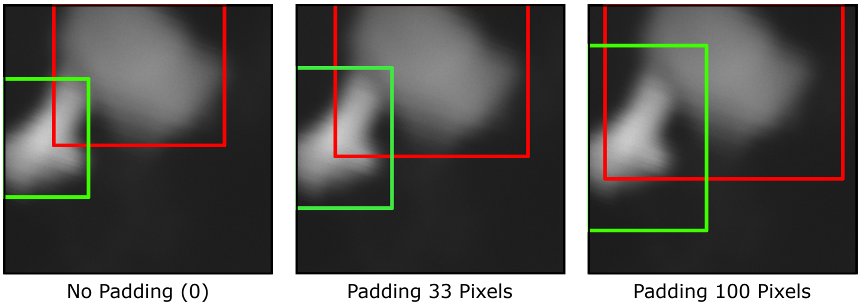


Figure A.3: Three examples of paddings. On the left, there is no padding. On the right, the maximum of 100 pixels, and in the middle, the found optimal value of 33. Adapted from [Ret24c].

A.7.2.2 SEM-DS Dataset details

During image acquisition, various sample locations with diverse particle arrangements were sampled for the SEM-DS. These include well-separated particles as well as agglomerated particle clusters. A range of particle sizes is also spanned. The particle counts for the training data are outlined in Table A.15, and the size distributions are depicted in Figure A.4.

Dataset	Certain Particles	Uncertain Particles
High Magnification	153	221
Low Magnification	260	2514

Table A.15: Number of certain and uncertain particles in the high and low magnification split of the SEM-DS.

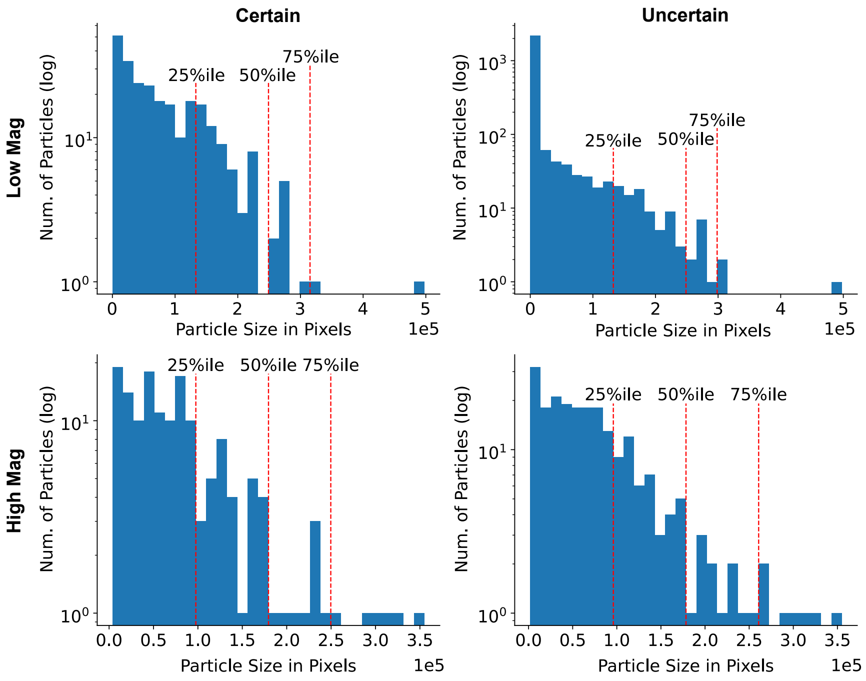


Figure A.4: Histograms for the particle size distributions in pixels for the SEM-DS, split by certain and uncertain labels. Red dashed lines denote percentile values. Adapted from [Ret24c].

List of Figures

List of Tables

Acronyms

DL	Deep Learning	1
ML	Machine Learning	1
NN	Neural Network	30
SEM	Scanning Electron Microscopy	2
IoU	Intersection-Over-Union	41
TEM	Transmission Electron Microscopy	120
CNN	Convolutional Neural Network	31
AJI$^+$	Aggregated Jaccard Index	42
FCN	Fully Convolutional Network	33
EDN	Encoder-Decoder Network	33
ResNet	Residual Neural Network	36
ROI	Region of Interest	38
RPN	Region Proposal Network	38
FPN	Feature Pyramid Network	38
EDF	Extended Depth of Field	47
PCC	Pearson Correlation Coefficient	42
AN	Artificial Neuron	30
ANN	Artificial Neural Network	31
FFN	Feed Forward Network	31
RNN	Recurrent Neural Network	31
SL	Supervised Learning	20
UL	Unsupervised Learning	21
AE	Autoencoder	22
Semi-SL	Semi-Supervised Learning	22
Self-SL	Self-Supervised Learning	22
DSC	Dice-Sørensen Coefficient	41

PCC	Pearson Correlation Coefficient	42
CKA	Centered Kernel Alignment	43
CAM	Class Activation Map	44
NNC	Nearest Neighbor Comparison	44
GAN	Generative Adversarial Network	35
CL	Contrastive Learning	35
MLP	Multilayer Perceptron	87
BYOL	Bootstrap Your Own Latent	87
ViT	Vision Transformer	34
GRN	Global Response Normalization	40
PCA	Principal Component Analysis	90
ERR	Error Reduction Rate	43
GRN	Global Response Normalization	40
MLP	Multi-Layer Perceptron	87
PA	Pixel Accuracy	40
ViT	Vision Transformer	34
ReLU	Rectified Linear Unit	39
GELU	Gaussian Error Linear Unit	39
Masked-AE	Masked Autoencoder	35
FCMAE	Fully Convolutional Masked Autoencoder	40
GRN	Global Response Normalization	40
PFR	Performance Frugality Ratio	60
CAL	Class Asymmetric Loss	60
HoreKa	Hochleistungsrechner Karlsruhe	154
HAICORE	Helmholtz AI Computing Resources	154
SLURM	Simple Linux Utility for Resource Management	154
SGD	Stochastic Gradient Descent	168
MoCo	Momentum Contrast	87
SimCLR	Simple Contrastive Learning	87
DenseCL	Dense Contrastive Learning	87
DetCo	Detection Contrast	87
NeuBoot	Neural Bootstrapping Framework	60
CoHeG	Controlled Heterogeneity Generator	60

SelfSLBench	Self-SL Benchmarking Framework	89
SelfSLAutoTune	Self-SL Automatic Tuning Framework	89
ArchSelect	Architecture Selection Framework	122
U-Mask R-CNN	Uncertainty-Mask R-CNN	122
$\psi_{\mathbf{PFR}}$	Performance Frugality Ratio	60
$\psi_{\mathbf{NQC}}$	Neighborhood Quality Criterion	90
$\psi_{\mathbf{RQC}}$	Runtime Quality Criterion	91
$\psi_{\mathbf{IQC}}$	Integrated Quality Criterion	91
$\psi_{\mathbf{CFM}}$	Class Frequency Metric	124
$\psi_{\mathbf{GOC}}$	General Overlap Criterion	124
$\psi_{\mathbf{OFC}}$	Overlap Frequency Criterion	125
Cyto-DS	Synthetic Cervical Cytology Dataset	47
Heart-DS	Medaka Fish Heart Segmentation Dataset	47
Droplet-DS	Droplet Microarray Dataset	47
ISIC-DS	Intl. Skin Imaging Collaboration Melanoma Dataset	48
MoNu-DS	Multi-Organ Nucleus Segmentation Dataset	48
SEM-DS	Powder Uncertainty Scanning Electron Microscopy Dataset	52
LGS-DS	$LiCoO_2$ Grid SEM Dataset	55
A-SEM-DS	Automated Scanning Electron Microscopy Scans Dataset	57

Own Publications

[Sch22d] M. Schutera, L. Rettenberger, C. Pylatiuk, and M. Reischl. Methods for the frugal labeler: Multi-class semantic segmentation on heterogeneous labels. *PLoS ONE*, volume 17, page e0263656, Public Library of Science, 2022. DOI: 10.1371/journal.pone.0263656.

[Ret22] L. Rettenberger, M. Schilling, and M. Reischl. Annotation efforts in image segmentation can be reduced by neural network bootstrapping. In *Current Directions in Biomedical Engineering*, volume 8, pages 329–332, Innsbruck, Austria, 2022. De Gruyter. DOI: 10.1515/cdbme-2022-1084.

[Ret23b] L. Rettenberger, M. Schilling, S. Elser, M. Böhland, and M. Reischl. Self-supervised learning for annotation efficient biomedical image segmentation. *IEEE Transactions on Biomedical Engineering*, IEEE, 2023. DOI: 10.1109/TBME.2023.3252889.

[Ret23a] L. Rettenberger, F. R. Münke, R. Bruch, and M. Reischl. Mask R-CNN outperforms U-Net in instance segmentation for overlapping cells. In *Current Directions in Biomedical Engineering*, volume 9, pages 335–338, Duisburg, Germany, 2023. De Gruyter. DOI: 10.1515/cdbme-2023-1084.

[Ret24a] L. Rettenberger, N. Beyer, I. Sieber, and M. Reischl. Fault detection in 3D-printing with deep learning. In *Proceedings of the IEEE ICCE*, pages 1–4, Las Vegas, USA, 2024. IEEE. DOI: 10.1109/ICCE59016.2024.10444198.

[Ret24c] L. Rettenberger, N. J. Szymanski, Y. Zeng, J. Schützke, S. Wang, G. Ceder, and M. Reischl. Uncertainty-aware particle segmentation for electron microscopy at varied length scales. *npj Computational Materials*, volume 10, page 124, Nature Publishing Group UK London, 2024. DOI: 10.1038/s41524-024-01302-w.

[Ret24b] L. Rettenberger, M. F. Münker, M. Schutera, C. M. Niemeyer, K. S. Rabe, and M. Reischl. Using large language models for extracting structured information from scientific texts. In *Current Directions in Biomedical Engineering*, volume 10, pages 526–529, Stuttgart, Germany, 2024. De Gruyter. DOI: 10.1515/cdbme-2024-2129.

[Ret25a] L. Rettenberger, M. Reischl, and M. Schutera. Assessing political bias in large language models. *Journal of Computational Social Science*, volume 8, pages 1–17, Springer, 2025. DOI: 10.1007/s42001-025-00376-w.

[Ret25b] L. Rettenberger, N. J. Szymanski, A. Giunto, O. Dartsi, A. Jain, G. Ceder, V. Hagenmeyer, and M. Reischl. Leveraging unlabeled SEM datasets with self-supervised learning for enhanced particle segmentation. *npj Computational Materials*, volume 11, page 289, Nature Publishing Group UK London, 2025. DOI: 10.1038/s41524-025-01802-3.

[Sch21a] M. P. Schilling, L. Rettenberger, F. Münke, H. Cui, A. A. Popova, P. A. Levkin, R. Mikut, and M. Reischl. Label assistant: a workflow for assisted data annotation in image segmentation tasks. In *Proceedings—31. Workshop Computational Intelligence*, pages 211–234, Berlin, Germany, 2021. DOI: 10.58895/ksp/1000138532-14.

[Sch22a] M. P. Schilling, N. Ahuja, L. Rettenberger, T. Scherr, and M. Reischl. Impact of annotation noise on histopathology nucleus segmentation. In *Current Directions in Biomedical Engineering*, volume 8, pages 197–200, Innsbruck, Austria, 2022. De Gruyter. DOI: 10.1515/cdbme-2022-1051.

[Sch22e] M. Schutera, L. Rettenberger, and M. Reischl. Automated zebrafish phenotype pattern recognition: 6 years ago, and now. *Zebrafish*, volume 19, pages 213–217, Mary Ann Liebert, Inc., 2022. DOI: 10.1089/zeb.2022.0027.

[Böh23] M. Böhland, R. Bruch, S. Bäuerle, L. Rettenberger, and M. Reischl. Improving generative adversarial networks for patch-based unpaired image-to-image translation. *IEEE Access*, volume 11, pages 127895–127906, IEEE, 2023. DOI: 10.1109/ACCESS.2023.3331819.

[Bor23] P. Borlinghaus, F. Tausch, and L. Rettenberger. A purely visual re-id approach for bumblebees (bombus terrestris). *Smart Agricultural Technology*, volume 3, page 100135, Elsevier, 2023. DOI: 10.1016/j.atech.2022.100135.

[RM23] F. R. Münke, L. Rettenberger, A. Popova, and M. Reischl. A lightweight framework for semantic segmentation of biomedical images. In *Current Directions in Biomedical Engineering*, volume 9, pages 190–193, Duisburg, Germany, 2023. De Gruyter. DOI: 10.1515/cdbme-2023-1048.

[Wüh24] L. Wührl, L. Rettenberger, R. Meier, E. Hartop, J. Graf, and C. Pylatiuk. Entomoscope: An open-source photomicroscope for biodiversity discovery. *IEEE Access*, IEEE, 2024. DOI: 10.1109/ACCESS.2024.3355272.

[Pol24] M. Polomoshnov, K.-M. Reichert, L. Rettenberger, M. Ungerer, G. Hernandez-Sosa, U. Gengenbach, and M. Reischl. Image-based identification of optical quality and functional properties in inkjet-printed electronics using machine learning. *Journal of Intelligent Manufacturing*, pages 1–18, Springer, 2024. DOI: 10.1007/s10845-024-02385-4.

[Exl25b] D. Exler, M. Schutera, M. Reischl, and L. Rettenberger. Large means left: Political bias in large language models increases with their number of parameters. *arXiv preprint arXiv:2505.04393*, 2025. DOI: 10.48550/arXiv.2505.04393.

Supervised Theses

[Bos22] B. Boschert. *Automatische Segmentation der Herzkammerwand beim Modellorganismus Zebrafisch zur Detektion von Kardiohypertrophien*. Master thesis, Karlsruhe Institute of Technology, Karlsruhe, 2022.

[Wag22] J. Wagner. *Automated Detection of Floral Diversity Using Computer Vision*. Master thesis, Karlsruhe Institute of Technology, Karlsruhe, 2022.

[Bey23] N. Beyer. *Machine Vision Based Error Detection for 3D-Printing*. Master thesis, Karlsruhe Institute of Technology, Karlsruhe, 2023.

[Lan23] M. Lanz. *Unified Loss Merging Framework for Enhanced Semantic Segmentation*. Master thesis, University of Applied Sciences Ravensburg-Weingarten, Karlsruhe, 2023.

[Sch24] A. Schwarz. *Entwicklung eines Systems zur automatisierten kamerabasierten Erkennung lebender Insekten für das Monitoring in der Biodiversitätsforschung*. Master thesis, Karlsruhe Institute of Technology, Karlsruhe, 2024.

[Wei24] K. Weinmann. *Automatisierte Segementierungsmaskengenerierung mit Deep Learning*. Master thesis, Hochschule Aalen, Aalen, 2024.

Bibliography

[Abd10] H. Abdi and L. J. Williams. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, volume 2, pages 433–459, Wiley Online Library, 2010. DOI: 10.1002/wics.101. (cited on page 90).

[Abd21] M. Abdar et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, volume 76, pages 243–297, Elsevier, 2021. DOI: 10.1016/j.inffus.2021.05.008. (cited on page 121).

[Aga23] S. Agarwal, A. Sawant, T. Wong, S. Copp, J. Reyes-Zacarias, and S. Zinkle. Comparing U-Net and Mask R-CNN algorithms for deep learning-based segmentation of electron microscopy images containing cavities for nuclear reactor applications. In *Proceedings of the ICECCME*, pages 1–4, Tenerife, Spain, 2023. IEEE. DOI: 10.1109/ICECCME57830.2023.10252280. (cited on page 121).

[Akk24] I. B. Akkaya, S. S. Kathiresan, E. Arani, and B. Zonooz. Enhancing performance of vision transformers on small datasets through local inductive bias incorporation. *Pattern Recognition*, volume 153, page 110510, Elsevier, 2024. DOI: 10.1016/j.patcog.2024.110510. (cited on page 34).

[Alf19] E. Alfaro, X. B. Fonseca, E. M. Albornoz, C. E. Martínez, and S. C. Ramrez. A brief analysis of U-Net and Mask R-CNN for skin lesion segmentation. In *Proceedings of the IEEE IWOBI*, pages 123–126, Budapest, Hungary, 2019. DOI: 10.1109/IWOBI47054.2019.9114436. (cited on page 121).

[Amj20] A. B. Amjoud and M. Amrouch. Convolutional neural networks backbones for object detection. In *International Conference on Image and Signal Processing*, pages 282–289, Marrakesh, Morocco, 2020. Springer. (cited on page 36).

[An22] S. An, H. Zhu, C. Guo, B. Fu, C. Song, P. Tao, W. Shang, and T. Deng. Noncontact human-machine interaction based on hand-responsive infrared structural color. *Nature Communications*, volume 13, page 1446, Nature Publishing Group UK London, 2022. DOI: 10.1038/s41467-022-29197-5. (cited on page 1).

[Ans17] M. A. Ansari, D. Kurchaniya, and M. Dixit. A comprehensive analysis of image edge detection techniques. *International Journal of Multimedia and Ubiquitous Engineering*, volume 12, pages 1–12, 2017. DOI: 10.14257/ijmue.2017.12.11.01. (cited on page 32).

[Bai23] B. Bai, J. Tian, S. Luo, T. Wang, and S. Lyu. Yuseg: Yolo and unet is all you need for cell instance segmentation. In *Competitions in Neural Information Processing Systems*, pages 1–15, New Orleans, US, 2023. PMLR. (cited on page 121).

[Bal18] T. Baltrušaitis, C. Ahuja, and L.-P. Morency. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, volume 41, pages 423–443, IEEE, 2018. (cited on page 62).

[Bal23] J. Bals and M. Epple. Deep learning for automated size and shape analysis of nanoparticles in scanning electron microscopy. *RSC advances*, volume 13, pages 2795–2802, Royal Society of Chemistry, 2023. DOI: 10.1039/D2RA07812K. (cited on page 146).

[Bel66] R. Bellman. Dynamic programming. *Science*, volume 153, pages 34–37, American Association for the Advancement of Science, 1966. DOI: 10.1126/science.153.3731.34. (cited on page 36).

[Ber19] S. Berg et al. Ilastik: interactive machine learning for (bio) image analysis. *Nature methods*, volume 16, pages 1226–1232, Nature Publishing Group US New York, 2019. DOI: 10.1038/s41592-019-0582-9. (cited on page 60).

[Ber24a] F. Berens, J. Ambs, S. Elser, and M. Reischl. A novel approach to light detection and ranging sensor placement for autonomous driving vehicles using deep deterministic policy gradient algorithm. *SAE International Journal of Connected and Automated Vehicles*, volume 7, pages 303–308, 2024. DOI: 10.4271/12-07-03-0019. (cited on page 1).

[Ber24b] F. Berens, Y. Koschinski, M. K. Badami, M. Geimer, S. Elser, and M. Reischl. Adaptive training for robust object detection in autonomous driving environments. *IEEE Transactions on Intelligent Vehicles*, IEEE, 2024. DOI: 10.1109/TIV.2024.3439001. (cited on page 1).

[Bey23] N. Beyer. *Machine Vision Based Error Detection for 3D-Printing*. Master thesis, Karlsruhe Institute of Technology, Karlsruhe, 2023. (cited on page 195).

[Bie20] L. Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com. (cited on page 153).

[Böh23] M. Böhland, R. Bruch, S. Bäuerle, L. Rettenberger, and M. Reischl. Improving generative adversarial networks for patch-based unpaired image-to-image translation. *IEEE Access*, volume 11, pages 127895–127906, IEEE, 2023. DOI: 10.1109/ACCESS.2023.3331819. (cited on page 193).

[Bor23] P. Borlinghaus, F. Tausch, and L. Rettenberger. A purely visual re-id approach for bumblebees (bombus terrestris). *Smart Agricultural Technology*, volume 3, page 100135, Elsevier, 2023. DOI: 10.1016/j.atech.2022.100135. (cited on page 193).

[Bos22] B. Boschert. *Automatische Segmentation der Herzkammerwand beim Modellorganismus Zebrafisch zur Detektion von Kardiohypertrophien*. Master thesis, Karlsruhe Institute of Technology, Karlsruhe, 2022. (cited on page 195).

[Boy23] M. J. Boyle, Y. E. Goldman, and R. J. Composto. Enhancing nanoparticle detection in interferometric scattering (iSCAT) microscopy using a Mask R-CNN. *The Journal of Physical Chemistry B*, volume 127, pages 3737–3745, ACS Publications, 2023. DOI: 10.1021/acs.jpcb.3c00097. (cited on page 121).

[Bra00] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. (cited on pages 153 and 161).

[Brb20] M. Brbić, M. Zitnik, S. Wang, A. O. Pisco, R. B. Altman, S. Darmanis, and J. Leskovec. Mars: discovering novel cell types across heterogeneous single-cell experiments. *Nature Methods*, volume 17, pages 1200–1206, Nature Publishing Group US New York, 2020. DOI: 10.1038/s41592-020-00979-3. (cited on page 62).

[Bru22] R. Bruch, M. Vitacolonna, R. Rudolf, and M. Reischl. Prediction of fluorescent ki67 staining in 3D tumor spheroids. In *Current Directions in Biomedical Engineering*, volume 8, pages 305–308, Innsbruck, Austria, 2022. De Gruyter. DOI: 10.1515/cdbme-2022-1078. (cited on page 121).

[Bru23] R. Bruch, F. Keller, M. Böhland, M. Vitacolonna, L. Klinger, R. Rudolf, and M. Reischl. Synthesis of large scale 3D microscopic images of 3D cell cultures for training and benchmarking. *PLoS ONE*, volume 18, page e0283828, Public Library of Science, 2023. DOI: 10.1371/journal.pone.0283828. (cited on page 1).

[Bru25] R. Bruch, M. Vitacolonna, E. Nürnberg, S. Sauer, R. Rudolf, and M. Reischl. Improving 3D deep learning segmentation with biophysically motivated cell synthesis. *Communications Biology*, volume 8, page 43, Nature Publishing Group UK London, 2025. DOI: 10.1038/s42003-025-07469-2. (cited on page 1).

[Bus18] A. Buslaev et al. Albumentations: fast and flexible image augmentations. *arXiv preprint arXiv:1809.06839*, 2018. DOI: 10.3390/info11020125. (cited on pages 153 and 154).

[Che20a] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the ICML*, pages 1597–1607, Virtual Conference, 2020. PMLR. (cited on pages 35, 87, 98, 100, 169, and 170).

[Che20b] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. (cited on page 35).

[Che23] Q. Chen, C. Zheng, Y. Cui, Y.-R. Lin, and S. J. Zinkle. A deep learning model for automatic analysis of cavities in irradiated materials. *Computational Materials Science*, volume 221, page 112073, Elsevier, 2023. DOI: 10.1016/j.commatsci.2023.112073. (cited on page 121).

[Chi20] W. Chi, L. Ma, J. Wu, M. Chen, W. Lu, and X. Gu. Deep learning-based medical image segmentation with limited labels. *Physics in Medicine & Biology*, volume 65, page 235001, IOP Publishing, 2020. DOI: 10.1088/1361-6560/abc363. (cited on page 86).

[Cho19] C. Choy, J. Gwak, and S. Savarese. 4D spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE CVPR*, pages 3075–3084, Long Beach, USA, 2019. DOI: 10.1109/CVPR.2019.00319. (cited on pages 40, 153, 154, and 158).

[Cod18] N. C. Codella et al. Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (ISBI), hosted by the international skin imaging collaboration (ISIC). In *Proceedings of the IEEE ISBI*, pages 168–172, Melbourne, Australia, 2018. IEEE. DOI: 10.1109/ISBI.2018.8363547. (cited on pages 48 and 49).

[Coh09] I. Cohen, Y. Huang, J. Chen, J. Benesty, J. Benesty, J. Chen, Y. Huang, and I. Cohen. Pearson correlation coefficient. *Noise reduction in speech processing*, pages 1–4, Springer, 2009. DOI: 10.1007/978-3-642-00296-0_5. (cited on page 42).

[Coh21] R. Cohn, I. Anderson, T. Prost, J. Tiarks, E. White, and E. Holm. Instance segmentation for direct measurements of satellites in metal powders and automated microstructural characterization from image data. *Jom*, volume 73, pages 2159–2172, Springer, 2021. DOI: 10.1007/s11837-021-04899-1. (cited on page 121).

[Dai15] J. Dai, K. He, and J. Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *Proceedings of the IEEE ICCV*, pages 1635–1643, Santiago, Chile, 2015. DOI: 10.1109/ICCV.2015.191. (cited on page 61).

[Der21] T. Deruyttere, V. Milewski, and M.-F. Moens. Giving commands to a self-driving car: How to deal with uncertain situations? *Engineering applications of artificial intelligence*, volume 103, page 104257, Elsevier, 2021. DOI: 10.1016/j.engappai.2021.104257. (cited on page 121).

[Dmi19] K. Dmitriev and A. E. Kaufman. Learning multi-class segmentations from single-class datasets. In *Proceedings of the IEEE CVPR*, pages 9501–9511, Long Beach, USA, 2019. IEEE. DOI: 10.1109/CVPR.2019.00973. (cited on page 62).

[Don16] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016. (cited on page 35).

[Don19] J. Donahue and K. Simonyan. Large scale adversarial representation learning. *arXiv preprint arXiv:1907.02544*, 2019. (cited on page 35).

[Dos20] A. Dosovitskiy et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the ICLR*, San Diego, USA, 2020. (cited on pages 34 and 88).

[Dum16] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016. (cited on page 35).

[Est17] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *nature*, volume 542, pages 115–118, Nature Publishing Group, 2017. DOI: 10.1038/nature21056. (cited on page 121).

[Exl25a] D. Exler, M. Raimann, M. Münker, M. Rosin, J. E. U. Gómez, C. Niemeyer, M. Reischl, and L. Rettenberger. LLM Playground: A framework to compare enhancement methods for large language models as scientific assistants. *Advanced Intelligent Discovery*, Wiley, 2025. **Submitted: In Review**. No citations.

[Exl25b] D. Exler, M. Schutera, M. Reischl, and L. Rettenberger. Large means left: Political bias in large language models increases with their number of parameters. *arXiv preprint arXiv:2505.04393*, 2025. DOI: 10.48550/arXiv.2505.04393. (cited on page 193).

[Fal19] W. Falcon and contributors. PyTorch Lightning. Accessed: Aug. 23, 2024, GitHub: https://github.com/Lightning-AI/lightning, version 1.4, 2019. (cited on pages 153 and 154).

[Fan20] X. Fang and P. Yan. Multi-organ segmentation over partially labeled datasets with multi-scale feature abstraction. *Transactions on Medical Imaging*, pages 3619–3629, IEEE, IEEE, 06 2020. DOI: 10.1109/TMI.2020.3001036. (cited on page 62).

[Fuj20] S. Fujita and X.-H. Han. Cell detection and segmentation in microscopy images with improved Mask R-CNN. In *Proceedings of the ACCV*, Kyoto, Japan, 2020. DOI: 10.1007/978-3-030-69756-3_5. (cited on page 121).

[Gem92] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, volume 4, pages 1–58, MIT Press, 1992. DOI: 10.1162/neco.1992.4.1.1. (cited on page 18).

[Gho21] A. Ghosh, B. G. Sumpter, O. Dyck, S. V. Kalinin, and M. Ziatdinov. Ensemble learning-iterative training machine learning for uncertainty quantification and automated experiment in atom-resolved microscopy. *npj Computational Materials*, volume 7, page 100, Nature Publishing Group UK London, 2021. DOI: 10.1038/s41524-021-00569-7. (cited on pages 120 and 121).

[Gil21] J. Gildenblat and contributors. Pytorch library for cam methods. Accessed: Aug. 23, 2024, GitHub: https://github.com/jacobgil/pytorch-grad-cam, version 1.4.5, 2021. (cited on pages 153 and 154).

[Gir15a] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE ECCV*, pages 1440–1448, Amsterdam, The Netherlands, 2015. DOI: 10.1109/ICCV.2015.169. (cited on page 168).

[Gir15b] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence*, volume 38, pages 142–158, IEEE, 2015. DOI: 10.1109/TPAMI.2015.2437384. (cited on page 25).

[Goh16] C. Gohlke. Tifffile library. https://doi.org/10.5281/zenodo.6795861, 2016. (cited on page 153).

[Gom13] W. H. Gomaa and A. A. Fahmy. A survey of text similarity approaches. *international journal of Computer Applications*, volume 68, Citeseer, 2013. (cited on page 36).

[Gon18] G. González, G. R. Washko, and R. S. J. Estépar. Multi-structure segmentation from partially labeled datasets. application to body composition measurements on ct scans. *Image Analysis for Moving Organ, Breast, and Thoracic Images*, pages 215–224, Springer, 2018. DOI: 10.1007/978-3-030-00946-5_22. (cited on page 62).

[Goo14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, volume 27, 2014. (cited on page 35).

[Goo16] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep learning*. MIT press, 2016. (cited on pages 1, 17, 18, 21, 30, 31, 33, and 155).

[Gra19] S. Graham, Q. D. Vu, S. E. A. Raza, A. Azam, Y. W. Tsang, J. T. Kwak, and N. Rajpoot. Hover-net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images. *Medical image analysis*, volume 58, page 101563, Elsevier, 2019. DOI: 10.1016/j.media.2019.101563. (cited on pages 42 and 92).

[Gre05] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *Proceedings of the ALT*, pages 63–77, Singapore, Republic of Singapore, 2005. Springer. DOI: 10.1007/11564089_7. (cited on page 43).

[Gri20] J.-B. Grill et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. (cited on pages 35, 87, 98, 100, 169, and 170).

[Gui24] J. Gui, T. Chen, J. Zhang, Q. Cao, Z. Sun, H. Luo, and D. Tao. A survey on self-supervised learning: Algorithms, applications, and future trends. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, 2024. DOI: 10.1109/TPAMI.2024.3415112. (cited on page 88).

[Hal09] A. Halevy, P. Norvig, and F. Pereira. The unreasonable effectiveness of data. *IEEE intelligent systems*, volume 24, pages 8–12, IEEE, 2009. DOI: 10.1109/MIS.2009.36. (cited on page 61).

[Har20] C. R. Harris et al. Array programming with numpy. *Nature*, volume 585, pages 357–362, Nature Publishing Group UK London, 2020. DOI: 10.1038/s41586-020-2649-2. (cited on page 153).

[He15] K. He et al. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE ECCV*, pages 1026–1034, Zurich, Switzerland, 2015. IEEE. DOI: 10.1109/ICCV.2015.123. (cited on page 94).

[He16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE CVPR*, pages 770–778, Las Vegas, USA, 2016. DOI: 10.1109/CVPR.2016.90. (cited on pages 36, 37, 96, 100, 105, 154, 155, 169, and 175).

[He17] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *Proceedings of the IEEE ECCV*, pages 2961–2969, Venice, Italy, 2017. DOI: 10.1109/TPAMI.2018.2844175. (cited on pages 38, 121, and 122).

[He20] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE ECCV*, pages 9729–9738, Virtual Conference, 2020. IEEE. DOI: 10.1109/CVPR42600.2020.00975. (cited on pages 87, 98, 100, 101, 169, and 170).

[He22] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE CVPR*, pages 16000–16009, New Orleans, USA, 2022. DOI: 10.1109/CVPR52688.2022.01553. (cited on pages 35 and 88).

[Hei21] F. Heidecker, A. Hannan, M. Bieshaar, and B. Sick. Towards corner case detection by modeling the uncertainty of instance segmentation networks. In *Proceedings of the ICPR*, pages 361–374, Virtual Conference, 2021. Springer. DOI: 10.1007/978-3-030-68799-1_26. (cited on page 121).

[Hei23] F. Heidecker, A. El-Khateeb, and B. Sick. Sampling-based uncertainty estimation for an instance segmentation network. *arXiv preprint arXiv:2305.14977*, 2023. (cited on page 122).

[Hil16] J. Hill, G. Mulholland, K. Persson, R. Seshadri, C. Wolverton, and B. Meredig. Materials science with large-scale data and informatics: unlocking new opportunities. *Mrs Bulletin*, volume 41, pages 399–409, Cambridge University Press, 2016. DOI: 10.1557/mrs.2016.93. (cited on page 2).

[Hin02] G. E. Hinton and S. Roweis. Stochastic neighbor embedding. *Advances in neural information processing systems*, volume 15, 2002. (cited on page 44).

[Hor20] J. P. Horwath, D. N. Zakharov, R. Mégret, and E. A. Stach. Understanding important features of deep learning models for segmentation of high-resolution transmission electron microscopy images. *npj Computational Materials*, volume 6, page 108, Nature Publishing Group UK London, 2020. DOI: 10.1038/s41524-020-00363-x. (cited on page 120).

[Hou23] S. Hou, T. Zhou, Y. Liu, P. Dang, H. Lu, and H. Shi. Teeth U-Net: A segmentation model of dental panoramic x-ray images for context semantics and contrast enhancement. *Computers in Biology and Medicine*, volume 152, page 106296, Elsevier, 2023. DOI: 10.1016/j.compbiomed.2022.106296. (cited on page 120).

[Hül21] E. Hüllermeier and W. Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine learning*, volume 110, pages 457–506, Springer, 2021. DOI: 10.1007/s10994-021-05946-3. (cited on page 27).

[Ibt20] N. Ibtehaz and M. S. Rahman. Multiresunet: Rethinking the U-Net architecture for multimodal biomedical image segmentation. *Neural networks*, volume 121, pages 74–87, Elsevier, 2020. DOI: 10.1016/j.neunet.2019.08.025. (cited on page 120).

[Ios22] A. Iosifidis and A. Tefas. *Deep learning for robot perception and cognition.* Academic Press, 2022. (cited on page 40).

[Isi20] L. F. Isikdogan et al. Semifreddonets: Partially frozen neural networks for efficient computer vision systems. In *Proceedings of the ECCV*, pages 193–208, Virtual Conference, 2020. Springer. DOI: 10.1007/978-3-030-58583-9_12. (cited on page 87).

[Ive62] K. E. Iverson. A programming language. In *Proceedings of the Spring Joint Computer Conference*, pages 345–351, San Francisco, USA, 1962. DOI: 10.1145/1460833.1460872. (cited on page 125).

[Jac12] P. Jaccard. The distribution of the flora in the alpine zone. *New phytologist*, volume 11, pages 37–50, Wiley Online Library, 1912. DOI: 10.1111/j.1469-8137.1912.tb05611.x. (cited on page 41).

[Jac22] R. Jacobs. Deep learning object detection in materials science: Current state and future directions. *Computational Materials Science*, volume 211, page 111527, Elsevier, 2022. DOI: 10.1016/j.commatsci.2022.111527. (cited on page 121).

[Jac23] R. Jacobs, P. Patki, M. J. Lynch, S. Chen, D. Morgan, and K. G. Field. Materials swelling revealed through automated semantic segmentation of cavities in electron microscopy images. *Scientific Reports*, volume 13, page 5178, Nature Publishing Group UK London, 2023. DOI: 10.1038/s41598-023-32454-2. (cited on page 121).

[Jai20] A. Jaiswal et al. A survey on contrastive self-supervised learning. *Technologies*, volume 9, pages 2–22, MDPI, 2020. DOI: 10.3390/technologies9010002. (cited on page 90).

[Jee22] K. Jeeveswaran, S. Kathiresan, A. Varma, O. Magdy, B. Zonooz, and E. Arani. A comprehensive study of vision transformers on dense prediction tasks. In *Proceedings of the VISIGRAPP*, pages 213–223, Virtual Conference, 2022. INSTICC, SciTePress. DOI: 10.5220/0010917800003124. (cited on page 34).

[Jia18] H. Jiang, B. Kim, M. Guan, and M. Gupta. To trust or not to trust a classifier. *Advances in neural information processing systems*, volume 31, 2018. (cited on page 121).

[Jin20] L. Jing and Y. Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, 2020. DOI: 10.1109/TPAMI.2020.2992393. (cited on page 23).

[Kan00] E. R. Kandel, J. H. Schwartz, T. M. Jessell, S. Siegelbaum, A. J. Hudspeth, and S. Mack. *Principles of neural science*, volume 4. McGraw-hill New York, 2000. (cited on page 30).

[Ken17] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, volume 30, 2017. (cited on page 26).

[Keo11] E. Keogh and A. Mueen. Curse of dimensionality. *Encyclopedia of Machine Learning and Data Mining*, pages 257–258, Springer, 2011. DOI: 10.1007/978-1-4899-7687-1_192. (cited on page 90).

[Kin14] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. (cited on pages 160, 162, 168, and 174).

[Kir19] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic segmentation. In *Proceedings of the IEEE CVPR*, pages 9404–9413, Long Beach, USA, 2019. DOI: 10.1109/CVPR.2019.00963. (cited on page 26).

[Kir22] I. Kiran, B. Raza, A. Ijaz, and M. A. Khan. Denseres-unet: Segmentation of overlapped/clustered nuclei from multi organ histopathology images. *Computers in Biology and Medicine*, volume 143, page 105267, Elsevier, 2022. DOI: 10.1016/j.compbiomed.2022.105267. (cited on page 121).

[Kol16] A. Kolesnikov and C. H. Lampert. Seed, expand and constrain: Three principles for weakly-supervised image segmentation. In *Proceedings of the ECCV*, pages 695–711, Amsterdam, The Netherlands, 2016. Springer. DOI: 10.1007/978-3-319-46493-0_42. (cited on page 61).

[Kol19] A. Kolesnikov, X. Zhai, and L. Beyer. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE CVPR*, pages 1920–1929, Long Beach, USA, 2019. DOI: 10.1109/CVPR.2019.00202. (cited on page 86).

[Kon19] F. Kong, C. Chen, B. Huang, L. M. Collins, K. Bradbury, and J. M. Malof. Training a single multi-class convolutional segmentation network using multiple datasets with heterogeneous labels: Preliminary results. In *Proceedings of the IGARSS*, pages 3903–3906, Yokohama, Japan, 2019. IEEE. DOI: 10.1109/IGARSS.2019.8898617. (cited on pages 62, 67, and 68).

[Kor19] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton. Similarity of neural network representations revisited. In *Proceedings of the ICML*, pages 3519–3529, Long Beach, USA, 2019. PMLR. (cited on page 43).

[Kra91] M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, volume 37, pages 233–243, Wiley Online Library, 1991. DOI: 10.1002/aic.690370209. (cited on page 22).

[Kum17] S. Kumra and C. Kanan. Robotic grasp detection using deep convolutional neural networks. In *Proceedings of the IEEE IROS*, pages 769–776, Vancouver, British Columbia, 2017. IEEE. DOI: 10.1109/IROS.2017.8202237. (cited on page 1).

[Kum19] N. Kumar et al. A multi-organ nucleus segmentation challenge. *IEEE transactions on medical imaging*, volume 39, pages 1380–1391, IEEE, 2019. DOI: 10.1109/TMI.2019.2947628. (cited on pages 42, 48, 49, and 92).

[Lan23] M. Lanz. *Unified Loss Merging Framework for Enhanced Semantic Segmentation.* Master thesis, University of Applied Sciences Ravensburg-Weingarten, Karlsruhe, 2023. (cited on page 195).

[LeC05] Y. LeCun and F. J. Huang. Loss functions for discriminative training of energy-based models. In *Proceedings of the IWSMAI*, pages 206–213, Bridgetown, Barbados, 2005. PMLR. (cited on page 36).

[LeC15] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, volume 521, pages 436–444, Nature Publishing Group UK London, 2015. DOI: 10.1038/nature14539. (cited on page 19).

[LeC21] Y. LeCun and I. Misra. Self-supervised learning: The dark matter of intelligence, 2021. (cited on pages 23, 24, and 86).

[Leo19] M. Leonardi, D. Mazzini, and R. Schettini. Training efficient semantic segmentation cnns on multiple datasets. In *Proceedings of the ICIAP*, pages 303–314, Trento, Italy, 2019. Springer. DOI: 10.1007/978-3-030-30645-8_28. (cited on page 62).

[Li20] S. Li et al. Pytorch distributed: Experiences on accelerating data parallel training. *Proceedings of the VLDB Endowment*, volume 13, 2020. DOI: 10.14778/3415478.3415530. (cited on page 154).

[Lin14] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the ECCV*, pages 740–755, Zurich, Switzerland, 2014. Springer. DOI: 10.1007/978-3-319-10602-1_48. (cited on page 61).

[Lin16] D. Lin, J. Dai, J. Jia, K. He, and J. Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *Proceedings of the IEEE CVPR*, pages 3159–3167, Las Vegas, USA, 2016. DOI: 10.1109/CVPR.2016.344. (cited on page 60).

[Lin17] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE CVPR*, pages 2117–2125, Honolulu, USA, 2017. DOI: 10.1109/CVPR.2017.106. (cited on page 38).

[Lin22] B. Lin, N. Emami, D. A. Santos, Y. Luo, S. Banerjee, and B.-X. Xu. A deep learned nanowire segmentation model using synthetic data augmentation. *npj Computational Materials*, volume 8, page 88, Nature Publishing Group UK London, 2022. DOI: 10.1038/s41524-022-00767-x. (cited on page 121).

[Liu21a] Y. Liu, E. Sangineto, W. Bi, N. Sebe, B. Lepri, and M. Nadai. Efficient training of visual transformers with small datasets. *Advances in Neural Information Processing Systems*, volume 34, pages 23818–23830, 2021. (cited on page 34).

[Liu21b] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE ICCV*, pages 10012–10022, Montreal, Canada, 2021. DOI: 10.1109/ICCV48922.2021.00986. (cited on pages 157 and 158).

[Liu22] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s. In *Proceedings of the IEEE CVPR*, pages 11976–11986, New Orleans, USA, 2022. DOI: 10.1109/CVPR52688.2022.01167. (cited on pages 39, 95, 109, and 112).

[Llo82] S. Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, volume 28, pages 129–137, IEEE, 1982. DOI: 10.1109/TIT.1982.1056489. (cited on page 64).

[Lon15] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE CVPR*, pages 3431–3440, Boston, USA, 2015. DOI: 10.1109/TPAMI.2016.2572683. (cited on page 33).

[Los16] I. Loshchilov and F. Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. (cited on page 168).

[Los17] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. (cited on pages 171 and 175).

[Lu16] Z. Lu, G. Carneiro, A. P. Bradley, D. Ushizima, M. S. Nosrati, A. G. Bianchi, C. M. Carneiro, and G. Hamarneh. Evaluation of three algorithms for the segmentation of overlapping cervical cells. *IEEE journal of biomedical and health informatics*, volume 21, pages 441–450, IEEE, 2016. DOI: 10.1109/JBHI.2016.2519686. (cited on pages 47 and 49).

[Mac67] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Berkeley Symposium on Mathematical Statistics and Probability*, volume 5, pages 281–298, 1967. (cited on page 64).

[Mah22] A. Mahbod, G. Schaefer, G. Dorffner, S. Hatamikia, R. Ecker, and I. Ellinger. A dual decoder U-Net-based model for nuclei instance segmentation in hematoxylin and eosin-stained histological images. *Frontiers in Medicine*, volume 9, page 978146, Frontiers Media SA, 2022. DOI: 10.3389/fmed.2022.978146. (cited on page 121).

[Mai16] T. Maintainers. Torchvision: Pytorch's computer vision library. https://github.com/pytorch/vision, 2016. (cited on page 153).

[Mas16] F. Massa, R. Marlet, and M. Aubry. Crafting a multi-task cnn for viewpoint estimation. In *Proceedings of the BMVC*, pages 91.1–91.12, York, United Kingdom, 2016. DOI: 10.5244/C.30.91. (cited on page 129).

[Mat21] A. Mathis, T. Biasi, S. Schneider, M. Yuksekgonul, B. Rogers, M. Bethge, and M. W. Mathis. Pretraining boosts out-of-domain robustness for pose estimation. In *Proceedings of the WACV*, pages 1859–1868, Virtual Conference, 2021. DOI: 10.1109/{WACV}48630.2021.00190. (cited on pages 2 and 85).

[McA05] J. McAuley, J. Ming, D. Stewart, and P. Hanna. Subband correlation and robust speech recognition. *IEEE Transactions on Speech and Audio Processing*, volume 13, pages 956–964, IEEE, 2005. DOI: 10.1109/TSA.2005.851952. (cited on page 43).

[McI18] L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018. (cited on page 44).

[Mel18] P. Meletis and G. Dubbelman. Training of convolutional networks on multiple heterogeneous datasets for street scene semantic segmentation. In *Proceedings of the IEEE IV*, pages 1045–1050, Changshu, China, 2018. IEEE. DOI: 10.1109/IVS.2018.8500398. (cited on page 62).

[Mik08] R. Mikut. *Data Mining in der Medizin und Medizintechnik*, volume 22. KIT Scientific Publishing, 2008. (cited on pages 19 and 20).

[Mis20] I. Misra and L. v. d. Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE CVPR*, pages 6707–6717, Seattle, USA, 2020. DOI: 10.1109/CVPR42600.2020.00674. (cited on page 35).

[Mit97] T. M. Mitchell et al. Machine learning. McGraw-hill New York, 1997. DOI: 10.1007/978-1-4613-2279-5. (cited on page 15).

[Moh18] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of machine learning*. MIT press, 2018. (cited on pages 15, 16, 17, 18, 19, and 20).

[Mün24] F. R. Münke, J. Schützke, F. Berens, and M. Reischl. A review of adaptable conventional image processing pipelines and deep learning on limited datasets. *Machine Vision and Applications*, volume 35, page 25, Springer, 2024. DOI: 10.1007/s00138-023-01501-3. (cited on page 61).

[Nor16] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Proceedings of the ICCV*, pages 69–84, Amsterdam, The Netherlands, 2016. Springer. DOI: 10.1007/978-3-319-46466-4_5. (cited on page 35).

[Ohr21] K. Ohri and M. Kumar. Review on self-supervised image recognition using deep neural networks. *Knowledge-Based Systems*, volume 224, page 107090, Elsevier, 2021. DOI: 10.1016/j.knosys.2021.107090. (cited on page 88).

[OMa] N. OMahony, S. Campbell, A. Carvalho, S. Harapanahalli, G. V. Hernandez, L. Krpalkova, D. Riordan, and J. Walsh. Deep learning vs. traditional computer vision. In *Proceedings of the CVC*, volume 1. (cited on page 86).

[Ots75] N. Otsu et al. A threshold selection method from gray-level histograms. *Automatica*, volume 11, pages 23–27, 1975. DOI: 10.1109/TSMC.1979.4310076. (cited on page 64).

[Ova19] Y. Ovadia et al. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, volume 32, 2019. (cited on page 121).

[Owe18] A. Owens and A. A. Efros. Audio-visual scene analysis with self-supervised multisensory features. In *Proceedings of the ECCV*, pages 631–648, Munich, Germany, 2018. DOI: 10.1007/978-3-030-01231-1_39. (cited on page 23).

[Pan09] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, volume 22, pages 1345–1359, IEEE, 2009. DOI: 10.1109/TKDE.2009.191. (cited on page 16).

[Pap24] L. Papa, P. Russo, I. Amerini, and L. Zhou. A survey on efficient vision transformers: algorithms, techniques, and performance benchmarking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, 2024. DOI: 10.1109/TPAMI.2024.3392941. (cited on page 34).

[Pas19] A. Paszke et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, volume 32, 2019. (cited on pages 153 and 155).

[Ped11] F. Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, volume 12, pages 2825–2830, 2011. (cited on page 153).

[Pei24] G. Pei, T. Chen, X. Jiang, H. Liu, Z. Sun, and Y. Yao. Videomac: Video masked autoencoders meet convnets. In *Proceedings of the IEEE CVPR*, pages 22733–22743, Seattle, USA, 2024. DOI: 10.1109/CVPR52733.2024.02145. (cited on page 158).

[Pet18] O. Petit, N. Thome, A. Charnoz, A. Hostettler, and L. Soler. Handling missing annotations for semantic segmentation with deep convnets. In *Proceedings of the DLMIA*, pages 20–28, Granada, Spain, 2018. Springer. DOI: 10.1007/978-3-030-00889-5_3. (cited on page 62).

[Pol24] M. Polomoshnov, K.-M. Reichert, L. Rettenberger, M. Ungerer, G. Hernandez-Sosa, U. Gengenbach, and M. Reischl. Image-based identification of optical quality and functional properties in inkjet-printed electronics using machine learning. *Journal of Intelligent Manufacturing*, pages 1–18, Springer, 2024. DOI: 10.1007/s10845-024-02385-4. (cited on page 193).

[Pop19] A. A. Popova, T. Tronser, K. Demir, P. Haitz, K. Kuodyte, V. Starkuviene, P. Wajda, and P. A. Levkin. Facile one step formation and screening of tumor spheroids using droplet-microarray platform. *Small*, volume 15, page 1901299, Wiley Online Library, 2019. DOI: 10.1002/smll.201901299. (cited on pages 47 and 49).

[Pri21] S. E. Price, M. A. Gleason, B. C. Sousa, D. L. Cote, and R. Neamtu. Automated and refined application of convolutional neural network modeling to metallic powder particle satellite detection. *Integrating Materials and Manufacturing Innovation*, volume 10, pages 661–676, Springer, 2021. DOI: 10.1007/s40192-021-00240-5. (cited on page 121).

[Rag23] M. Ragone, R. Shahabazian-Yassar, F. Mashayek, and V. Yurkiv. Deep learning modeling in microscopy imaging: A review of materials science applications. *Progress in Materials Science*, page 101165, Elsevier, 2023. DOI: 10.1016/j.pmatsci.2023.101165. (cited on page 121).

[Rah21] G. Rahmon, I. E. Toubal, and K. Palaniappan. Extending U-Net network for improved nuclei instance segmentation accuracy in histopathology images. In *Proceedings of the IEEE AIPR*, pages 1–7, Washington DC, USA, 2021. IEEE. DOI: 10.1109/AIPR52630.2021.9762213. (cited on page 121).

[Ram20] H. G. Ramaswamy et al. Ablation-cam: Visual explanations for deep convolutional network via gradient-free localization. In *Proceedings of the WACV*, pages 983–991, Snowmass Village, USA, 2020. (cited on page 114).

[Ren15] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, volume 28, 2015. DOI: 10.1109/TPAMI.2016.2577031. (cited on page 38).

[Ret22] L. Rettenberger, M. Schilling, and M. Reischl. Annotation efforts in image segmentation can be reduced by neural network bootstrapping. In *Current Directions in Biomedical Engineering*, volume 8, pages 329–332, Innsbruck, Austria, 2022. De Gruyter. DOI: 10.1515/cdbme-2022-1084. (cited on pages 64, 73, and 191).

[Ret23a] L. Rettenberger, F. R. Münke, R. Bruch, and M. Reischl. Mask R-CNN outperforms U-Net in instance segmentation for overlapping cells. In *Current Directions in Biomedical Engineering*, volume 9, pages 335–338, Duisburg, Germany, 2023. De Gruyter. DOI: 10.1515/cdbme-2023-1084. (cited on pages 1, 124, 133, 135, 137, and 191).

[Ret23b] L. Rettenberger, M. Schilling, S. Elser, M. Böhland, and M. Reischl. Self-supervised learning for annotation efficient biomedical image segmentation. *IEEE Transactions on Biomedical Engineering*, IEEE, 2023. DOI: 10.1109/TBME.2023.3252889. (cited on pages 92, 97, 103, 171, and 191).

[Ret24a] L. Rettenberger, N. Beyer, I. Sieber, and M. Reischl. Fault detection in 3D-printing with deep learning. In *Proceedings of the IEEE ICCE*, pages 1–4, Las Vegas, USA, 2024. IEEE. DOI: 10.1109/ICCE59016.2024.10444198. (cited on pages 1 and 191).

[Ret24b] L. Rettenberger, M. F. Münker, M. Schutera, C. M. Niemeyer, K. S. Rabe, and M. Reischl. Using large language models for extracting structured information from scientific texts. In *Current Directions in Biomedical Engineering*, volume 10, pages 526–529, Stuttgart, Germany, 2024. De Gruyter. DOI: 10.1515/cdbme-2024-2129. (cited on page 192).

[Ret24c] L. Rettenberger, N. J. Szymanski, Y. Zeng, J. Schützke, S. Wang, G. Ceder, and M. Reischl. Uncertainty-aware particle segmentation for electron microscopy at varied length scales. *npj Computational Materials*, volume 10, page 124, Nature Publishing Group UK London, 2024. DOI: 10.1038/s41524-024-01302-w. (cited on pages 1, 2, 53, 109, 130, 139, 141, 143, 145, 146, 160, 172, 179, 181, and 192).

[Ret25a] L. Rettenberger, M. Reischl, and M. Schutera. Assessing political bias in large language models. *Journal of Computational Social Science*, volume 8, pages 1–17, Springer, 2025. DOI: 10.1007/s42001-025-00376-w. (cited on page 192).

[Ret25b] L. Rettenberger, N. J. Szymanski, A. Giunto, O. Dartsi, A. Jain, G. Ceder, V. Hagenmeyer, and M. Reischl. Leveraging unlabeled SEM datasets with self-supervised learning for enhanced particle segmentation. *npj Computational Materials*, volume 11, page 289, Nature Publishing Group UK London, 2025. DOI: 10.1038/s41524-025-01802-3. (cited on pages 34, 57, 109, 111, 112, 114, 116, and 192).

[Rez14] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and variational inference in deep latent gaussian models. In *Proceedings of the ICML*, volume 2, page 2, Beijing, China, 2014. Citeseer. (cited on page 35).

[RM23] F. R. Münke, L. Rettenberger, A. Popova, and M. Reischl. A lightweight framework for semantic segmentation of biomedical images. In *Current Directions in Biomedical Engineering*, volume 9, pages 190–193, Duisburg, Germany, 2023. De Gruyter. DOI: 10.1515/cdbme-2023-1048. (cited on page 193).

[Ron15] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Proceedings of the MICCAI*, pages 234–241, Munich, Germany, 2015. Springer. DOI: 10.1007/978-3-319-24574-4_28. (cited on pages 34, 37, 65, 96, 120, and 161).

[Rus02] S. Russell and P. Norvig. Artificial intelligence: a modern approach. *Pearson*, 2002. (cited on pages 20, 21, and 22).

[Rus15] O. Russakovsky et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, volume 115, pages 211–252, Springer, 2015. DOI: 10.1007/s11263-015-0816-y. (cited on pages 2, 61, and 85).

[San18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE CVPR*, pages 4510–4520, Salt Lake City, USA, 2018. (cited on pages 39 and 157).

[Sch19] M. Schutera, S. Just, J. Gierten, R. Mikut, M. Reischl, and C. Pylatiuk. Machine learning methods for automated quantification of ventricular dimensions. *Zebrafish*, volume 16, pages 542–545, Mary Ann Liebert, Inc., 2019. DOI: 10.1089/zeb.2019.1754. (cited on pages 47 and 49).

[Sch20] T. Scherr, K. Löffler, M. Böhland, and R. Mikut. Cell segmentation and tracking using cnn-based distance predictions and a graph-based matching strategy. *PLoS ONE*, volume 15, page e0243219, Public Library of Science, 2020. DOI: 10.1371/journal.pone.0243219. (cited on pages 38, 121, and 168).

[Sch21a] M. P. Schilling, L. Rettenberger, F. Münke, H. Cui, A. A. Popova, P. A. Levkin, R. Mikut, and M. Reischl. Label assistant: a workflow for assisted data annotation in image segmentation tasks. In *Proceedings—31. Workshop Computational Intelligence*, pages 211–234, Berlin, Germany, 2021. DOI: 10.58895/ksp/1000138532-14. (cited on page 192).

[Sch21b] M. P. Schilling, S. Schmelzer, J. E. U. Gómez, A. A. Popova, P. A. Levkin, and M. Reischl. Grid screener: A tool for automated high-throughput screening on biochemical and biological analysis platforms. *IEEE Access*, volume 9, pages 166027–166038, IEEE, 2021. DOI: 10.1109/ACCESS.2021.3135709. (cited on page 3).

[Sch22a] M. P. Schilling, N. Ahuja, L. Rettenberger, T. Scherr, and M. Reischl. Impact of annotation noise on histopathology nucleus segmentation. In *Current Directions in Biomedical Engineering*, volume 8, pages 197–200, Innsbruck, Austria, 2022. De Gruyter. DOI: 10.1515/cdbme-2022-1051. (cited on page 192).

[Sch22b] M. P. Schilling, T. Scherr, F. R. Münke, O. Neumann, M. Schutera, R. Mikut, and M. Reischl. Automated annotator variability inspection for biomedical image segmentation. *IEEE Access*, volume 10, pages 2753–2765, IEEE, 2022. DOI: 10.1109/ACCESS.2022.3140378. (cited on pages 2 and 63).

[Sch22c] M. P. Schilling, S. Schmelzer, L. Klinger, and M. Reischl. Kaida: a modular tool for assisting image annotation in deep learning. *Journal of Integrative Bioinformatics*, volume 19, page 20220018, De Gruyter, 2022. DOI: 10.1515/jib-2022-0018. (cited on page 75).

[Sch22d] M. Schutera, L. Rettenberger, C. Pylatiuk, and M. Reischl. Methods for the frugal labeler: Multi-class semantic segmentation on heterogeneous labels. *PLoS ONE*, volume 17, page e0263656, Public Library of Science, 2022. DOI: 10.1371/journal.pone.0263656. (cited on pages 29, 52, 70, 76, 77, 78, 80, 81, 120, and 191).

[Sch22e] M. Schutera, L. Rettenberger, and M. Reischl. Automated zebrafish phenotype pattern recognition: 6 years ago, and now. *Zebrafish*, volume 19, pages 213–217, Mary Ann Liebert, Inc., 2022. DOI: 10.1089/zeb.2022.0027. (cited on page 193).

[Sch23] M. P. Schilling et al. Automated high-throughput image processing as part of the screening platform for personalized oncology. *Scientific Reports*, volume 13, page 5107, Nature Publishing Group UK London, 2023. DOI: 10.1038/s41598-023-32144-z. (cited on page 1).

[Sch24] A. Schwarz. *Entwicklung eines Systems zur automatisierten kamerabasierten Erkennung lebender Insekten für das Monitoring in der Biodiversitätsforschung*. Master thesis, Karlsruhe Institute of Technology, Karlsruhe, 2024. (cited on page 195).

[Shi21] G. Shi, L. Xiao, Y. Chen, and S. K. Zhou. Marginal loss and exclusion loss for partially supervised multi-organ segmentation. *Medical Image Analysis*, volume 70, page 101979, Elsevier, 2021. DOI: 10.1016/j.media.2021.101979. (cited on page 62).

[Sho19] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, volume 6, pages 1–48, Springer, 2019. DOI: 10.1186/s40537-019-0197-0. (cited on page 19).

[Si23] C. Si, M. S. M. Rahim, Y. Mianzhou, N. Li, and C. Hongyu. Unet-based polyp segmentation: A survey. In *Proceedings of the IEEE ICOCO*, pages 154–159, Langkawi Island, Malaysia, 2023. IEEE. DOI: 10.1109/ICOCO59262.2023.10397673. (cited on page 120).

[Sia18] M. Siam, M. Gamal, M. Abdel-Razek, S. Yogamani, M. Jagersand, and H. Zhang. A comparative study of real-time semantic segmentation for autonomous driving. In *Proceedings of the IEEE CVPR Workshops*, pages 587–597, Salt Lake City, USA, 2018. DOI: 10.1109/CVPRW.2018.00101. (cited on page 120).

[Sid24] Q. M. Siddiqui, S. Starke, and P. Steinbach. Uncertainty estimation in instance segmentation with star-convex shapes. In *Proceedings of the WACV*, pages 1424–1433, Waikoloa, USA, 2024. DOI: 10.1109/{WACV}57701.2024.00145. (cited on page 122).

[Smi23] M. Smith and F. Ferrie. Uncertainty estimation in deep learning for panoptic segmentation. *arXiv preprint arXiv:2304.02098*, 2023. (cited on page 122).

[Sno12] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, volume 25, 2012. (cited on page 176).

[Sub21] A. Subramanian. Pytorch model compare. Accessed: Sep. 23, 2024, GitHub: https://github.com/AntixK/PyTorch-Model-Compare, version 0.21, 2021. (cited on pages 153 and 154).

[Sud17] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Proceedings of the DLMIA*, pages 240–248, Québec, Canada, 2017. Springer. DOI: 10.1007/978-3-319-67558-9_28. (cited on pages 42, 68, 92, 160, 168, and 174).

[Sun17] C. Sun, A. Shrivastava, S. Singh, and A. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE ECCV*, pages 843–852, Venice, Italy, 2017. DOI: 10.1109/ICCV.2017.97. (cited on page 86).

[Sze22] R. Szeliski. *Computer vision: algorithms and applications*. Springer Nature, 2022. DOI: 10.1007/978-1-84882-935-0. (cited on pages 24 and 25).

[Szy21] N. J. Szymanski, Y. Zeng, H. Huo, C. J. Bartel, H. Kim, and G. Ceder. Toward autonomous design and synthesis of novel inorganic materials. *Materials Horizons*, volume 8, pages 2169–2198, Royal Society of Chemistry, 2021. DOI: 10.1039/D1MH00495F. (cited on page 2).

[Szy23] N. J. Szymanski et al. An autonomous laboratory for the accelerated synthesis of novel materials. *Nature*, volume 624, pages 86–91, Nature Publishing Group UK London, 2023. DOI: 10.1038/s41586-023-06734-w. (cited on page 1).

[Tra19] L.-A. Tran and M.-H. Le. Robust U-Net-based road lane markings detection for autonomous driving. In *Proceedings of the ICSSE*, pages 62–66, Dong Hoi City, Vietnam, 2019. IEEE. DOI: 10.1109/ICSSE.2019.8823532. (cited on page 120).

[Tri22] P. Trivedi, E. S. Lubana, M. Heimann, D. Koutra, and J. Thiagarajan. Analyzing data-centric properties for graph contrastive learning. *Advances in Neural Information Processing Systems*, volume 35, pages 14030–14043, 2022. (cited on pages 20 and 22).

[Tun22] L. Tunstall, L. Von Werra, and T. Wolf. Natural language processing with transformers. *O'Reilly Media, Inc.*, 2022. (cited on page 34).

[Tur19] A. Tureckova and A. J. Rodríguez-Sánchez. Isles challenge: U-shaped convolution neural network with dilated convolution for 3D stroke lesion segmentation. In *BrainLes: International MICCAI Brainlesion Workshop*, pages 319–327, Granada, Spain, 2019. Springer. DOI: 10.1007/978-3-030-11723-8_32. (cited on page 121).

[VdW14] S. Van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu. scikit-image: image processing in python. *PeerJ*, volume 2, page e453, PeerJ Inc., 2014. DOI: 10.7717/peerj.453. (cited on pages 153 and 161).

[Vie22] T. Viering and M. Loog. The shape of learning curves: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 45, pages 7799–7819, IEEE, 2022. DOI: 10.1109/TPAMI.2022.3220744. (cited on page 26).

[Vin08] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the ICML*, pages 1096–1103, Helsinki, Finland, 2008. PMLR. DOI: 10.1145/1390156.1390294. (cited on page 35).

[Vir20] P. Virtanen et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, volume 17, pages 261–272, 2020. DOI: 10.1038/s41592-019-0686-2. (cited on page 153).

[Vu21] M. H. Vu, G. Norman, T. Nyholm, and T. Löfstedt. A data-adaptive loss function for incomplete data and incremental learning in semantic image segmentation. *IEEE Transactions on Medical Imaging*, volume 41, pages 1320–1330, IEEE, 2021. (cited on pages 62 and 67).

[Wag22] J. Wagner. *Automated Detection of Floral Diversity Using Computer Vision*. Master thesis, Karlsruhe Institute of Technology, Karlsruhe, 2022. (cited on page 195).

[Wak01] Y. Wakamatsu, S. Pristyazhnyuk, M. Kinoshita, M. Tanaka, and K. Ozato. The see-through medaka: a fish model that is transparent throughout life. *Proceedings of the National Academy of Sciences*, volume 98, pages 10046–10050, National Acad Sciences, 2001. DOI: 10.1073/pnas.181204298. (cited on page 47).

[Wan21a] F. Wang and H. Liu. Understanding the behaviour of contrastive loss. In *Proceedings of the IEEE CVPR*, pages 2495–2504, Nashville, USA, 2021. DOI: 10.1109/CVPR46437.2021.00252. (cited on page 61).

[Wan21b] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li. Dense contrastive learning for self-supervised visual pre-training. In *Proceedings of the IEEE ECCV*, pages 3024–3033, Montreal, Canada, 2021. IEEE. DOI: 10.1109/CVPR46437.2021.00304. (cited on pages 87, 98, 100, 168, 169, 170, and 171).

[Wei17] Y. Wei, J. Feng, X. Liang, M.-M. Cheng, Y. Zhao, and S. Yan. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *Proceedings of the IEEE CVPR*, pages 1568–1576, Honolulu, USA, 2017. DOI: 10.1109/CVPR.2017.687. (cited on page 61).

[Wei24] K. Weinmann. *Automatisierte Segementierungsmaskengenerierung mit Deep Learning*. Master thesis, Hochschule Aalen, Aalen, 2024. (cited on page 195).

[Wol20] C. Wolff. Using otsu's method to generate data for training deep learning image segmentation models. Accessed: May. 05, 2025, URL: https://devblogs.microsoft.com/ise/using-otsus-method-generate-data-training-deep-learning-image-segmentation-models/, 2020. (cited on page 61).

[Woo23] S. Woo, S. Debnath, R. Hu, X. Chen, Z. Liu, I. S. Kweon, and S. Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE CVPR*, pages 16133–16142, Vancouver, Canada, 2023. DOI: 10.1109/CVPR52729.2023.01548. (cited on pages 39, 40, 88, 112, 157, and 171).

[Wu22] H. Wu, N. Souedet, C. Jan, C. Clouchoux, and T. Delzescaux. A general deep learning framework for neuron instance segmentation based on efficient unet and morphological post-processing. *Computers in Biology and Medicine*, volume 150, page 106180, Elsevier, 2022. DOI: 10.1016/j.compbiomed.2022.106180. (cited on page 121).

[Wüh24] L. Wührl, L. Rettenberger, R. Meier, E. Hartop, J. Graf, and C. Pylatiuk. Entomoscope: An open-source photomicroscope for biodiversity discovery. *IEEE Access*, IEEE, 2024. DOI: 10.1109/ACCESS.2024.3355272. (cited on page 193).

[Xia18] X. Xiao, S. Lian, Z. Luo, and S. Li. Weighted res-unet for high-quality retina vessel segmentation. In *Proceedings of the ITME*, pages 327–331, Hangzhou, China, 2018. IEEE. DOI: 10.1109/ITME.2018.00080. (cited on page 120).

[Xie18] Y. Xie and D. Richmond. Pre-training on grayscale imagenet improves medical image classification. In *Proceedings of the ECCV*, pages 476–484, Munich, Germany, 2018. DOI: 10.1007/978-3-030-11024-6_37. (cited on pages 2 and 85).

[Xie21] E. Xie, J. Ding, W. Wang, X. Zhan, H. Xu, P. Sun, Z. Li, and P. Luo. Detco: Unsupervised contrastive learning for object detection. In *Proceedings of the IEEE ECCV*, pages 8392–8401, Montreal, Canada, 2021. IEEE. DOI: 10.1109/ICCV48922.2021.00828. (cited on pages 87, 98, 100, 169, and 170).

[Xu19] J. Xu, L. Xiao, and A. M. López. Self-supervised domain adaptation for computer vision tasks. *IEEE Access*, volume 7, pages 156694–156706, IEEE, 2019. DOI: 10.1109/ACCESS.2019.2949697. (cited on page 86).

[Xu21] J. Xu. A review of self-supervised learning methods in the field of medical image analysis. *International Journal of Image, Graphics and Signal Processing (IJIGSP)*, volume 13, pages 33–46, MECS Press, 2021. DOI: 10.5815/ijigsp.2021.04.03. (cited on page 87).

[Yan21] S.-H. Yang et al. Deep learning-assisted quantification of atomic dopants and defects in 2D materials. *Advanced Science*, volume 8, page 2101099, Wiley Online Library, 2021. DOI: 10.1002/advs.202101099. (cited on page 120).

[Yoo03] A. B. Yoo, M. A. Jette, and M. Grondona. Slurm: Simple linux utility for resource management. In *Proceedings of the JSSPP Workshop*, pages 44–60, Cambridge, USA, 2003. Springer. DOI: 10.1007/10968987_3. (cited on page 154).

[Zbo21] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*, 2021. (cited on pages 35, 86, 87, 98, 100, 169, and 170).

[Zha18] T. Zhao, Y. Yang, H. Niu, D. Wang, and Y. Chen. Comparing U-Net convolutional network with Mask R-CNN in the performances of pomegranate tree canopy segmentation. In *Proceedings of the SPIE*, volume 10780, pages 210–218, Bellingham, USA, 2018. (cited on page 121).

[Zha19] T. Zhang, G. Lin, J. Cai, T. Shen, C. Shen, and A. C. Kot. Decoupled spatial neural attention for weakly supervised semantic segmentation. *IEEE Transactions on Multimedia*, volume 21, pages 2930–2941, IEEE, 2019. DOI: 10.1109/TMM.2019.2914870. (cited on page 61).

[Zha22a] C. Zhang, C. Zhang, J. Song, J. S. K. Yi, K. Zhang, and I. S. Kweon. A survey on masked autoencoder for self-supervised learning in vision and beyond. *arXiv preprint arXiv:2208.00173*, 2022. (cited on page 88).

[Zha22b] X. Zhang et al. Seuneter: Channel attentive U-Net for instance segmentation of the cervical spine mri medical image. *Frontiers in Physiology*, volume 13, page 1081441, Frontiers Media SA, 2022. DOI: 10.3389/fphys.2022.1081441. (cited on page 121).

[Zhe22] S. Zheng, X. Li, M. Bi, Y. Wang, H. Liu, X. Feng, Y. Fan, and L. Shen. Contrastive learning-based adenoid hypertrophy grading network using nasoendoscopic image. In *Proceedings of the CBMS*, pages 377–382, Shenzhen, China, 2022. IEEE. DOI: 10.1109/CBMS55023.2022.00074. (cited on page 87).

[Zho16] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE CVPR*, pages 2921–2929, Las Vegas, USA, 2016. DOI: 10.1109/CVPR.2016.319. (cited on page 44).

[Zho18] Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang. Unet++: A nested U-Net architecture for medical image segmentation. In *Proceedings of the DLMIA Workshops*, pages 3–11, Granada, Spain, 2018. Springer. DOI: 10.1007/978-3-030-00889-5_1. (cited on page 121).

[Zhu19] W. Zhu et al. Anatomynet: deep learning for fast and fully automated whole-volume segmentation of head and neck anatomy. *Medical physics*, volume 46, pages 576–589, Wiley Online Library, 2019. DOI: 10.1002/mp.13300. (cited on page 120).