

DISSERTATION

Streamlining medical data annotation in deep learning through  
self-supervised approaches

Annotationsreduktion medizinischer Daten in Deep-Learning-  
Verfahren durch selbstüberwachte Lernansätze

zur Erlangung des akademischen Grades  
Doctor rerum medicinalium (Dr. rer. medic.)

vorgelegt der Medizinischen Fakultät  
Charité – Universitätsmedizin Berlin

von  
Benjamin Pascal Voigt

Erstbetreuung: Professor Dr. med. Dipl.-Phys. Frederick Klauschen

Datum der Promotion: 29. November 2024



---

## Table of contents

List of figures .....	ii
List of abbreviations.....	iii
Abstract .....	1
1 Introduction.....	4
2 Histopathological Image Data.....	10
2.1 Methods.....	11
2.2 Results .....	12
3 DNA Sequence Data .....	15
3.1 Methods.....	15
3.2 Results .....	17
4 Discussion and Outlook.....	19
5 Conclusions.....	22
Reference list.....	23
Statutory Declaration .....	28
Declaration of your own contribution to the publications.....	29
Printing copy(s) of the publication(s) .....	31
Curriculum Vitae .....	64
Publication list.....	66
Acknowledgments .....	67

**List of figures**

Figure 1: Pretraining and finetuning pipeline .....8  
Figure 2: Benefit distribution of different training methods..... 13  
Figure 3: Taxonomic domain classification pipeline for NGS reads..... 16

**List of abbreviations**

DNA	Deoxyribonucleic acid
EBM	Energy Based Model
GDC	Genomic Data Commons
NGS	Next Generation Sequencing

## Abstract

The continuous increase in available specimens and data is a catalyst for medical research and development and a prerequisite for personalized medicine. One way to effectively process the growing amount of data and derive new diagnostic tools or insights from it is through machine learning algorithms, particularly those based on artificial neural networks and summarized under the term deep learning. These algorithms have made significant progress in solving various problems in recent years. This is evident in image recognition and processing, where solutions based on neural networks promise to support, optimize, or even replace established medical procedures.

Most neural networks are currently trained using the supervised learning approach, which requires a large, annotated training dataset. Due to the expertise and legal restrictions in the annotation process, this poses a significant challenge and an enormous cost factor in the medical domain. This work demonstrates that self-supervised Learning methods implement an alternative learning paradigm that addresses this issue by drastically reducing the need for annotated training data. This effect is demonstrated using sequential and image data, two fundamental types of machine learning data.

Multiple state-of-the-art self-supervised learning methods are adapted to the characteristics of histopathological image data, compared under controlled conditions, and evaluated for their suitability in the domain. It is shown that the trained models, across different network architectures and configurations, positively impact performance in histopathological classification tasks, even with limited annotated data. The observations suggest that the learned features of such models are transferable to a certain extent within the domain, for example, from one tissue type to another. Sequential data demonstrates how a prototype for reference-free taxonomic classification of DNA sequences can be developed based on a self-supervised trained amino acid language model. It is shown that existing tools can be improved, and new research-relevant questions can be investigated using such models.

The results of the two case studies show that the adoption of the self-supervised Learning paradigm is of essential importance in the field of medicine. Models trained in this manner are expected to substitute existing ones and establish themselves as baseline models in various medical domains.

## Zusammenfassung

Der stetige Zuwachs an verfügbaren Bioproben und Daten ist ein Katalysator für die medizinische Forschung und Entwicklung und eine Voraussetzung für die personalisierte Medizin. Eine Möglichkeit, die wachsenden Datenmengen gewinnbringend zu verarbeiten und daraus neue diagnostische Werkzeuge oder Erkenntnisse zu gewinnen, sind Algorithmen des maschinellen Lernens, insbesondere solche, die unter dem Begriff Deep-Learning zusammengefasst werden und auf künstlichen neuronalen Netzwerken basieren. Diese Algorithmen haben in den vergangenen Jahren einen signifikanten Fortschritt bei der Lösung unterschiedlichster Probleme erzielt. Dies wird beispielsweise im Bereich der Bilderkennung und -verarbeitung deutlich, wo diese Ansätze versprechen, etablierte medizinische Verfahren zu unterstützen, zu optimieren oder sogar zu ersetzen.

Der überwiegende Teil neuronaler Netzwerke wird aktuell auf Basis von überwachtem Lernen trainiert, welches einen großen annotierten Trainingsdatensatz erfordert. In der medizinischen Domäne ist dies aufgrund der für den Annotationsprozess erforderlichen Fachkenntnisse und rechtlichen Einschränkungen eine große Herausforderung sowie ein enormer Kostenfaktor. In dieser Arbeit wird gezeigt, dass selbst-überwachte Methoden ein alternatives Lernparadigma umsetzen, welches dieser Problematik entgegenwirkt, da dies die Anzahl der benötigten annotierten Trainingsdaten drastisch reduziert. Anhand von sequenziellen Daten sowie Bilddaten, zwei elementaren Datentypen des maschinellen Lernens, wird dies belegt.

Verschiedene selbst-überwachte Lernmethoden, die dem aktuellen Stand der Forschung entsprechen, werden an die Eigenheiten histopathologischer Bilddaten angepasst, unter kontrollierten Bedingungen verglichen und für die Eignung in der Domäne bewertet. Es wird gezeigt, dass die trainierten Modelle über verschiedene Netzwerkarchitekturen und -konfigurationen hinweg eine positive Auswirkung auf die Leistung bei histopathologischen Klassifizierungsaufgaben haben. Die Beobachtungen deuten darauf hin, dass die erlernten Merkmale der Modelle bisher nur zu einem gewissen Grad auf andere Bereiche innerhalb der Domäne transferierbar sind, beispielsweise von einem Gewebetyp auf einen anderen. Für sequenzielle Daten wird gezeigt, wie basierend auf einem selbst-überwacht trainierten Amino-Säure-Sprachmodell ein Prototyp zur referenzfreien taxonomi-

schen Klassifizierung von DNS-Sequenzen entwickelt werden kann. Es wird demonstriert, dass sowohl existierende Werkzeuge mit Hilfe solcher Modelle verbessert als auch neue forschungsrelevante Fragestellungen untersucht werden können.

Die Ergebnisse der beiden Fallstudien zeigen, dass die Verbreitung des selbst-überwachten Lernparadigmas für die Medizin von essenzieller Bedeutung ist. Es wird erwartet, dass derartig trainierte Modelle existierende substituieren und sich als Basismodelle in verschiedenen medizinischen Bereichen etablieren werden.



## 1 Introduction

High-quality specimens and their associated clinical data are essential for biomedical research and are the foundation for rapid progress in personalized medicine. The continuous digitalization of medical processes and improvement of procedures accelerate the collection of such information, leading to growing amounts stored in associated biobanks and databases. Through this development, specimens often become data themselves. For instance, the digital transformation of pathology workflows results in the digital archiving of microscope slides. This increases the usability of these slides for research and development of supporting analytical tools, as digitized information is more reliably available, easily accessible, and effortlessly shareable than its physical counterpart. Likewise, improvements in the sequencing technique of biospecimens led to an explosion of genetic information available to researchers. The ability to process and analyze this bulk of data can be considered a crucial factor for further developing diagnostic and therapeutic methods.

One promising class of algorithms that can be used for this purpose and utilize the increasing amount of data belongs to machine learning, a subfield of artificial intelligence. Machine learning algorithms generate (probabilistic) computational models to perform specific tasks, usually classification or decision problems, without being explicitly programmed how to do so. Instead of operating on predefined rules or heuristics, such algorithms learn characteristic properties or underlying patterns to solve the task from data by processing examples. One particular group of those algorithms, well-known as deep learning, significantly pushed the performance boundaries in several problem fields, like natural language processing or natural image recognition, in the last decade [1]. The performance improvement provided by deep learning models was so disruptive that these quickly replaced existing computational models based on feature-engineering or system rules designed by human experts. Moreover, the methods have been promptly adapted by other fields of research and development to investigate domain-specific problems, including the medical domain [2]. So far, most deep learning models, especially outside the

machine learning community, are built upon neural networks<sup>1</sup> implementing the supervised learning paradigm. This training strategy uses a fixed predefined number of supervisory signals  $T$  providing information about the desired output when processing samples of a problem set  $X$ . Thus, a dataset  $D = \{(x, t) | x \in X, t \in T\}$  is mandatory when training a model using the supervised learning approach since each training example must be associated with a supervisory signal. The connected signal of a specific data point is often called its ground truth annotation or label.

An early successful application of neural networks, the recognition of zip codes [4], can serve as an example to illustrate this learning approach. During its training process, the model received images of single handwritten numbers along with the information about the number actually shown. In this case,  $X$  consists of all images, and the distinct label information describing the content (a number between 0 and 9) represents the set of possible supervisory signals  $T$ . Processing an input image through the model predicted likewise in a number between 0 and 9, which could be compared to the label information to verify the prediction. Based on this feedback loop, the model is optimized to maximize the correct predictions. Thus, the neural network learned to recognize digits without prior domain knowledge, e.g., handwriting properties, instead independently identifying characteristic features of the data to distinguish these numbers.

The supervised learning approach can be formalized, following the empirical risk minimization principle, as an iterative optimization problem over the model parameters  $w$  minimizing the expectation of a per-example loss function  $L(t, x, w)$  measuring the difference between the supervisory signal and the model prediction. Evidently, this learning strategy crucially depends on label information, consequently influencing the resulting model. Thus, the availability of huge, diverse datasets with high-quality ground truth annotations was a significant prerequisite for the success of algorithms implementing the paradigm [5, Chapter 1.2.2-1.2.4]. Moreover, these attributes of datasets remain critical to the performance of models trained using this approach. This results in a challenge for the medical domain. Even though the collected data is steadily increasing, the cost and effort to process this data into proper machine learning datasets are hardly feasible due to the

---

<sup>1</sup> Strictly speaking, such a network is an artificial neural network consisting of artificial neurons corresponding to a mathematical function rather than a biological circuit [3]. However, the term *artificial* is generally omitted for brevity in the machine learning community.

demanding nature of annotations, requiring expert knowledge, and the restrictive regulations. In pathology, for example, the existing datasets are either of limited size or designed for specific tasks, thereby failing to capture the full spectrum of tissue types, morphological changes, and disease characteristics. Hence, learning models incorporating features representing general domain patterns are somewhat challenging. Comparing the total size of publicly annotated datasets, frequently used [6–10], with the raw whole slide image research data available through the GDC Data Repository [11], there is a considerable difference in orders of magnitude (gigabyte versus terabyte). Consequently, unlocking the potential of using unannotated datasets is essential to exploit deep learning algorithms in the domain fully.

Using alternative learning paradigms is one approach to address the requirement for diverse and extensive annotations. In contrast, to a supervised learning approach, self-supervised learning methods enable a training process with unannotated samples,  $D = \{x|x \in X\}$ . This is because no external supervisory signal is required. Instead, such signals are specified within these methods by solving a particular internal task from which the term self-supervised is derived. Hence, learning from orders of magnitude more data becomes achievable, allowing the opportunity to recognize and understand patterns more nuanced, even of less frequent structures. For instance, a common technique today to train a language model with such a task is to mask certain input parts, e.g., words in a sentence, and train the neural network to infer appropriate tokens from the training data to fill these gaps., i.e., assign high probabilities to such tokens compared to other from a vocabulary. Note that the supervisory signal here is self-created by deliberately omitting information from the training data. This information can then be used to validate the predictions later. Therefore, no labeled data is needed. Predicting missing parts of the input has become, in fact, a standard technique of self-supervised learning and is called denoising autoencoder [5, Chapter 14.5]. One method for implementing this strategy is the Bidirectional Transformer (BERT) [12], which can learn hidden structures and patterns in sequential data this way, e.g., the underlying grammatical rules in natural language.

The application of denoising autoencoders based on the Transformer architecture has only recently been adapted for computer vision tasks [13]. Compared to a natural language processing task, where a finite vocabulary is usually given, image analysis tasks often have significantly larger solution spaces that quickly become computationally intrac-

table. Therefore, implementations of the denoising learning strategy are much more challenging. These initial attempts [13] are promising but have yet to be established in application domains outside the machine learning community. In contrast to the denoising learning strategy, a common approach in computer vision has been constructing the supervisory signal by maximizing the similarity in the embedding space (feature space) between the actual input and a slightly modified version [14–19]. Roughly speaking, the strategy of this approach is to learn that different versions of an image share the same features and to extract these features in the learning process. An analogy in human learning would be that one is shown different pictures of the same object without knowing its name and thus learns to recognize this object without being able to name it.

So far, no accepted unified theoretical approach formalizes the various self-supervised learning approaches. LeCun and Misra [20] suggest interpreting the methods within the energy-based model framework (EMB) [21]. A disadvantage of this interpretation is that purely supervised learning can also be formalized into the EMB framework and, thus, lacks differentiation [22]. Dubois *et al.* recently presented a narrower conceptual framework. The authors self-critically assess, however, that *"There are many limitations that should be addressed for a more realistic prescriptive framework for ISSL"* (invariant self-supervised learning) [23].

Models trained with a method implementing the self-supervised learning approach are not problem-related, i.e., they do not solve a specific task associated with its training data. Consequently, these models incorporate features reflecting more general patterns in the data produced by method-related tasks that are universal and data-independent. However, these models, known as encoders, can be adapted later in time to a specific task, e.g., a classification (predicting a class of an object) or regression problem (predicting the next token in a sequence). For this purpose, only a few annotated data points are necessary to adjust the generic encoder features to the concrete task and train a small prediction head mapping the features into problem space. These process steps are often called *pertaining* (training an encoder with a self-supervised learning task to extract generic features) and *finetuning* (adjusting encoder models to a specific task using supervised learning with a small amount of annotated samples), see Figure 1.

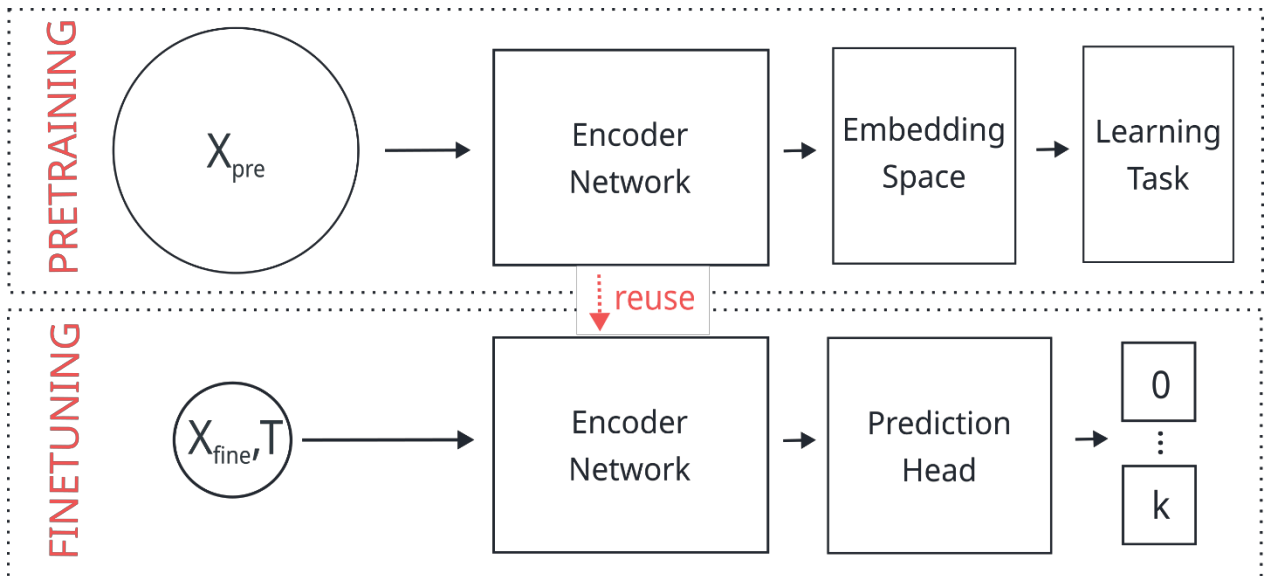


Figure 1: Pretraining and finetuning pipeline - During pretraining, a network is trained to encode the input into an embedding space representing the feature space. This training is performed with an unannotated dataset  $X_{pre}$  solving a self-supervised learning task. The resulting encoder network can later be adjusted in a finetuning stage to solve a particular task utilizing only a small number of annotated examples  $X_{fine}$ . Here a classification task into  $k+1$  different classes is shown as an example [author illustration].

In the field of natural language processing and natural image classification, working with sequential [12, 24–27] and image data [13–19], respectively, finetuned encoder models have demonstrated significantly better performance compared to models trained solely in a supervised manner. In fact, the self-supervised approach decouples the learning process of generic features inside a domain from actual problems, opening up the possibility of making more versatile models publicly available. An immediate consequence could be significantly saving computational resources since complex models for tasks building upon such feature space do not have to be thoroughly trained repeatedly. Furthermore, it allows the (research) community to train models for problems with usually insufficient data or to use more complex models than the dataset would allow, e.g., due to overfitting threads.

This thesis summarizes contributions to training deep learning algorithms with unannotated data in the medical domain. For this purpose, the application and investigation of the self-supervised learning paradigm are considered for the two fundamental data types, sequential and image data, using deoxyribonucleic acid (DNA) sequences and histopathological image data as examples. The contributions are based on the following publications:

**Voigt, B.**, Fischer, O., Schilling, B., Krumnow, C., & Herta, C. (2023). Investigation of semi-and self-supervised learning methods in the histopathological domain. *Journal of Pathology Informatics*, 100305.

**Voigt, B.\***, Fischer, O.\*, Krumnow, C., Herta, C., & Dabrowski, P. W. (2021). NGS read classification using AI. *Plos one*, 16(12), e0261548.  
\* equal contribution

## 2 Histopathological Image Data

Since semi- and self-supervised learning methods has the potential to drastically reduce the effort of data annotation without limiting the predictive performance of the models. This development is essential in application areas such as computational pathology, where large amounts of data are available, but open, high-quality annotated datasets are rare or non-existent. Hence, training a well-generalizing classifier is challenging. Training methods utilizing non-annotated data have, accordingly, strong potential. The first attempts have already been made to transfer such methods to histopathological data and explore their performance in this domain. While most follow the semi-supervised [28–30] or contrastive self-supervised learning approach [31–33], fewer studies have implemented the non-contrast learning principle so far, e.g., [34, 35].

Contrastive self-supervised learning methods focus on creating useful representations by contrasting positive and negative pairs of samples from the training data. These methods aim to maximize the similarity between positive pairs while minimizing the similarity between negative pairs. In other words, they encourage the model to distinguish between different samples in a learned embedding space. Non-contrastive self-supervised learning methods, on the other hand, do not rely on explicit positive and negative pairs. Instead, they employ a self-supervised task that involves constructing multiple views of the same sample and training the model to predict one view from another. The key idea is to learn invariant representations that capture important information about the data.

Even though the initial transfer studies suggest histological domain encoders as a benefit on in-domain classification and segmentation problems compared to non-domain feature extractors (trained on natural images) or purely supervised training, the learned embeddings have been limited studied, especially across different methods. An exception is the work of Ciga et al. [36], who performed a deeper analysis of SimCLR [15] across different datasets. In this work, we developed a framework to study encoder models under uniform conditions to investigate different pretraining approaches' capabilities to learn effective histopathological domain embeddings. To perform a cross-sectional study, we investigated advanced techniques that utilize different learning paradigms for training encoder models. Specifically, we evaluated PAWS [37] as a semi-supervised learning method, SimCLR as a contrastive self-supervised learning method, and SimSiam [18] as a non-contrastive self-supervised learning method.

## 2.1 Methods

For the investigation and comparison, we intended to produce each method's best encoder in the new data domain, keeping the approaches of the different pretraining methods as close to the initial proposed implementation as possible. However, to maintain internal comparability throughout the experiments, we defined an experiment protocol for the pretraining, restricting the encoder networks to an equal baseline architecture and unifying some training settings. In particular, we ensured that all random operations, such as weight initialization or training data sampling, were standardized across the pretraining runs. Thus, the overall encoder training environment was nearly identical (e.g., the embedding space varied across the methods) allowing encoders to be compared later, even if trained method-specifically.

We performed several experiments on multiple in-domain classification tasks based on these structurally aligned encoders. Likewise, we created another protocol that defined the environment for the finetuning processes. Yet, this protocol was much stricter, guaranteeing that all experimental runs performed similarly. We acknowledge that this approach may lead to slightly lower performance on a given task than individually possible for each classifier, particularly compared to published state-of-the-art results. However, optimizing each experiment individually makes it nearly impossible to disentangle and compare the impact of the pretrained models. On the other hand, the fixed finetune process prevents individual training effects and enhances the comparability of results across the different learning methods, enabling the inference of learned encoder embeddings. Accordingly, we consider this approach justified. Both protocols combined form the complete experimental pipeline, i.e., the designated framework.

Utilizing the framework as a first experiment, we studied the methods regarding possible performance fluctuations. In critical domains like medical applications, it is essential that (deep learning) algorithms yield consistent results. Similarly, in medical research, traceability and reproducibility are crucial aspects. Thus, we trained 5 encoders for each investigated method by only varying the experiment random seeds, i.e., changing the initial network weights. After, we finetuned these encoders on the corresponding classification problem of the dataset used in pretraining five times, reusing the same seeds. Accord-



ingly, this resulted in 25 data points on the downstream task per method for each measured performance metric. Finally, standard deviations were examined to determine whether performance fluctuations were due to pretraining or finetuning.

An assumed benefit of training a classification model based on a powerful encoder is that the annotated training data can be drastically reduced since the general patterns only need to be tuned for the specific task. We examined this assumption quantitatively while still looking for differences between the pretraining approaches. Employing the protocols, we trained multiple encoder variations per method on the PCam [6] and NCT-CRC-HE-100K [7] histopathological patch datasets. We finetuned each across both corresponding classification problems using different subsets of the training data (100%, 8%, 0.8%, and 0.2%) to simulate settings with sparse annotated data available. In addition, the encoders were evaluated on the Lizard [9] dataset representing a tissue type never utilized in pretraining and, hence, were unknown to all encoder models. Pretraining and finetuning experiment runs were executed with several configuration settings to explore different aspects that may influence the performance, e.g., the encoder's network complexity or the prediction head's complexity while finetuning. Overall, we trained 288 classifiers per method. Further, we computed method-wise a fully supervised complement for each experiment setting to calculate an absolute benefit when built upon an encoder by subtracting the performance metric values from the complementary runs. Based on this comprehensive result data, not only the initial assumption could be examined with statistical significance, but also other essential aspects influencing the overall performance. Moreover, through examining experiment runs in which the encoder training data differed from that of the finetuning task, it was possible to conclude about the learned embedding quality and properties, e.g., feature transferability, of histopathological domain encoder.

In addition, we qualitatively investigated specific classifier with explainable AI techniques to study encoder embeddings in more detail. Therefore, using one model across each method, we created activation maps for randomly chosen samples using Grad-Cam [38] to observe if the decision-relevant image content represented domain features or was more of a general nature.

## 2.2 Results

The results from our experimental analysis demonstrate that pretraining generally provides a positive impact, as indicated by the average benefit (Figure 2a) and absolute

performance when compared to exclusively supervised training. This effect was more substantial for complex network architectures and remained consistent even when reducing the amount of training data. Notably, it was observed that fewer annotated data samples were necessary to achieve comparable performance when building upon a domain encoder. Additionally, the evidence suggests that these models are capable of learning domain-specific features that are applicable across different tissue types. However, we also noted that the transferability of these features was limited and depended on the encoder training and downstream task data divergence (as depicted in Figure 2b). Despite this limitation, based on the overall findings, it is recommended to construct a neural network classifier using a histopathological domain encoder model, especially if there is less annotated data available.

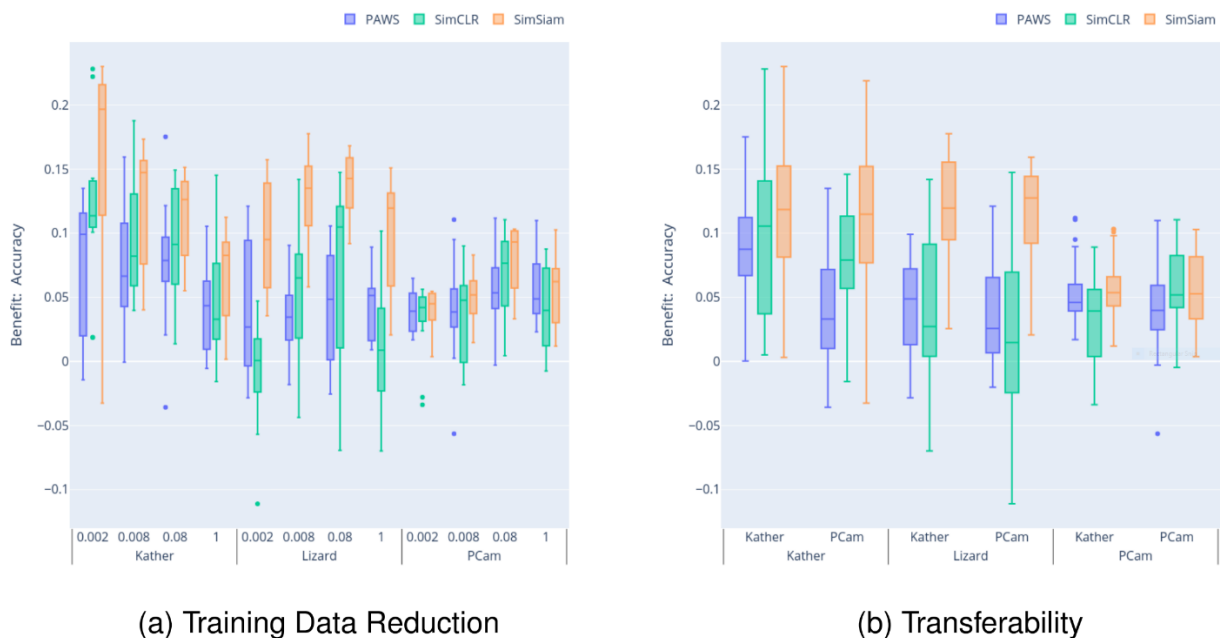


Figure 2: Benefit distribution of different training methods - Accuracy benefit distribution is shown for successively decreasing finetuning training data (a) and encoder transferability, i.e., encoder training data distinguishes from finetuning data (b); graphics were taken from [39].

It should be noted that using a unified evaluation strategy has restrained the performance of all methods studied. However, it has facilitated the comparison of the encoders and the examined learning strategies, which was the key objective of this study. We expect each method to improve relatively if individually optimized for a downstream task, i.e., not constrained by the finetuning protocol. Regarding the methods comparison, PAWS achieved mainly lower values than the other methods. Hence, the additional information the method received due to its semi-supervised learning approach showed no advantage

in the experimental setup of this work. SimCLR, a representative of contrastive self-supervised learning, exhibited the most reliable performance across all experimental configurations. Regardless of the encoder training data and network architecture, its results revealed the least amount of fluctuations. Thus, it can be inferred that this methodology generates the most stable embeddings for the domain inside the comparison group. On the other hand, SimSiam, as the non-contrastive self-supervised learning approach, was less sensitive to weight initialization if the encoder was finetuned. Its evaluation metrics were the highest and obtained the greatest relative benefit overall. However, training a high-performance encoder was most challenging with this method, and the encoders also provided significantly more variable results than the SimCLR approach. The findings imply that SimSiam may demand more customized tuning than the fixed experimental design permits. Hence, testing SimSiam could be a favorable complement or alternative to SimCLR, which appears the preferred choice for obtaining consistent results in the histopathological domain.

### 3 DNA Sequence Data

For processing medical sequential data, DNA sequences (metagenomic datasets) are used as an example in this work. The sequencing process of metagenomics generates extensive nonspecific data due to its nature. Thus, the relevant sequences must be filtered out to prepare the data for further analysis, e.g., a diagnostic process. The standard approach for cleansing metagenomic data involves comparing it to reference databases. As an established method, recent optimization of implementations has efficiently detected pathogens in the utilized databases. However, a common issue when analyzing a metagenomic patient sample is the inability to identify pathogen sequences. Lennon & Locey [40] estimated that approximately  $10^{14}$  bacterial taxa exist, but only around  $10^6$  have been identified, while Anthony *et al.* [41] suggested the presence of over  $3 \times 10^5$  undetected mammalian viruses. Determining whether the absence of pathogen evidence in the data is due to the pathogen's nonexistence in the genome or its unavailability in the database, and thus being impossible to classify the sequences, presents a challenge.

Previous studies have demonstrated the effectiveness of machine learning in addressing this issue. For instance, Deneke *et al.* [42] utilized Random Forests to accurately anticipate the occurrence of human pathogenic organism sequences in metagenomic datasets. Although various methods have employed machine learning to enhance the parameters of existing tools [43, 44], Deneke *et al.* [42] research is the only known to us, utilizing machine learning to identify microbial sequences without depending on reference sequences. We aspire to extend the understanding of such approaches' usefulness by applying deep learning language models to this problem. Using a self-supervised protein language model, we have developed a proof-of-concept tool to detect novel pathogens in metagenomic datasets. This is accomplished by a taxonomic classification of proteins encoded by sequences from these datasets. We thereby demonstrate that such a model developed for a different purpose can be adjusted to create new diagnostic tools and investigate other research questions by intra-domain transferring its learned features.

#### 3.1 Methods

We developed a prototype pipeline for classifying Next Generation Sequencing (NGS) reads into the taxonomic domains of *virus*, *bacteria*, or *human/mammalian*. The pipeline

is founded on the self-supervised trained protein language model ProtTrans [45], employed as an encoder. Multiple data processing and classification tasks are performed for taxonomic prediction. To solve these tasks, we built upon the knowledge ProtTrans learned about the protein domain. Thus, the initial model was finetuned to individual pipeline problems using customized datasets created from amino acid sequences of the UniProtKB/Swiss-Prot database [46, 47] and DNA reference sequences of the RefSeq database [48]. Figure 3 provides a conceptual outline of the entire taxonomic classification pipeline.

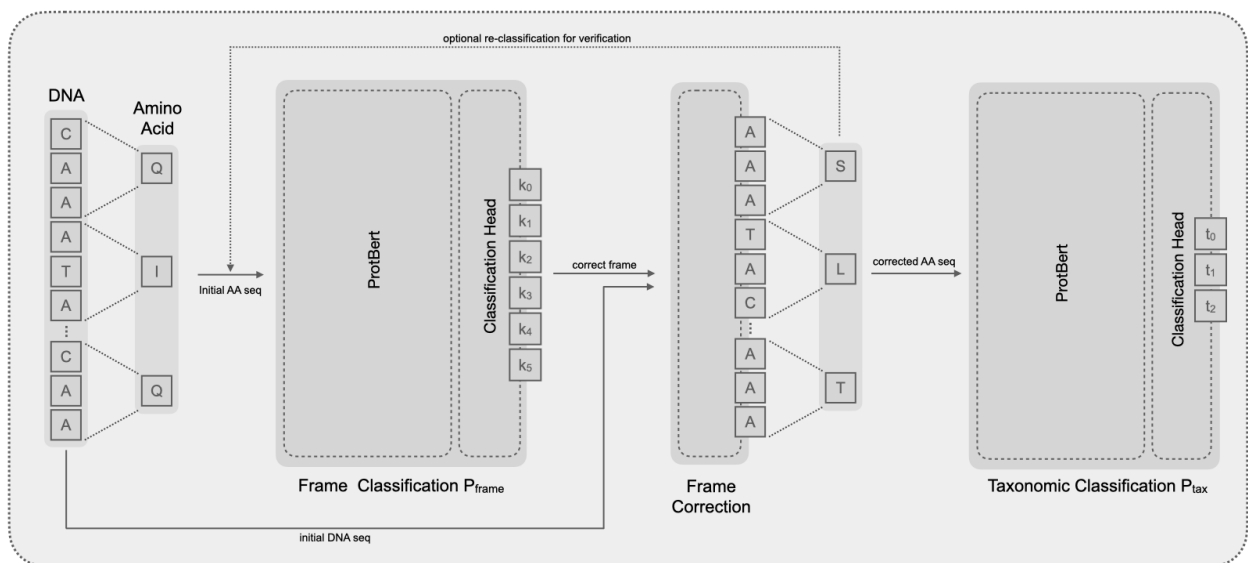


Figure 3: Taxonomic domain classification pipeline for NGS reads - The DNA sequences are translated into an amino acid sequence. A first model  $P_{\text{frame}}$  predicts if the amino acid sequence is correct translated (on-frame) or if the initial DNA sequence needed to be shifted before translation (off-frame). After the correct translation, which can be verified by the using  $P_{\text{frame}}$  again (dotted arrow), the amino acid sequence can be taxonomic classified by the second model  $P_{\text{tax}}$ . is performed a second; Graphic is taken from [49].

The pipeline input consisted of NGS reads and accepted nucleotide sequences up to 300 base pairs each. One restriction for this proof-of-concept pipeline was that all reads lie within a coding sequence, i.e., coding to a protein. Considering that the ProtTrans model is trained on amino acids, it was necessary to translate the DNA sequences into an amino acid sequence, which was implemented using biopython [50] as an established tool for this task. Thus, the effective model input length is reduced to 100 or less since three nucleotide base pairs are coding one protein. Translating these fragments into a proper coding sequence poses a significant challenge due to the absence of distinct start or stop

codons, which typically serve as indicators of the correct reading frame. Hence, this translation process can be regarded as a non-trivial problem. However, this problem can be represented as a classification task comprising six classes: the correct reading frame (on-frame) and five possible offset frames. Therefore, ProtTrans was finetuned to a first classification model returning a probability distribution over the frames classes for a given amino acid sequence. Figure 3 illustrates this pipeline step as  $P_{\text{frame}}$ .

After determining an amino acid sequence's most likely reading frame, the initial nucleotide sequence was shifted accordingly and translated into the correct amino acid sequence used for further processing. Note that for minor additional computational resources,  $P_{\text{frame}}$  could be reused to validate the new translation, i.e., the most likely prediction of the corrected sequence should be on-frame. In the last pipeline step, another finetuned ProtTrans model  $P_{\text{tax}}$  classified the on-frame amino acid sequence into taxonomic domains *virus*, *bacteria*, or *human/mammalian*.

The final models were evaluated on a 10% split of the customized datasets reserved for internal testing. Further, the applicability of the trained models and the complete pipeline was tested by classifying real-world reads. Therefore, data from two metagenomic sequencing projects available within the SRA [51] were classified, i.e., human skin metagenomic study (SRR7188139) and a swine feces metagenome (ERR3013343). Compared to the training derived from curated reference databases, these test reads contained sequencing errors. Moreover, the frame classification was evaluated against existing tools determining the correct translation frame directly from short NGS reads: frag-GeneScan [52] and CNN-MGP [53].

### 3.2 Results

Benchmarking the frame classification  $P_{\text{frame}}$  on the internal test set yielded an overall accuracy of 0.98, significantly outperforming both CNN-MGP (0.34 accuracy) and frag-GeneScan (0.59 accuracy) within the experimental setting. Also, reexamining the shifted sequences with the classifier showed the expected behavior of almost all sequences moved into the proper frame. For the taxonomic classification model  $P_{\text{tax}}$ , an accuracy of 0.91 on the internal test data was measured. Upon closer examination of the inner-class accuracies, we found that most reads predicted as bacterial (0.94) were correctly classified. In contrast, only 0.88 of sequences classified as viral were of viral origin, with 0.8 being mammalian. Aligned patterns were observed for mammalian classified reads. The

classifier faced significant challenges in differentiating between mammalian and viral reads, as evident from the higher frequency of mammalian reads (0.92) compared to viral reads (0.06). This behavior can be explained to retroviral sequences within the human genome.

However, the exemplary analysis of real metagenomic sequencing data was more challenging for the pipeline, likely due to its noisy nature. The swine feces metagenomic dataset could achieve a taxonomic domain classification precision of 0.87. This could be improved to 0.90 if cleaning the test data of erroneous sequences, e.g., discarding reads containing a stop codon in every possible frame, which is impossible within a coding sequence. In contrast, the human skin metagenome classification had a precision of 0.62. This apparent drop resulted mainly from bacterial reads being wrongly predicted as viral. Yet, the extent to which this performance difference between the test datasets is due to a distributional shift requires clarification through further investigation, which was beyond the scope of this proof-of-concept study.

Altogether, the findings suggest that a sufficiently trained contemporary language model can acquire the necessary domain knowledge to develop a domain-level taxonomic classifier using short amino acid sequence fragments of an organism's proteins without depending on a reference database. Moreover, the frame classification task results demonstrated such language models' ability to determine the frame of a short DNA sequence within an open reading frame without prior knowledge of the reference sequence. This approach could potentially serve as an alternative to the conventional practice of comparing all possible translations with a protein sequence database, provided that the existing restriction of reads being within coding sequences can be removed without any performance loss.

## 4 Discussion and Outlook

It was shown that models classifying histopathological images based on domain-specific encoders have performance advantages over models trained exclusively with supervised learning, especially if only a few training examples are available. This confirms previous research results on individual self-supervised learning methods in this domain [31–36]. However, comparing methods within the controlled experiment setting highlighted differences in performance and methodological domain challenges. The findings suggest that these are due to both learning approaches and the peculiarities of histopathology data compared to natural images.

Training with the SimCLR, representative of contrastive self-supervised learning methods, showed the most stable learning behavior based on optimized hyperparameters and consistently produced meaningful features leading to similar results on a downstream task. These observations are consistent with a recent theoretical investigation of such methods. Pokle *et al.* [54] argue that self-supervised training with the support of negative pairs converges more stably to a local minimum corresponding to a valuable representation of the training data features.

In contrast, for non-contrastive self-supervised learning methods, Pokle *et al.* [54] defines certain initial criteria that must be satisfied in order to converge to a meaningful representation. In general, non-contrastive methods tend to quickly converge to poor local minima if these criteria are not met, e.g., optimized hyperparameter settings, integration of weight normalization, or a network warm-up. This may explain the initial problems observed when adapting the SimSiam method to the histopathological domain. Although a non-collapsing training process was observable, the SimSiam encoders performed poorly on the downstream tasks during the initial method exploration. Data and weight normalization as well as optimization of hyperparameters and augmentation strategies fixed this issue. So, this is also aligned with the proposition of Pokle *et al.* [54]. It can be inferred that non-contrastive self-supervised learning methods have sensitive learning processes. A deeper investigation will be necessary to identify optimal factors for the respective medical training data to generate high-performance encoders in the domain.

Another relevant study finding concerns the domain-specific scope of encoder models, i.e., histopathological feature transferability. The results indicate that the usefulness of an encoder correlates with the similarity of its training data and the downstream data. This



effect is generally expected, as the usefulness of learned features decreases with the distributional shift of the data. Still, it is unexpectedly prominent in the homogeneous – compared to the variety of natural images – histopathology data. Thus, it cannot be concluded with certainty to what extent the learned features have general domain-specific validity and usefulness. However, since the study investigates the feature transferability only on three very different patch datasets, this aspect must be considered in a differentiated manner and, above all, analyzed further. Training neural networks through self-supervised learning methods allow unrestricted use of non-annotated data. This potential has not yet been fully exploited in the present study. Recent work on histopathological encoder models extensively trained on larger amounts of TCGA whole slide image data, such as Ciga *et al.* [36] and Chen *et al.* [55], shows significantly better transfer performance, i.e., achieving good performance results on different domain-specific classification and segmentation problems. This emphasizes the correlation between the encoder training data and the applicability of its features within the domain spectrum. Compiling optimized encoder training datasets could be an essential aspect of general histopathological base models and, accordingly, another research topic. Stacke *et al.* [56] reached similar conclusions in their recent work analyzing the SimCLR method in the domain.

The case study on sequential medical data details the successful development of an analysis tool that utilizes a self-supervised trained language model for the taxonomic domain-level classification of short protein sections. Notably, this tool demonstrates promising results as a proof-of-concept in the absence of matching sequences with a reference database, which is the current standard practice. Additionally, it has been demonstrated that short DNA sequences can be frame-predicted using a language model without prior knowledge of the reference sequence or the typical comparison of a six-frame translation with a protein sequence database. This reference-free recognition suggests that self-supervised trained language models can decipher the underlying grammar of nature, as supported by parallel advancements in the field, most notably DeepMind's AlphaFold model [57, 58], which predicts protein structure based on amino acid sequence and has become a research database in its own right.

Due to the models' capability of reference-free recognition, diverse application possibilities arise for the tool. For example, it enables further investigation of previously unknown sequences in an unclassified read set that typically remains after analysis of meta-

genomic NGS data. With the present model, this would be possible for the potential presence of bacterial and viral proteins, thus unknown organisms. Moreover, it can assist in analyzing metaproteomic experiments by classifying peptides at a high level. This classification can aid protein sequence assemblers, like PASS [59], by reducing the number of sequences and minimizing the risk of misleading overlaps. Furthermore, frame classification can be decoupled and integrated into other algorithms to make them more efficient by eliminating matching and translation operations employed for frame identification steps. As another example, this classifier could detect novel frame-shift mutations in unknown genomes without needing a high-quality reference sequence.

The ability to learn hidden structures from (unannotated) data and the emerging possibilities that arise from this, such as reference-free recognition, rapidly drive the development of such models in the medical field. Thus, improved protein language models have been developed in the past two years [60, 61], and models based entirely on DNA sequence data have been published [62]. Related work on taxonomic classification, which builds on DNA language models, has already emerged. For example, Mock et al. [63] presents their BERTax model for the taxonomic classification of DNA sequences at the domain and phylum levels.

Transformer language models have been utilized in novel medical applications, showcasing the potential of self-supervised learning in the field. The taxonomic and frame classification are two examples that demonstrate the value of this approach in 2021. However, the classifiers are limited by requiring the DNA sequence to be within an open reading frame. Resolving this constraint is necessary to advance the tool from a proof-of-concept to productive status. One solution could be to build upon a DNA language model instead of a protein one. Another limitation is the maximum sequence length that the models can process. The sequence length is due to the available computational capacity during development and could be modified by a re-training with more resources. A fundamental – but more general – challenge for future development and research is making the knowledge incorporated in these language models accessible and comprehensible to facilitate learning from it, especially in the medical domain.

## 5 Conclusions

In this thesis, training neural networks with self-supervised learning methods was investigated through two case studies on two elementary data types, sequential and image data, to evaluate the usefulness of such training methods for the medical domain.

Different state-of-the-art training methods were compared in a controlled experiment regarding the image data. It could be shown that all methods have an advantage over purely supervised training and that encoders, respectively their features, are transferable within the domain to a certain degree. After assessing the individual methods, a recommendation for usefulness in the histopathological context could be given. In the case of sequential data, it could be shown how an application with diagnostic and scientific relevance was (further) developed based on a protein language model. The implemented proof-of-concept analysis tool taxonomic classifies (domain-level) short amino acid segments without relying on a reference database.

Even though these methods are predominantly novel and proportionally lack a proven theoretical framework, their research potential and feasible performance improvements must be addressed even for this critical application domain. In the context of rapidly growing data volumes in the medical sector, self-supervised deep learning methods can catalyze the further development of therapeutic and diagnostic procedures.

## Reference list

1. LeCun Y, Bengio Y, Hinton G (2015) Deep Learning. *Nature* 521:436–444
2. Esteva A, Robicquet A, Ramsundar B, Kuleshov V, DePristo M, Chou K, Cui C, Corrado G, Thrun S, Dean J (2019) A guide to deep learning in healthcare. *Nat Med* 25:24–29
3. McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. *Bull Math Biophys* 5:115–133
4. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD (1989) Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Comput* 1:541–551
5. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT press
6. Bejnordi BE, Veta M, Van Diest PJ, et al (2017) Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. *Jama* 318:2199–2210
7. Kather JN, Halama N, Marx A (2018) 100,000 histological images of human colorectal cancer and healthy tissue. <https://doi.org/10.5281/zenodo.1214456>
8. Sirinukunwattana K, Snead DR, Rajpoot NM (2015) A stochastic polygons model for glandular structures in colon histology images. *IEEE Trans Med Imaging* 34:2366–2378
9. Graham S, Jahanifar M, Azam A, et al (2021) Lizard: A Large-Scale Dataset for Colonic Nuclear Instance Segmentation and Classification. In: *Proc. IEEE CVF Int. Conf. Comput. Vis.* pp 684–693
10. Graham S, Chen H, Gamper J, Dou Q, Heng P-A, Snead D, Tsang YW, Rajpoot N (2019) MILD-Net: Minimal information loss dilated network for gland instance segmentation in colon histology images. *Med Image Anal* 52:199–211
11. National Cancer Institute Genomic Data Commons Data Portal.
12. Devlin J, Chang M-W, Lee K, Toutanova K (2018) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. <https://doi.org/10.48550/ARXIV.1810.04805>
13. Dosovitskiy A, Beyer L, Kolesnikov A, et al (2020) An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv Prepr. ArXiv201011929*
14. He K, Fan H, Wu Y, Xie S, Girshick RB (2019) Momentum Contrast for Unsupervised Visual Representation Learning. 2020 IEEE CVF Conf Comput Vis Pattern Recognit CVPR 9726–9735

15. Chen T, Kornblith S, Norouzi M, Hinton G (2020) A Simple Framework for Contrastive Learning of Visual Representations. In: *Int. Conf. Mach. Learn.* PMLR, pp 1597–1607
16. Caron M, Misra I, Mairal J, Goyal P, Bojanowski P, Joulin A (2020) Unsupervised learning of visual features by contrasting cluster assignments. *Adv Neural Inf Process Syst* 33:9912–9924
17. Grill J-B, Strub F, Altché F, et al (2020) Bootstrap your own latent—a new approach to self-supervised learning. *Adv Neural Inf Process Syst* 33:21271–21284
18. Chen X, He K (2020) Exploring Simple Siamese Representation Learning. 2021 *IEEECVF Conf Comput Vis Pattern Recognit CVPR* 15745–15753
19. Zbontar J, Jing L, Misra I, LeCun Y, Deny S (2021) Barlow Twins: Self-Supervised Learning via Redundancy Reduction. *Int. Conf. Mach. Learn.*
20. LeCun Y, Misra I (2021) Self-supervised learning: The dark matter of intelligence. <https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence>. Accessed 8 Feb 2023
21. Lecun Y, Chopra S, Hadsell R, Ranzato MA, Huang FJ (2006) A tutorial on energy-based learning. *Predict. Struct. Data*
22. Grathwohl W, Wang K-C, Jacobsen J-H, Duvenaud D, Norouzi M, Swersky K (2020) Your classifier is secretly an energy based model and you should treat it like one. *Int. Conf. Learn. Represent.*
23. Dubois Y, Ermon S, Hashimoto TB, Liang PS (2022) Improving self-supervised learning by characterizing idealized representations. *Adv Neural Inf Process Syst* 35:11279–11296
24. Radford A, Narasimhan K, Salimans T, Sutskever I, others (2018) Improving language understanding by generative pre-training.
25. Brown T, Mann B, Ryder N, et al (2020) Language models are few-shot learners. *Adv Neural Inf Process Syst* 33:1877–1901
26. Thoppilan R, De Freitas D, Hall J, et al (2022) Lamda: Language models for dialog applications. *ArXiv Prepr. ArXiv220108239*
27. Chowdhery A, Narang S, Devlin J, et al (2022) Palm: Scaling language modeling with pathways. *ArXiv Prepr. ArXiv220402311*
28. Marini N, Otálora S, Müller H, Atzori M (2021) Semi-supervised training of deep convolutional neural networks with heterogeneous data and few local annotations: An experiment on prostate histopathology image classification. *Med Image Anal* 73:102165
29. Liu K, Liu Z, Liu S (2022) Semi-supervised breast histopathological image classification with self-training based on non-linear distance metric. *IET Image Process.*

30. Wu H, Wang Z, Song Y, Yang L, Qin J (2022) Cross-Patch Dense Contrastive Learning for Semi-Supervised Segmentation of Cellular Nuclei in Histopathologic Images. In: Proc. IEEE CVF Conf. Comput. Vis. Pattern Recognit. CVPR. pp 11666–11675
31. Srinidhi CL, Martel AL (2021) Improving Self-Supervised Learning With Hardness-Aware Dynamic Curriculum Learning: An Application to Digital Pathology. In: Proc. IEEE CVF Int. Conf. Comput. Vis. ICCV Workshop. pp 562–571
32. Ozen Y, Aksoy S, Kösemehmetoğlu K, Önder S, Üner A (2021) Self-Supervised Learning with Graph Neural Networks for Region of Interest Retrieval in Histopathology. In: 2020 25th Int. Conf. Pattern Recognit. ICPR. pp 6329–6334
33. Gong R, Wang L, Wang J, Ge B, Yu H, Shi J (2022) Self-Distilled Supervised Contrastive Learning for diagnosis of breast cancers with histopathological images. *Comput Biol Med* 146:105641
34. Li T, Feng M, Wang Y, Xu K (2021) Whole Slide Images Based Cervical Cancer Classification Using Self-supervised Learning and Multiple Instance Learning. In: 2021 IEEE 2nd Int. Conf. Big Data Artif. Intell. Internet Things Eng. ICBAIE. pp 192–195
35. Liu Q, Louis PC, Lu Y, et al (2021) SimTriplet: Simple Triplet Representation Learning with a Single GPU. *ArXiv Prepr. ArXiv210305585*
36. Ciga O, Xu T, Martel AL (2022) Self supervised contrastive learning for digital histopathology. *Mach Learn Appl* 7:100198
37. Assran M, Caron M, Misra I, Bojanowski P, Joulin A, Ballas N, Rabbat M (2021) Semi-Supervised Learning of Visual Features by Non-Parametrically Predicting View Assignments with Support Samples. *ArXiv Prepr. ArXiv210413963*
38. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D (2017) Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proc. IEEE Int. Conf. Comput. Vis. pp 618–626
39. Voigt B, Fischer O, Schilling B, Krumnow C, Herta C (2023) Investigation of semi- and self-supervised learning methods in the histopathological domain. *J Pathol Inform* 100305
40. Lennon JT, Locey KJ (2020) More support for Earth’s massive microbiome. *Biol Direct* 15:5
41. Anthony SJ, Epstein JH, Murray KA, et al (2013) A strategy to estimate unknown viral diversity in mammals. *mBio* 4:e00598-00513
42. Deneke C, Rentzsch R, Renard BY (2017) PaPrBaG: A machine learning approach for the detection of novel pathogens from NGS data. *Sci Rep* 7:39194
43. Vilne B, Meistere I, Grantina-Ievina L, Kibilds J (2019) Machine Learning Approaches for Epidemiological Investigations of Food-Borne Disease Outbreaks. *Front Microbiol* 10:1722

44. Sedlar K, Kupkova K, Provaznik I (2017) Bioinformatics strategies for taxonomy independent binning and visualization of sequences in shotgun metagenomics. *Comput Struct Biotechnol J* 15:48–55
45. Elnaggar A, Heinzinger M, Dallago C, et al (2020) ProtTrans: Towards Cracking the Language of Life's Code Through Self-Supervised Deep Learning and High Performance Computing. *bioRxiv*. <https://doi.org/10.1101/2020.07.12.199554>
46. TheUniProtConsortium (2018) UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Res* 47:D506–D515
47. TheUniProtConsortium (2020) UniProtKB/Swiss-Prot.
48. O'Leary NA, Wright MW, Brister JR, et al (2016) Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. *Nucleic Acids Res* 44:D733–745
49. Voigt B, Fischer O, Krumnow C, Herta C, Dabrowski PW (2021) NGS read classification using AI. *PLOS ONE* 16:1–13
50. Cock PJ, Antao T, Chang JT, et al (2009) Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* 25:1422–1423
51. Health NI of (2020) Sequence Read Archive (SRA). <https://www.ncbi.nlm.nih.gov/sra/>. Accessed 27 Nov 2020
52. Rho M, Tang H, Ye Y (2010) FragGeneScan: predicting genes in short and error-prone reads. *Nucleic Acids Res* 38:e191
53. Al-Ajlan A, El Allali A (2019) CNN-MGP: Convolutional Neural Networks for Metagenomics Gene Prediction. *Interdiscip Sci* 11:628–635
54. Pokle A, Tian J, Li Y, Risteski A (2022) Contrasting the landscape of contrastive and non-contrastive learning. In: *Int. Conf. Artif. Intell. Stat.* PMLR, pp 8592–8618
55. Chen RJ, Chen C, Li Y, Chen TY, Trister AD, Krishnan RG, Mahmood F (2022) Scaling vision transformers to gigapixel images via hierarchical self-supervised learning. In: *Proc. IEEE CVF Conf. Comput. Vis. Pattern Recognit.* pp 16144–16155
56. Stacke K, Unger J, Lundström C, Eilertsen G (2022) Learning Representations with Contrastive Self-Supervised Learning for Histopathology Applications. *J. Mach. Learn. Biomed. Imaging* 1:
57. Senior AW, Evans R, Jumper J, et al (2020) Improved protein structure prediction using potentials from deep learning. *Nature* 577:706–710
58. Jumper J, Evans R, Pritzel A, et al (2021) Highly accurate protein structure prediction with AlphaFold. *Nature* 596:583–589
59. Warren RL, Sutton GG, Jones SJ, Holt RA (2007) Assembling millions of short DNA sequences using SSAKE. *Bioinformatics* 23:500–501

60. Lin Z, Akin H, Rao R, et al (2023) Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science* 379:1123–1130
61. Madani A, Krause B, Greene ER, et al (2023) Large language models generate functional protein sequences across diverse families. *Nat Biotechnol* 1–8
62. Ji Y, Zhou Z, Liu H, Davuluri RV (2021) DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome. *Bioinformatics* 37:2112–2120
63. Mock F, Kretschmer F, Kriese A, Böcker S, Marz M (2022) Taxonomic classification of DNA sequences beyond sequence similarity using deep neural networks. *Proc Natl Acad Sci* 119:e2122636119



## Statutory Declaration

“I, Benjamin Pascal Voigt, by personally signing this document in lieu of an oath, hereby affirm that I prepared the submitted dissertation on the topic “*Streamlining medical data annotation in deep learning through self-supervised approaches*” (“Annotationsreduktion medizinischer Daten in Deep-Learning-Verfahren durch selbstüberwachte Lernansätze”), independently and without the support of third parties, and that I used no other sources and aids than those stated.

All parts which are based on the publications or presentations of other authors, either in letter or in spirit, are specified as such in accordance with the citing guidelines. The sections on methodology (in particular regarding practical work, laboratory regulations, statistical processing) and results (in particular regarding figures, charts and tables) are exclusively my responsibility.

Furthermore, I declare that I have correctly marked all of the data, the analyses, and the conclusions generated from data obtained in collaboration with other persons, and that I have correctly marked my own contribution and the contributions of other persons (cf. declaration of contribution). I have correctly marked all texts or parts of texts that were generated in collaboration with other persons.

My contributions to any publications to this dissertation correspond to those stated in the below joint declaration made together with the supervisor. All publications created within the scope of the dissertation comply with the guidelines of the ICMJE (International Committee of Medical Journal Editors; [www.icmje.org](http://www.icmje.org)) on authorship. In addition, I declare that I shall comply with the regulations of Charité – Universitätsmedizin Berlin on ensuring good scientific practice.

I declare that I have not yet submitted this dissertation in identical or similar form to another Faculty.

The significance of this statutory declaration and the consequences of a false statutory declaration under criminal law (Sections 156, 161 of the German Criminal Code) are known to me.”

---

Date

---

Signature

## Declaration of your own contribution to the publications

Benjamin Pascal Voigt contributed the following to the below listed publications:

**Publication 1: Voigt B**, Fischer O, Schilling B, Krumnow C, Herta C. Investigation of semi- and self-supervised learning methods in the histopathological domain. *Journal of Pathology Informatics*. 2023;100305.

Contribution Benjamin Voigt:

- Involved in conceptualization; Mainly responsible for study design; Literature review; Methodological domain adaptation of SimSiam and SimCLR
- Involved in implementation of SimSiam, SimCLR, PAWS; Processing of research data used; Conducting SimSiam experiments; Proportionate conduction of SimCLR experiments; Validation of the experimental results
- Analysis and synthesis of experimental results -- created analysis: Graphical abstract, Table 1, Figure 1, Table 2, Figure 2, Figure 3, Figure 4, Figure A.1, Table A.1, Figure B.1, Figure B.2, Figure B.3, Figure B.4, Figure B.5, Figure C.1, Figure C.2, Figure C.3, Figure C.4
- Publishing the implementation in an open source repository; Writing the publication draft; Incorporating the co-authors' comments; Finalizing the manuscript; Corresponding author and leading the publication process; Incorporating reviewers' feedback and communication with editors

**Publication 2: Voigt B\***, Fischer O\*, Krumnow C, Herta C, Dabrowski PW. NGS read classification using AI. *PLOS ONE*. 2021 Dec;16(12):1–13.

\* equal contribution

Contribution Benjamin Voigt:

- Involved in conceptualization; Mainly responsible for study design; Literature review Transformer
- Development and implementation of the classification pipeline; Model training P-Frame and P-Tax; Conduction of the classification experiments; Validation of the experiment results
- Involved in the evaluation design; Involved in analysis and synthesis of the experimental results - created analysis: Figure 1, Table 1, Figure 2, Figure 3, Figure 4, Figure S.1, Figure S.2, Table 2
- Involved in writing the publication draft (Abstract, Introduction, Methods, and Implementation [excluding Data generation], Results, Supporting Information); Reviewed co-authors' draft parts; Incorporated co-authors' comments; Finalized the manuscript; Published the implementation in an open source repository and the classification models on Zenodoo
- Based on reviewer feedback: conduct additional experiments and incorporate desired modifications

Contribution Oliver Fischer:

- Involved in conceptualization; Involved in study design; Literature review eXplainable AI
- Development of classification pipeline; Implementation and conduction eXplainable AI analysis of P<sub>frame</sub> and P<sub>tax</sub> models
- Main responsible for evaluation design; Involved in analysis and synthesis of experimental results

- Involved in writing publication draft (Methods and Implementation [excluding Data generation], Results); Review of co-authors' draft parts

---

Signature, date and stamp of first supervising university professor / lecturer

---

Signature of doctoral candidate (Benjamin Voigt)

---

Signature of co-author (Oliver Fischer)

## Printing copy(s) of the publication(s)

Printing copies of the publications are listed in the following order:


- **Voigt, B.\***, Fischer, O.\*, Krumnow, C., Herta, C., & Dabrowski, P. W. (2021). NGS read classification using AI. Plos one, 16(12), e0261548.
- **Voigt, B.**, Fischer, O., Schilling, B., Krumnow, C., & Herta, C. (2023). Investigation of semi-and self-supervised learning methods in the histopathological domain. Journal of Pathology Informatics, 100305.

## RESEARCH ARTICLE

## NGS read classification using AI

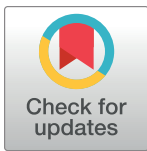
Benjamin Voigt , Oliver Fischer , Christian Krumnow, Christian Herta<sup>‡</sup>, Piotr Wojciech Dabrowski <sup>‡\*</sup>

Center for Bio-Medical image and Information processing (CBMI), HTW University of Applied Sciences, Berlin, Germany

 These authors contributed equally to this work.

<sup>‡</sup> CH and PWD also contributed equally to this work.

\* [piotr.dabrowski@htw-berlin.de](mailto:piotr.dabrowski@htw-berlin.de)

 OPEN ACCESS

**Citation:** Voigt B, Fischer O, Krumnow C, Herta C, Dabrowski PW (2021) NGS read classification using AI. PLoS ONE 16(12): e0261548. <https://doi.org/10.1371/journal.pone.0261548>

**Editor:** Yanbin Yin, University of Nebraska-Lincoln, UNITED STATES

**Received:** June 24, 2021

**Accepted:** December 3, 2021

**Published:** December 22, 2021

**Peer Review History:** PLOS recognizes the benefits of transparency in the peer review process; therefore, we enable the publication of all of the content of peer review and author responses alongside final, published articles. The editorial history of this article is available here: <https://doi.org/10.1371/journal.pone.0261548>

**Copyright:** © 2021 Voigt et al. This is an open access article distributed under the terms of the [Creative Commons Attribution License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

**Data Availability Statement:** All data files are available from zenodo (accessions 4306240, 4306248, 4307779, 4306420, 4306499). The source code is available from github: <https://github.com/CBMI-HTW/TaxonomicClassification-NGS-NN>.

## Abstract

Clinical metagenomics is a powerful diagnostic tool, as it offers an open view into all DNA in a patient's sample. This allows the detection of pathogens that would slip through the cracks of classical specific assays. However, due to this unspecific nature of metagenomic sequencing, a huge amount of unspecific data is generated during the sequencing itself and the diagnosis only takes place at the data analysis stage where relevant sequences are filtered out. Typically, this is done by comparison to reference databases. While this approach has been optimized over the past years and works well to detect pathogens that are represented in the used databases, a common challenge in analysing a metagenomic patient sample arises when no pathogen sequences are found: How to determine whether truly no evidence of a pathogen is present in the data or whether the pathogen's genome is simply absent from the database and the sequences in the dataset could thus not be classified? Here, we present a novel approach to this problem of detecting novel pathogens in metagenomic datasets by classifying the (segments of) proteins encoded by the sequences in the datasets. We train a neural network on the sequences of coding sequences, labeled by taxonomic domain, and use this neural network to predict the taxonomic classification of sequences that can not be classified by comparison to a reference database, thus facilitating the detection of potential novel pathogens.

## Introduction

Over the past one and a half decades, Next Generation Sequencing (NGS) has revolutionized genomics and adjacent fields of research. The ability to sequence massive amounts of DNA at ever-decreasing costs per base has led to an explosion of the genetic information available for researchers. For instance, since the introduction of the Roche 454, the first commercially successful NGS machine [1], in 2005, the number of bases in GenBank has grown from about  $10^{10}$  to almost  $10^{12}$ , at a staggering average rate of  $5 \times 10^{10}$  bases per month—the same number of bases every two months that it had previously taken 22 years to accumulate [2]. And this is just the analyzed tip of the iceberg: The Sequence Read Archive (SRA) currently holds over  $4 \times 10^{16}$  bases of raw NGS data [3].

**Funding:** BV and OF are funded by the Federal Ministry of Education and Research of Germany (BMBF, <https://www.bmbf.de/>) in the project deep. Health (project number 13FH770IX6). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

**Competing interests:** The authors have declared that no competing interests exist.

This massive amount of available data presents diverse challenges when it comes to data analysis. One common application of NGS is metagenomic sequencing, where all genetic material in a complex sample, such as a patient's body liquid (clinical metagenomics) or a piece of arctic ice (environmental metagenomics), is sequenced. While in targeted approaches, such as sequencing a cultured bacterium, the composition of the sample is known a priori and that knowledge can be used to inform the analysis, in metagenomic sequencing the primary data analysis task is determining that composition. The common approach to this challenge is using known reference sequences. Very broadly speaking, for each read from an NGS sample the similarity to all sequences within a reference database is determined and the read is classified as belonging to a taxon based on this comparison. While this approach allows highly successful detection of organisms with already sequenced relatives, as evidenced by the results of studies such as CAMI where all entries used variations on the above-mentioned approach [4], it does not allow the detection of entirely novel organisms—especially if those are only represented at low levels in the sample. Data from such organisms disappears within the thousands to millions of “unclassified” reads that remain as a byproduct of any metagenomic NGS dataset analysis—hidden among technical artifacts and reads from known organisms that could not be clearly assigned to a taxon.

Although such “unclassified” reads are usually discarded within the analysis workflow, the hints towards novel bacteria and viruses contained therein would be a valuable resource if they could be identified and isolated for further, more detailed analysis. This would facilitate the detection and characterization of the huge number of organisms that have not yet been sequenced—for instance, [5] estimate that there are around  $10^{14}$  bacterial taxa, of which only around  $10^6$  have been described, and [6] estimate that there are over  $3 \times 10^5$  undetected mammalian viruses.

It has previously been shown that machine learning can be a valuable tool in overcoming this challenge: [7] have successfully used Random Forests to predict the presence of sequences from human pathogenic organisms in metagenomic datasets. While several other approaches have used machine learning to optimize parameters of existing tools [8, 9], to our knowledge the work by [7] represents the only attempt to detect microbial sequences using machine learning without relying on reference sequences. Here, we aim to extend the understanding of such approaches' usefulness by applying transformer neural networks to the classification of NGS reads as mammalian, bacterial or viral in origin.

The application of neural network models is profoundly successful in the field of natural language processing [10, 11]. In particular, models based on the Transformer architecture [12] have led to significant breakthroughs in developing so-called language models that have shown state-of-the-art performance on a variety of natural language processing tasks [13–15]. One tremendous advantage is that such models can be trained using a self-supervised approach, i.e., there is no requirement for labeled data like in a supervised learning regime.

In recent work, language models from various transformer networks, primarily multi-layer bidirectional Transformer [14], have been trained on datasets containing a large number of unlabeled protein sequences, e.g., UniProt [16]. [17] developed the protein generation language model (ProtGen) that creates proteins that exhibit near-native structure energies. [18] investigated the learned representations of a protein language model. Their findings show that high-capacity networks reflect biological structure at multiple levels, including amino acids, proteins, and evolutionary homology. [19] compared the performance of the embeddings generated by several network architectures [14, 20–22] on multiple supervised learning tasks, e.g., classification into membrane and non-membrane proteins.

In this paper we build upon the aforementioned work on the application of transformer networks in protein classification to demonstrate their applicability to taxonomic classification

of NGS reads. Since the overarching goal is the detection of entirely novel organisms from metagenomic datasets, in this initial work we focus on a classification on the domain level, specifically into mammalian (i.e. host in the case of clinical metagenomic datasets), viral and bacterial reads. This will allow extraction and specific examination of reads representing hitherto undescribed viruses and bacteria from reads that remain in the “unclassified” bin after traditional metagenomic data analysis and taxonomic classification has been performed with tools such as Kraken [23], RIEMS [24], PathoScope [25], PAIPline [26] or MetaMeta [27]. Since we are building upon large existing models that have been trained on protein sequences, we limit this proof of concept to the classification of reads that lie within a coding sequence (CDS). While, in order to be able to correctly perform classification independent of a read’s offset within the CDS, we also automatically determine which of the six possible frame the read is in. This pre-requisite step in itself is a novel application of machine learning to ORF detection, as current tools either (i) rely purely on presence/absence of start/stop codons without further interpretation of the sequence (such as getorf [28] or OrfM [29]) or return all candidate sequences for each read without clearly resolving potentially contradictory hypotheses (such as FragGeneScan [30], CNN-MGP [31] or geneRFinder [32]). However, since this is a proof-of-concept work, we do not—in contrast to these existing tools—examine reads that are outside of CDSs in this paper.

## Methods and implementation

We developed a proof-of-concept for the classification of NGS-reads into the taxonomic domains viruses, bacteria, and mammals. The classification is done by concatenating multiple data processing and sub-classification steps.

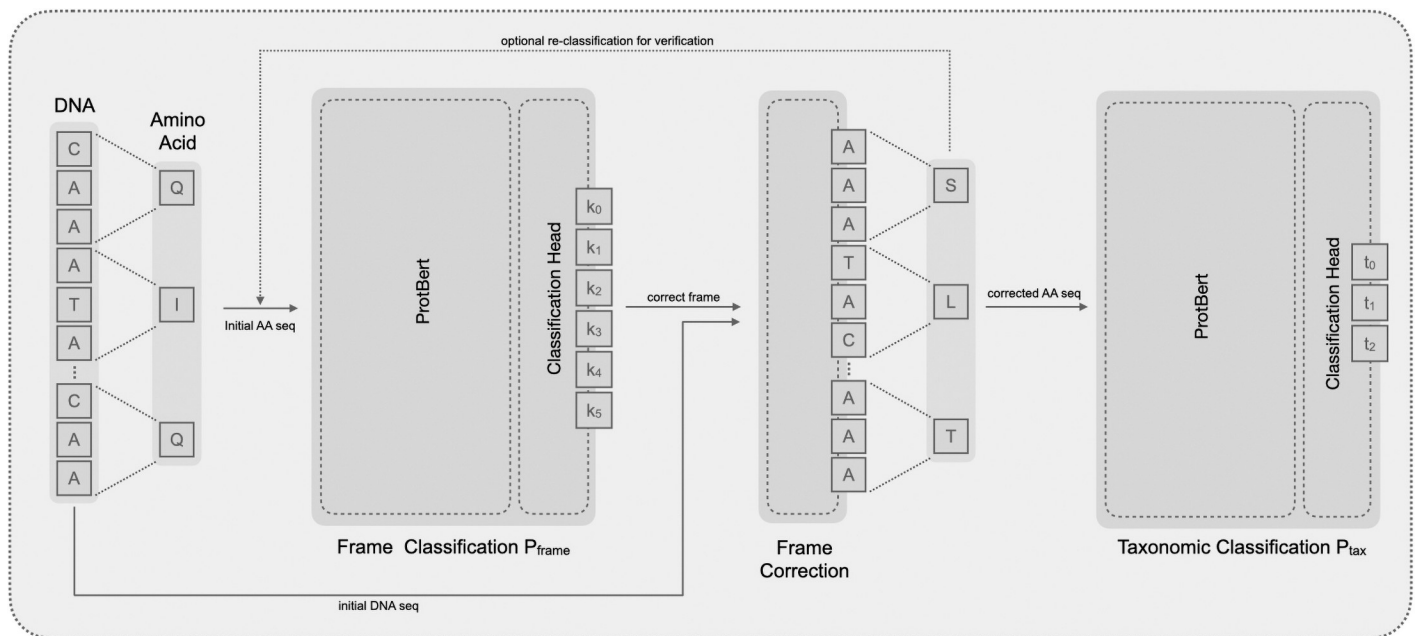
At first, the frame of a read within its CDS must be recognized to translate the DNA sequence fragments into amino acid sequences correctly. This is a non-trivial step because there are typically no start or stop codons in the fragments. We developed a classifier based on a language model to detect the correct frame of a read. Using this information, the read sequences can be translated into amino acid sequences. In a final step, the amino acid sequences are classified into taxonomic domains by another language model.

In this section, we describe this proof-of-concept pipeline in more detail. Then, we provide information about the design and training process of the individual classifiers which are used in different steps of the pipeline. Finally, we describe how training and test data sets were generated.

## Classification pipeline

The pipeline was implemented as a python script (see <https://github.com/CBMI-HTW/TaxonomicClassification-NGS-NN>). A general overview is shown in Fig 1.

As input the pipeline gets a file in the *FASTA* format with NGS reads. We expect the nucleotide sequences  $\mathbf{s}$  to be of 300 base pairs each, i.e.,  $\mathbf{s} = (s_1, \dots, s_{300})$ . This is not a strict requirement as shown by our experiments with different amino acid sequence length (S1 Appendix; S1 Fig), but since the classifiers were optimized on that length it will lead to the best results. While for the prototype, we assume that all reads lie within CDSs, we plan to add an automatic selection of such reads from raw NGS data in future work. Each nucleotide sequence is translated into a protein sequence  $\mathbf{x} = (x_1, \dots, x_{100})$  using biopython [33]. Often, this is not the correct translation because most reads are off-frame. Such wrong translations are detected in the next step and are then re-translated. However, if there are any stop codons in the initial translation, a full six-frame translation of the read is performed and, if a translation  $\hat{\mathbf{x}}$  without stop codons is found, this  $\hat{\mathbf{x}}$  is used instead.



**Fig 1. Overview of the complete neural network classification pipeline.** The pipeline consists of four major blocks: (1) preprocessing NGS reads, (2) frame classification of NGS reads, (3) frame correction and translation of NGS reads, (4) taxonomic classification of amino acid sequence. The dotted arrow line shows an optional loop of the frame classification used for checking the frame correction block, as shown in Fig 2.

<https://doi.org/10.1371/journal.pone.0261548.g001>

The resulting amino acid sequence  $\mathbf{x}$  from the initial translation is fed into the frame classification model that returns a probability distribution over the classes  $p_{frame}(k | \mathbf{x})$ . There are six possible classes  $k \in \{0, \dots, 5\}$  which are predicted by the model for each sequence: on-frame ( $k = 0$ ), offset by one base ( $k = 1$ ), offset by two bases ( $k = 2$ ), reverse-complementary ( $k = 3$ ), reverse-complementary and offset by one base ( $k = 4$ ) and reverse complementary and offset by two bases ( $k = 5$ ). Based on the classification result  $\hat{k}$ , with  $\hat{k} = \arg \max_k p_{frame}(k | \mathbf{x})$ , the transformations (shifting, reverse-complementing) necessary to make each amino acid sequence on-frame are performed on the original DNA sequences  $s$  before they are again translated into amino acid sequences  $\hat{\mathbf{x}}$  for further processing. Finally, each on-frame amino acid sequence  $\hat{\mathbf{x}}$  is classified into one of the taxonomic domains  $t$  used in this prototype: *virus*, *bacteria*, or *human/mammalian* using the second classification model

$$\hat{t} = \arg \max_t p_{tax}(t | \hat{\mathbf{x}}).$$

The output of the pipeline is the input *FASTA* file with a modified identifier line, i.e., information about the frame classification  $\hat{k}$  and species classification  $\hat{t}$  is appended.

### Model description, training, and fine-tuning

The classifiers are based on pre-trained (protein) language models. A language model is an assignment of a probability distribution to a sequence of tokens for a language. In our context, the tokens are amino acid symbols. Such language models are trained (self-supervised) on large corpora of sequences/sentences of a language, e.g. on protein data bases. In the training process the language model must continue a sequence (autoregressive) or should predict missing tokens which are masked in the input [14], i.e. it must assign high probabilities to the corresponding tokens of the training data. Therefore, no labeled data is needed.

A neural network language model can be used after training as a *feature extractor*, i.e., the token sequence is transformed by the model into a sequence of feature vectors. The sequence



of features can be used for other tasks, e.g., for the classification of a complete sequence. Here, the sequence of features is pooled into one feature vector with a fixed size. With additional labeled data a classification model can be trained on pairs of feature vectors and labels. This is an example of *transfer learning*: The knowledge learned by modeling the language is used for another task, e.g., a classification. The parameters of the feature extractor can also be modified for the specific task. Typically, a classification head replaces the last layer(s) of the pre-trained language model neural network such that the new neural network can be trained end-to-end on the classification data (sequence-label pairs). I.e., the parameters of the neural network are *fine-tuned* to improve the classification performance.

We used a pre-trained language model called ProtBert [19], a multi-layer bidirectional transformer neural network [14] with 30 layers and 420 million parameters. The language model was trained on the Uniref100 dataset [16]. As an important detail we note that the dataset contains protein sequences in their correct reading frame. Such sequences don't contain any stop codon which we take into account in our pipeline as discussed above in Section "Classification Pipeline" and in our data generation as described in Section "Data generation".

The language model maps an amino acid sequence  $\mathbf{x} = (x_1, \dots, x_n)$  into a sequence of feature vectors  $(\mathbf{h}'_1, \dots, \mathbf{h}'_n)$ . In case of the pre-trained language model ProtBert each  $\mathbf{h}'$  consists of 1024 elements. We reduce these sequence of feature vectors into a single feature vector  $\mathbf{h}$  by using different pooling strategies. We explored mean, max, and dot product self-attention functions for pooling. For classification, we fed the feature vector  $\mathbf{h}$  into a two-layer dense network (classification head), projecting this representation into unnormalized log probabilities  $\mathbf{z} = (z_1, \dots, z_c)$  with  $c$  being the number of classes of the task. The probabilities are computed from  $\mathbf{z}$  by a softmax operation. Note that the language models are trained on complete protein sequences. In contrast, the classification is done on (protein) fragments.

Both classification models  $P_{frame}$  and  $P_{tax}$  were trained with two different approaches: (pure) feature extraction and fine-tuning. In the first variant, the feature vectors generated by the transformer were fixed, i.e., the parameters that were trained by the language model objective are frozen. Only, the parameters of the classification head are adapted during training. This approach has the advantage of significantly reduced training time. Since only the small dense network parameters have to be updated, the batch size can be increased. However, updating all weights during the training process lets the pre-trained models adapt to the specific task. We explored this alternative using smaller batch sizes and different learning rates for the dense networks and the pre-trained language model. Regardless of the training type, we used the LAMB optimizer [34] to update the model parameters. We optimized model hyperparameters using the the ASHA algorithm [35]. We summarize the final hyperparameter settings of our experiments in Table 1.

**Table 1. Hyperparameter settings used for the training process of  $P_{frame}$  and  $P_{tax}$ .**

	$P_{frame}$	$P_{tax}$
Epochs	2	10
Batch size	128(8)	64(4)
LM Learning Rate	$1e^{-5}$	$25e^{-5}$
CN Learning Rate	0.0025	0.0005
CN Feature Number	512	256
CN Dropout Rate	0.0	0.25

For brevity in the table *LM* is short for language model and *CN* referring to the classification network.

<https://doi.org/10.1371/journal.pone.0261548.t001>

The model calculation was done using a small cluster consisting of 8 *Nvidia V100* GPUs. We realized distributed training through data parallelism, i.e., distribute the same model with different batches across the nodes. In [Table 1](#) we report the global batch size with the number of nodes in brackets if calculated distributed, e.g., 256(4), meaning a batch size of 64 per GPU.

## Data generation

To train the models as described above with the most reliable data available, we used amino acid sequences from Uniprot [16] for the taxonomic classification model and RefSeq [36] reference sequences for the frame classification model. We then tested the applicability of the trained models and the whole pipeline by classifying reads from two metagenomic sequencing projects available within the SRA [37], after selecting only the reads matching the criteria for this pipeline (i.e. those that lie within a viral, bacterial or mammalian CDS). The steps for generating the input sequences  $\mathbf{x}$  for the classifiers from the initial sequences  $\mathbf{s}$  of the three raw data sets are described in detail in the following sections.

**Training data for taxonomic classification.** For training of the taxonomic classification, we downloaded the 2020–04 release of the fully manually annotated and curated UniProtKB/Swiss-Prot database for bacteria, viruses and human as representative for mammalian sequences [16, 38]. For each amino acid sequence  $\mathbf{x} = (x_1, \dots, x_{N(\mathbf{s})})$  we created with a sliding window all possible patches  $(x_l, \dots, x_{l+99})$  for all  $l \leq N(\mathbf{s}) - 100$  of length 100.  $N(\mathbf{x})$  denotes the varying length of the initial sequence  $\mathbf{x}$  and sequences with  $N(\mathbf{x}) < 100$  were discarded.

The initial data is strongly unbalanced with respect to its biological origin. In order to split the data in training, validation and test sets we iteratively draw without replacement from all sequences  $\mathbf{x}$  and fill all patches generated from  $\mathbf{x}$  successively either in the test, validation or training. Further we balanced the data by considering all viral sequences and down sampling sequences with bacterial and human origin until all data sets contain the same number of patches for all three classes with an approximate ratio of (10% test, 10% validation, training 80%) of the total sizes.

The final data sets contain approximately  $1.8 \times 10^7$  patches and are deposited at zenodo [39].

**Training data for frame classification.** For training of the frame classification, randomly selected viral and bacterial genomes and the human (GRCh38.p13) reference genome were downloaded from GenBank [40]. From these genomes, all annotated CDS DNA sequences  $\mathbf{s}$  were extracted. Similar to the amino acid data, for each nucleic acid sequence  $\mathbf{s} = (s_1, \dots, s_{N(\mathbf{s})})$  using a sliding window all possible patches  $(s_l, \dots, s_{l+299})$  of length 300 as well as their reverse complemented versions  $(\overline{s_{l+299}}, \dots, \overline{s_l})$  for all  $l \leq N(\mathbf{s}) - 300$  were created and translated to amino acids using biopython [33]. By this, we create the on-frame sequence, as well as all possible off-frame configurations.

In order to avoid the model from relying on the presence of a stop codon for the classification of off-frame sequences, all sequences whose translation contained a stop codon were discarded. We split the data into three sets with approximate ratios of (10%, 10%, 80%) of the total sizes by placing all patches generated from one initial sequence  $\mathbf{s}$  into one of the three sets.

Due to the removal of sequences containing stop codons which are only present in off-frame sequences, on-frame sequences were heavily over-represented in these data sets. We balanced the data sets by discarding sequences in over-represented frames until all frames were present at the same ratio—and the final three data sets have the exact size ratios (10% test, 10% validation, 80% training). The resulting data sets contain a total of  $1.2 \times 10^7$  patches and are deposited at zenodo [41].

**Application data.** To test the applicability of the trained models to real data, we downloaded the raw NGS reads from two metagenomic SRA runs using a read length of 300: A human skin metagenomic study (SRR7188139) and a swine feces metagenome (ERR3013343). Since the proof-of-concept pipeline only classifies reads that lie within a CDS, eligible reads were extracted by mapping. To that end, all RefSeq viral and bacterial genomes, the human reference genome (GRCh38.p13) and the *Sus scrofa* reference genome (GCF\_000003025.6) were downloaded, annotated CDS sequences were extracted and raw reads were mapped to these using bowtie2 [42] with the `-end-to-end` parameter. Reads mapping to either only viral, bacterial or mammalian ORFs were selected for the application test.

## Benchmarking of frame classification

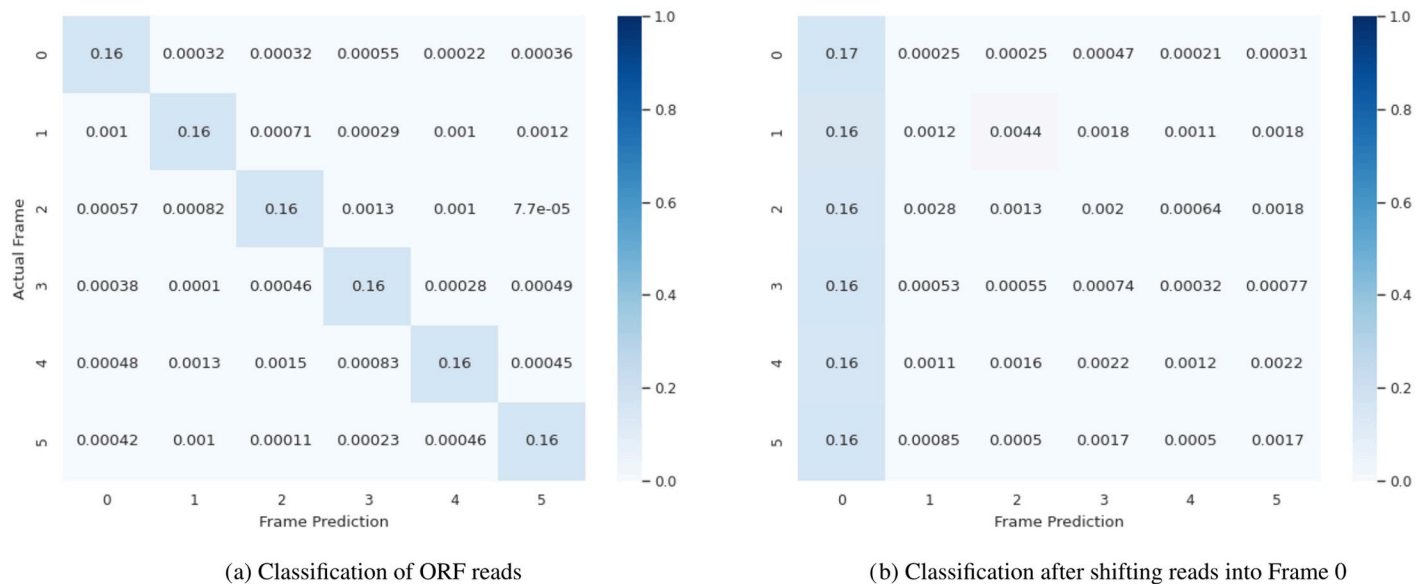
The first step of the pipeline—determining the frame for translating a read’s sequence—is a task that is also tackled by other existing tools. It is therefore not immediately obvious whether the best performance can be achieved by using frame classification using ProtBert, as shown in Fig 1, or by using one of these existing tools. In order to answer this question, we have compared the ability of our classifier to predict the correct frame of a read to that of other tools.

We have found six tools that tackle the task of determining the correct translation frame directly from short NGS reads: MGC [43], metaGun [44], metaGene [45], Orphelia [46], fragGeneScan [30] and CNN-MGP [31]. The last one of these, CNN-MGP, also uses a neural network to perform the classification. Unfortunately, out of these six tools, only two were suitable for our comparison. Orphelia requires a java binary that was built against gcc version 3.4, which has been superseded by version 4.0 in late 2006. Setting up a system with such old package versions was outside of the scope of this work. The websites referenced in the publications for metaGun and metaGene are offline, and MGC does not even mention any download website or include the binaries in the supplementary materials. We were not able to find any other resources such as mirrors or git repositories from which source code or binaries can be downloaded, making it impossible to run any of these tools. Accordingly, we only included fragGeneScan and CNN-MGP in our benchmarks.

In order to make the benchmark reproducible, we have implemented it as a nextflow [47] pipeline (see <https://gitlab.com/dabrowskiw/cdsfinderbenchmark>). For the evaluation, we have used the above-mentioned test dataset [41]. Since the documentation of CNN-MGP output is not entirely clear on how the reported reverse reading frames are encoded, we have manually tested all possible interpretations and chosen the one yielding the best results for CNN-MGP. We have also excluded reads for which CNN-MGP or fragGeneScan reported no reading frame from the calculation, since including these would have given our approach an unfair advantage—while we know that in this proof-of-concept work we have only included reads from within CDS and we thus predict a frame for each read, CNN-MGP and fragGeneScan can be applied to real data also including reads from non-coding regions and thus need to be able to predict that a read contains no valid frame.

## Results

In this section, we report the results of the trained model for two different settings. First, we test the taxonomic and frame classification models separately on the test data of the corresponding training setups. Then we use the full pipeline on real data from metagenomic sequencing studies.



**Fig 2. Prediction results of the frame classification model on the test dataset.** Predictions of the most probable class  $\hat{k}$  on the frame test data [41] are shown as an error matrix. The classes are as follows: on-frame (0), offset by one base (1), offset by two bases (2), reverse-complementary (3), reverse-complementary and offset by one base (4), reverse complementary and offset by two bases (5). (A) represents the initial classification results of the reads. (B) Re-evaluation of the reads after applying the frame correction to validate that the reads were correctly shifted to be on-frame, i.e.,  $k = 0$ .

<https://doi.org/10.1371/journal.pone.0261548.g002>

## Evaluation of both models

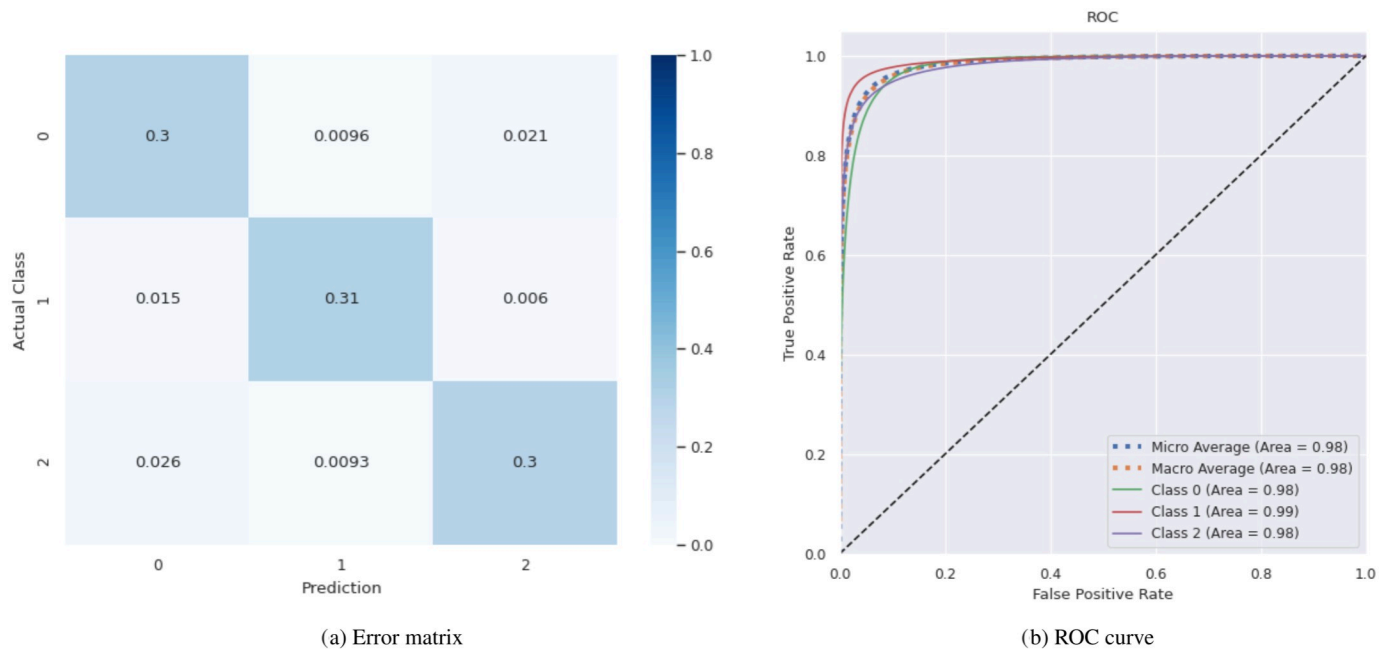
We evaluated the final classification models using a 10% split of generated data. As metrics, we report the ROC curve and error matrix as a heatmap for both tasks.

On the test dataset, the frame classification  $p_{frame}$  achieved an overall accuracy of 0.98 (S2 Fig in S1 Appendix). Since the datasets' classes are balanced, we expected a strong diagonal in the classification task's error matrix. This expectation has been confirmed; see Fig 2A. After applying the frame correction, we used the classification model to verify that the reads were correctly shifted into frame zero ( $k = 0$ ). We observed that almost all sequences had been moved accordingly (Fig 2B).

We measured an accuracy of 0.91 on the corresponding test data for the taxonomic classification model  $p_{tax}$  as shown in the error matrix (Fig 3A). We calculated the inner class accuracies to inspect that result in more detail. We observe that for reads predicted as bacterial, indeed 94% were correctly classified. In contrast, for sequences classified as viral, only 88% were actually of viral origin and 8% were mammalian. We observed a similar behavior in the reads classified as mammalian (92% mammalian and 6% were viral). This indicates that the classifier has the most problems in differentiating between these two classes. This observation is also reflected in Fig 3B, the classifier's ROC curve, where class 0 (viral) and 2 (human) are slightly worse compared to class 1 (bacterial). This is likely due to the presence of retroviral sequences in the human genome.

## Exemplary analyses

The exemplary analysis of data from real metagenomic sequencing studies presented a more challenging classification task, most likely due to more noisy data. Firstly, training and testing was performed using error-free sequences derived from curated references, while real NGS reads contain sequencing errors. Secondly, since filtering of reads belonging to a CDS was



**Fig 3. Prediction results of the taxonomic classification model on the test dataset.** Error matrix (A) and ROC curve (B) on the taxonomic test dataset [39] are shown. The classes are as follows: 0—viral, 1—bacterial, 2—mammalian.

<https://doi.org/10.1371/journal.pone.0261548.g003>

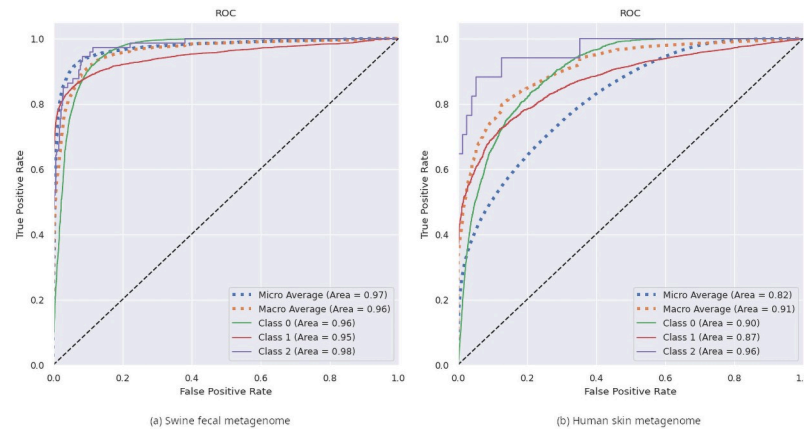
performed by mapping to all CDS sequences annotated in RefSeq, all automatic annotations that mistakenly classified a non-coding open reading frame as CDS cause reads that do not encode a real protein sequence to remain in the dataset. Still, a taxonomic domain classification precision of 0.87 could be achieved on the swine feces metagenomic dataset, or 0.90 if all reads that had a stop codon in every possible frame were discarded (since this should not happen in a correct read within a CDS and can thus be seen as indicative of an error). These results are visualized in Fig 4A.

Using SKESA [48] with default parameters on the reads classified as viral it was possible to de novo assemble five out of six CDSs of the porcine epidemic diarrhea virus present in the sample, each in a single config—the only CDS that could not be assembled was that encoding the envelope protein, since it is shorter than the read length of 300 bases used in the analysis and accordingly reads containing this CDS were discarded during dataset generation.

However, the classification of reads from the human skin metagenome only showed a precision of 0.62, due to a large number of bacterial reads being wrongly classified as viral. The resulting ROC curve is shown in Fig 4B. This disparate performance on different datasets warrants closer examination in the future. One possible explanation could be the presence of unannotated bacteriophage sequences on the bacterial reference genomes used for the mapping-based classification of the reads, which would lead to the observed discrepancy between the neural network's and the mapper's assessment of whether a sequence is bacterial or viral in origin.

### Benchmarking of frame classification

In the classification of the frame in a read, our approach using ProtBert (98.18% correctly classified frames) significantly outperformed both CNN-MGP (33.62% correctly classified frames) and fragGeneScan (58.65% correctly classified frames). However, it is important to note that



**Fig 4. Prediction results of the complete classification pipeline on data from real metagenomic sequencing studies.** ROC curves for taxonomic classification in swine feces metagenome (A) and human skin metagenome (B) datasets. The classes are as follows: 0—viral, 1—bacterial, 2—mammalian.

<https://doi.org/10.1371/journal.pone.0261548.g004>

due to the limitations described in the methods section, these results are only meaningful in the context of this specific proof of concept work. Especially given the current limitation to recognizing the frame of reads that wholly lie within CDS, this step of the pipeline does not represent a production-ready alternative method of determining the correct frame of NGS reads in general.

## Discussion

With this work, we have shown that a taxonomic classification on the domain level based on short sections of the amino acid sequence of an organism's proteins is possible using a transformer neural network without relying on a reference database. We have also demonstrated that it is possible to determine the frame of a short DNA sequence within an ORF using a transformer neural network without knowledge of the reference sequence or the usual comparison of a six-frame translation with a protein sequence database.

This novel application of transformer neural networks to sequence classification will support the reference-free detection of hitherto unknown bacterial and viral proteins—and, by proxy, unknown organisms—in the “unclassified” readset typically left over after metagenomic NGS data analysis. It could also support the analysis of metaproteomic experiments, allowing an initial high-level classification of peptides and thus aiding protein sequence assemblers such as SSAKE-based PASS [49] by presenting them with a reduced number of sequences with a lower likelihood of misleading overlaps.

Additionally, the ability to classify the frame of a short DNA sequence should be useful in diverse fields of study. For instance, it is very likely that this classification is possible due to an ability to recognize biologically functional amino acid sequences. In that case, the application of this classifier could allow the detection of recent frame-shift mutations in new genomes without requiring a high-quality reference sequence. It could also be integrated into gene detection algorithms to aid in the ranking of ORFs based on their likelihood of encoding a functional protein, complementing approaches such as that described by [50].

In order to allow simple integration into analysis workflows using raw NGS reads, in the future we will add an initial classification to determine which reads lie within an ORF and can thus be used for frame and taxonomic classification.



## Supporting information

**S1 Appendix. Classification with variable sequence length [51].**

(PDF)

**S2 Appendix. Frame classification with realistic NGS reads [41, 52, 53].**

(PDF)

## Acknowledgments

CK acknowledges the helpful support of Elisabeth Gasteiger concerning questions around the Swiss-Prot dataset.

We acknowledge Patrick Baumann for supporting the distributed training of the classification models.

Also available at zenodo are the trained models for frame classification [54] and taxonomic classification [55].

The code of the classification pipeline can be found in the github repository <https://github.com/CBMI-HTW/TaxonomicClassification-NGS-NN>.

The datasets analyzed in this study have been downloaded from the SRA and have been referenced by SRA accessions in the text.

## Author Contributions

**Conceptualization:** Benjamin Voigt, Oliver Fischer, Christian Krumnow, Christian Herta, Piotr Wojciech Dabrowski.

**Resources:** Christian Krumnow, Piotr Wojciech Dabrowski.

**Software:** Benjamin Voigt, Oliver Fischer, Christian Krumnow.

**Supervision:** Christian Herta, Piotr Wojciech Dabrowski.

**Writing – original draft:** Benjamin Voigt, Oliver Fischer, Christian Krumnow, Christian Herta, Piotr Wojciech Dabrowski.

**Writing – review & editing:** Benjamin Voigt, Oliver Fischer, Christian Krumnow, Christian Herta, Piotr Wojciech Dabrowski.

## References

1. Heather JM, Chain B. The sequence of sequencers: The history of sequencing DNA. *Genomics*. 2016; 107(1):1–8. <https://doi.org/10.1016/j.ygeno.2015.11.003> PMID: 26554401
2. NCBI. Genbank growth statistics; 2020. Available from: <https://www.ncbi.nlm.nih.gov/genbank/statistics/>.
3. NCBI. SRA growth statistics; 2020. Available from: <https://www.ncbi.nlm.nih.gov/sra/docs/sragrowth/>.
4. Sczyrba A, Hofmann P, Belmann P, Koslicki D, Janssen S, Droge J, et al. Critical Assessment of Metagenome Interpretation—a benchmark of metagenomics software. *Nat Methods*. 2017; 14(11):1063–1071. <https://doi.org/10.1038/nmeth.4458> PMID: 28967888
5. Lennon JT, Locey KJ. More support for Earth's massive microbiome. *Biol Direct*. 2020; 15(1):5. <https://doi.org/10.1186/s13062-020-00261-8> PMID: 32131875
6. Anthony SJ, Epstein JH, Murray KA, Navarrete-Macias I, Zambrana-Torrel CM, Solovyov A, et al. A strategy to estimate unknown viral diversity in mammals. *mBio*. 2013; 4(5):e00598–00513. <https://doi.org/10.1128/mBio.00598-13> PMID: 24003179
7. Deneke C, Rentzsch R, Renard BY. PaPrBaG: A machine learning approach for the detection of novel pathogens from NGS data. *Sci Rep*. 2017; 7:39194. <https://doi.org/10.1038/srep39194> PMID: 28051068

8. Vilne B, Meistere I, Grantina-Ievina L, Kibilds J. Machine Learning Approaches for Epidemiological Investigations of Food-Borne Disease Outbreaks. *Front Microbiol.* 2019; 10:1722. <https://doi.org/10.3389/fmicb.2019.01722> PMID: 31447800
9. Sedlar K, Kupkova K, Provaznik I. Bioinformatics strategies for taxonomy independent binning and visualization of sequences in shotgun metagenomics. *Comput Struct Biotechnol J.* 2017; 15:48–55. <https://doi.org/10.1016/j.csbj.2016.11.005> PMID: 27980708
10. Goldberg Y, Hirst G. *Neural Network Methods in Natural Language Processing.* Morgan & Claypool Publishers; 2017.
11. Deng L, Liu Y. *Deep Learning in Natural Language Processing.* 1st ed. Springer Publishing Company, Incorporated; 2018.
12. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. In: *Advances in neural information processing systems*; 2017. p. 5998–6008.
13. Radford A, Narasimhan K, Salimans T, Sutskever I. *Improving language understanding by generative pre-training*; 2018.
14. Devlin J, Chang MW, Lee K, Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805.* 2018.
15. Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, et al. Language Models are Few-Shot Learners. *CoRR.* 2020;abs/2005.14165.
16. TheUniProtConsortium. UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Research.* 2018; 47(D1):D506–D515.
17. Madani A, McCann B, Naik N, Keskar NS, Anand N, Eguchi RR, et al. ProGen: Language Modeling for Protein Generation. *arXiv preprint arXiv:2004.03497.* 2020.
18. Rives A, Goyal S, Meier J, Guo D, Ott M, Zitnick CL, et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *bioRxiv.* 2019; p. 622803.
19. Elnaggar A, Heinzinger M, Dallago C, Rehawi G, Wang Y, Jones L, et al. ProtTrans: Towards Cracking the Language of Life's Code Through Self-Supervised Deep Learning and High Performance Computing. *bioRxiv.* 2020.
20. Lan Z, Chen M, Goodman S, Gimpel K, Sharma P, Soricut R. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942.* 2019.
21. Dai Z, Yang Z, Yang Y, Carbonell J, Le QV, Salakhutdinov R. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860.* 2019.
22. Yang Z, Dai Z, Yang Y, Carbonell J, Salakhutdinov RR, Le QV. Xlnet: Generalized autoregressive pre-training for language understanding. In: *Advances in neural information processing systems*; 2019. p. 5753–5763.
23. Wood DE, Salzberg SL. Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome Biol.* 2014; 15(3):R46. <https://doi.org/10.1186/gb-2014-15-3-r46> PMID: 24580807
24. Scheuch M, Höper D, Beer M. RIEMS: a software pipeline for sensitive and comprehensive taxonomic classification of reads from metagenomics datasets. *BMC Bioinformatics.* 2015; 16:69. <https://doi.org/10.1186/s12859-015-0503-6> PMID: 25886935
25. Hong C, Manimaran S, Shen Y, Perez-Rogers JF, Byrd AL, Castro-Nallar E, et al. PathoScope 2.0: a complete computational framework for strain identification in environmental or clinical sequencing samples. *Microbiome.* 2014; 2:33. <https://doi.org/10.1186/2049-2618-2-33> PMID: 25225611
26. Andrusch A, Dabrowski PW, Klenner J, Tausch SH, Kohl C, Osman AA, et al. PAIPline: pathogen identification in metagenomic and clinical next generation sequencing samples. *Bioinformatics.* 2018; 34(17):i715–i721. <https://doi.org/10.1093/bioinformatics/bty595> PMID: 30423069
27. Piro VC, Matschkowski M, Renard BY. MetaMeta: integrating metagenome analysis tools to improve taxonomic profiling. *Microbiome.* 2017; 5(1):101. <https://doi.org/10.1186/s40168-017-0318-y> PMID: 28807044
28. Rice P, Longden I, Bleasby A. EMBOSS: the European Molecular Biology Open Software Suite. *Trends Genet.* 2000; 16(6):276–277. [https://doi.org/10.1016/S0168-9525\(00\)02024-2](https://doi.org/10.1016/S0168-9525(00)02024-2) PMID: 10827456
29. Woodcroft BJ, Boyd JA, Tyson GW. OrfM: a fast open reading frame predictor for metagenomic data. *Bioinformatics.* 2016; 32(17):2702–2703. <https://doi.org/10.1093/bioinformatics/btw241> PMID: 27153669
30. Rho M, Tang H, Ye Y. FragGeneScan: predicting genes in short and error-prone reads. *Nucleic Acids Res.* 2010; 38(20):e191. <https://doi.org/10.1093/nar/gkq747> PMID: 20805240
31. Al-Ajlan A, El Allali A. CNN-MGP: Convolutional Neural Networks for Metagenomics Gene Prediction. *Interdiscip Sci.* 2019; 11(4):628–635. <https://doi.org/10.1007/s12539-018-0313-4> PMID: 30588558



32. Silva R, Padovani K, Góes F, Alves R. geneRFinder: gene finding in distinct metagenomic data complexities. *BMC Bioinformatics*. 2021; 22(1):87. <https://doi.org/10.1186/s12859-021-03997-w> PMID: 33632132
33. Cock PJ, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, et al. Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*. 2009; 25(11):1422–1423. <https://doi.org/10.1093/bioinformatics/btp163> PMID: 19304878
34. You Y, Li J, Reddi S, Hseu J, Kumar S, Bhojanapalli S, et al. Large Batch Optimization for Deep Learning: Training BERT in 76 minutes. *International Conference on Learning Representations*. 2020.
35. Li L, Jamieson K, Rostamizadeh A, Gonina E, Ben-Tzur J, Hardt M, et al. A system for massively parallel hyperparameter tuning. *Proceedings of Machine Learning and Systems*. 2020; 2:230–246.
36. O'Leary NA, Wright MW, Brister JR, Ciufu S, Haddad D, McVeigh R, et al. Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation. *Nucleic Acids Res*. 2016; 44(D1):D733–745. <https://doi.org/10.1093/nar/gkv1189> PMID: 26553804
37. NCBI. Sequence Read Archive (SRA); 2009–2020. Available from: <https://www.ncbi.nlm.nih.gov/sra/>.
38. TheUniProtConsortium. UniProtKB/Swiss-Prot; 2020. Available from: [ftp://ftp.uniprot.org/pub/databases/uniprot/previous\\_major\\_releases/release-2020\\_03/knowledgebase/](ftp://ftp.uniprot.org/pub/databases/uniprot/previous_major_releases/release-2020_03/knowledgebase/).
39. Voigt B, Fischer O, Krumnow C, Herta C, Dabrowski PW. Uniprot datasets for training taxonomic classification; 2020. Available from: <https://zenodo.org/record/4306240>.
40. Clark K, Karsch-Mizrachi I, Lipman DJ, Ostell J, Sayers EW. GenBank. *Nucleic Acids Res*. 2016; 44(D1):67–72. <https://doi.org/10.1093/nar/gkv1276>
41. Voigt B, Fischer O, Krumnow C, Herta C, Dabrowski PW. Refseq datasets for training frame classification; 2020. Available from: <https://zenodo.org/record/4306248>.
42. Langmead B, Salzberg SL. Fast gapped-read alignment with Bowtie 2. *Nat Methods*. 2012; 9(4):357–359. <https://doi.org/10.1038/nmeth.1923> PMID: 22388286
43. El Allali A, Rose JR. MGC: a metagenomic gene caller. *BMC Bioinformatics*. 2013; 14 Suppl 9:S6. <https://doi.org/10.1186/1471-2105-14-S9-S6> PMID: 23901840
44. Liu Y, Guo J, Hu G, Zhu H. Gene prediction in metagenomic fragments based on the SVM algorithm. *BMC Bioinformatics*. 2013; 14 Suppl 5:S12. <https://doi.org/10.1186/1471-2105-14-S5-S12> PMID: 23735199
45. Noguchi H, Park J, Takagi T. MetaGene: prokaryotic gene finding from environmental genome shotgun sequences. *Nucleic Acids Res*. 2006; 34(19):5623–5630. <https://doi.org/10.1093/nar/gkl723> PMID: 17028096
46. Hoff KJ, Lingner T, Meinicke P, Tech M. Orphelia: predicting genes in metagenomic sequencing reads. *Nucleic Acids Res*. 2009; 37(Web Server issue):W101–105. <https://doi.org/10.1093/nar/gkp327> PMID: 19429689
47. Di Tommaso P, Chatzou M, Floden EW, Barja PP, Palumbo E, Notredame C. Nextflow enables reproducible computational workflows. *Nat Biotechnol*. 2017; 35(4):316–319. <https://doi.org/10.1038/nbt.3820> PMID: 28398311
48. Souvorov A, Agarwala R, Lipman DJ. SKESA: strategic k-mer extension for scrupulous assemblies. *Genome Biol*. 2018; 19(1):153. <https://doi.org/10.1186/s13059-018-1540-z> PMID: 30286803
49. Warren RL, Sutton GG, Jones SJ, Holt RA. Assembling millions of short DNA sequences using SSAKE. *Bioinformatics*. 2007; 23(4):500–501. <https://doi.org/10.1093/bioinformatics/btl629> PMID: 17158514
50. Guo FB, Xiong L, Teng JL, Yuen KY, Lau SK, Woo PC. Re-annotation of protein-coding genes in 10 complete genomes of Neisseriaceae family by combining similarity-based and composition-based methods. *DNA Res*. 2013; 20(3):273–286. <https://doi.org/10.1093/dnares/dst009> PMID: 23571676
51. Voigt B, Fischer O, Krumnow C, Herta C, Dabrowski PW. Uniprot datasets for testing taxonomic classification for varying sequence length; 2020. Available from: <https://zenodo.org/record/4307779>.
52. Huang W, Li L, Myers JR, Marth GT. ART: a next-generation sequencing read simulator. *Bioinformatics*. 2011; 28(4):593–594.
53. Voigt B, Fischer O, Krumnow C, Herta C, Dabrowski PW. Refseq Test Subset for Frame Classification with and without Errors; 2021. Available from: <https://zenodo.org/record/5549620>.
54. Voigt B, Fischer O, Krumnow C, Herta C, Dabrowski PW. PyTorch model for frame classification; 2020. Available from: <https://zenodo.org/record/4306420>.
55. Voigt B, Fischer O, Krumnow C, Herta C, Dabrowski PW. PyTorch model for taxonomic classification; 2020. Available from: <https://zenodo.org/record/4306499>.



## Investigation of semi- and self-supervised learning methods in the histopathological domain

Benjamin Voigt<sup>a,b,\*</sup>, Oliver Fischer<sup>a,b</sup>, Bruno Schilling<sup>b</sup>, Christian Krumnow<sup>b</sup>, Christian Herta<sup>b</sup>

<sup>a</sup> Charité - Universitätsmedizin Berlin, Corporate Member of Freie Universität Berlin and Humboldt-Universität zu Berlin, Charitéplatz 1, 10117 Berlin, Germany

<sup>b</sup> University of Applied Sciences (HTW) Berlin, Center for Biomedical Image and Information Processing, Ostendstraße 25, 12459 Berlin, Germany



### ARTICLE INFO

#### Keywords:

Neural networks  
Deep learning  
Self-supervised learning  
Semi-supervised learning  
Computational pathology  
Tissue analysis

### ABSTRACT

Training models with semi- or self-supervised learning methods is one way to reduce annotation effort since they rely on unlabeled or sparsely labeled datasets. Such approaches are particularly promising for domains with a time-consuming annotation process requiring specialized expertise and where high-quality labeled machine learning datasets are scarce, like in computational pathology. Even though some of these methods have been used in the histopathological domain, there is, so far, no comprehensive study comparing different approaches. Therefore, this work compares feature extractors models trained with state-of-the-art semi- or self-supervised learning methods PAWS, SimCLR, and SimSiam within a unified framework. We show that such models, across different architectures and network configurations, have a positive performance impact on histopathological classification tasks, even in low data regimes. Moreover, our observations suggest that features learned from a particular dataset, i.e., tissue type, are only in-domain transferable to a certain extent. Finally, we share our experience using each method in computational pathology and provide recommendations for its use.

### Introduction

Supervised learning is still the dominant paradigm for building machine learning applications. A large, diverse, and qualitatively annotated dataset is needed for a robust and adequate prediction performance of classifiers trained with such a paradigm. For natural images, such datasets are publicly available<sup>1</sup> and have significantly contributed to the success of deep learning.<sup>2</sup> Models trained on such datasets are publicly available for standard deep learning frameworks as part of their corresponding model zoo. Those models typically consist of a feature extractor and an additional classification head, where the latter is usually highly trained on the specific task, and the former can be transferred to and reused for other applications in different data domains.

A problem with such a transfer approach arises when a certain proportion of the learned features are not helpful for the classification task, and on the other hand, necessary nuances in the data are not faithfully differentiated. Usually, this is the case in the histopathological domain, where the images differ significantly, e.g., from natural images.<sup>3</sup> Therefore, the need for appropriate datasets in the domain to apply the supervised learning approaches is significant. Unfortunately, publicly available labeled histopathological datasets suitable for machine learning barely exist due to the required expertise for annotations and legal regulations that restrict the

use of such data. Furthermore, the few existing datasets are either small or specialized for one particular task. Consequently, one of these datasets cannot represent the tremendous variety of tissue types, tissue characteristics, and changes due to different diseases.

In general, a remedy to overcome the need for a diverse large-scale labeled dataset is using unlabeled data to improve the prediction (classification or regression) performance. Thus, different approaches utilizing semi- or self-supervised learning paradigms have been developed to improve the model performance on various downstream tasks. Some of these approaches have recently been transferred to or optimized for the field of histopathology (Section Related Work).

This work explores the effectiveness of such learning paradigms aside from supervised learning in computational pathology. For this, we compare the performance of feature extractors resulting from 3 different state-of-the-art pretraining methods on different histopathological classification tasks. For a more comprehensive study, we chose methods with opposing learning paradigms, i.e., PAWS<sup>4</sup> as a semi-supervised learning method, SimCLR<sup>5</sup> as a contrastive self-supervised learning method (SSL), and SimSiam<sup>6</sup> as a non-contrastive self-supervised learning method (NCSSL). In Section Models and pretraining protocol, we provide a unified review of the details of each method from which their similarities and differences become evident.

\* Corresponding author.

E-mail address: [benjamin.voigt@htw-berlin.de](mailto:benjamin.voigt@htw-berlin.de) (B. Voigt).

<sup>1</sup> URL: <https://github.com/bensnajdar/histopathology-ssl>.

We trained feature extractors using publicly available histopathological datasets with each method and evaluated them on various in-domain classification tasks.<sup>7-9</sup> To allow for comparability of the different methods, we hereby aligned the corresponding training protocols as much as possible. We especially use a unified finetuning protocol for all 3 methods that allows comparing the impact of the different pretraining methods directly. Furthermore, we explored how applicable the methods were to a new data domain and investigated them with different data augmentation strategies suited for histopathological data. We examined the performance difference between the feature extractor, finetuned models, and models trained from scratch in a standard supervised fashion on different amounts of labeled data to study the benefit of pretraining in such settings. Furthermore, we analyzed how the representations of the feature extractors transfer to additional in-domain data, i.e., other downstream tasks.

Our results show that differences in the performance of the different learning paradigms under the same conditions are observable.<sup>2</sup> The results indicate that pretraining with any method on in-domain data seems beneficial. However, we found that SimCLR yields the most stable performance, while SimSiam has the best overall results. Surprisingly, despite explicitly using label information, PAWS showed the weakest performance in our experiments.

## Related work

Due to the challenges associated with creating a usable labeled dataset for machine learning, early efforts were made to use fewer or no labels, i.e., semi-supervised learning or unsupervised learning. In particular, established unsupervised learning methods, like autoencoder, random forest, clustering, transfer learning, or newer ones, e.g., generative adversarial networks, have been studied in the histopathological domain for some time. McAlpine et al<sup>10</sup> reviews some of these methods and their application. However, in this study, we investigate a learning paradigm that recently came into focus: self-supervised learning.<sup>11</sup>

**Self-supervised learning** is well suited for computer vision tasks with large neural networks. It divides into 2 classes contrastive self-supervised learning method (SSL) and non-contrastive self-supervised learning method (NCSSL).

SSL generates representations (views) of samples by using data augmentation techniques. Representations of the same samples denote positive pairs and representations of different inputs as negative pairs. In this context, these representations are typically mapped via a CNN, also called a feature extractor, into an embedding space. A contrastive loss function minimizes the distance between embeddings of positive representations while the distance to embeddings of negative representations is maximized. Widespread implementations of this concept are CPC,<sup>12</sup> SimCLR,<sup>5</sup> and MoCo.<sup>13</sup> These approaches are already present in different studies on histopathological tasks and indicate a benefit if a feature extractor is pretrained within the data domain. Saillard et al<sup>14</sup> trained feature extractors using MoCo on a public cancer dataset<sup>15</sup> to show that these consistently outperform their counterparts pretrained using ImageNet. Baykaner et al<sup>16</sup> presented their DIME pipeline to evaluate pretrained CNNs using SimCLR as self-supervised learning methods on an extensive, diverse whole slide images (WSI) dataset created from The Cancer Genome Atlas (TCGA). They investigated the feature embedding space of these models in a detailed graphical fashion using UMAP visualizations, McInnes and Healy.<sup>17</sup> Lu et al<sup>18</sup> combined CPC with multiple instance learning to classify breast cancer on the publicly available BACH dataset.<sup>19</sup> Additional papers, which differ mainly in using different datasets for pretraining or other downstream tasks also show the benefit.<sup>20-22</sup>

NCSSL methods utilize pretraining without the use of negative examples. Possible implementations usually fulfill domain-agnostic or domain-specific subtasks to train a feature extractor. An agnostic task can be any transformation that does not require expert or domain knowledge.

For example, an image could be tiled and shuffled. The task would be to arrange the tiles in the correct order to create the original or the identification of different rotations of a sample. A histopathological-specific task could be to order different magnification levels of a WSI patch. Such self-supervised strategies are used as pretraining by Koohbanani et al,<sup>23</sup> and Srinidhi et al.<sup>24</sup> Another class of NCSSL methods attempts to learn embedding space mappings, such that different augmented views of an image are mapped closely together. BYOL<sup>25</sup> uses 2 distinct networks to create such representations, while SimSiam utilizes Siamese Networks.<sup>26</sup> These methods have also been used for pathological tasks.<sup>27-29</sup> Furthermore, SimTriplet is another implementation designed explicitly for histopathological data. This approach emerged as a further development of SimSiam using the multi-view nature of histopathological WSIs. In addition to 2 augmented views from 1 patch, SimTriplet crops a second patch within the spatial neighborhood of the first one from the WSI, which is included in the similarity measure. Since tissue is assumed to be locally similar, SimTriplet adds additional positive pair information, resulting in better embedding space clustering.

So far, only a few works have analyzed self-supervised learning methods for histopathological downstream tasks in greater detail, e.g. Ciga et al.<sup>30</sup> They pretrained multiple CNN architectures on 57 fully unlabeled histopathology datasets drawn from TCGA using SimCLR and tested the performance on different downstream classification and regression tasks. In addition, they investigated the transferability of feature extractors between different tissue types, analyzed the impact of augmentation stacks and patch resolution on pretraining, and the impact of dataset size on performance in the downstream task. We serve as a complementary study to their work and expand this by evaluating additional methods.

**Semi-supervised learning** approaches allow the usage of unlabeled data combined with a small amount of labeled data. This technique is particularly suitable for the medical field, where insufficient resources with the necessary expertise and a lack of time lead to a bottleneck in annotating data. Methods, which implement the semi-supervised learning paradigm, are numerous. We limit ourselves here to give some examples of the application of this paradigm in the context of artificial neural networks and histopathological data.

Wang et al<sup>31</sup> created a semi-supervised learning pipeline for nuclei detection. The approach reconstructs images from detection maps created from unlabeled data. The spatial consistency between the original image and the reconstruction is used as a regularizing effect in the actual training of the detection network. In addition, several works exist utilizing unlabeled and labeled data to train generative adversarial networks (GANs) to solve the stain normalization problem, i.e., a color shift between different digitalized tissues caused by the staining and scanning process.<sup>32-34</sup> Finally, an interesting approach was recently published by Liu et al.<sup>35</sup> They used a small amount of labeled training data to stabilize the pseudo-label prediction of an unsupervised trained feature extractor. Instead of using a few data points with classical direct labels, it is also possible to use so-called weak-labels corresponding to assigning a high-level label for multiple unlabeled data (e.g., images) belonging to one category; see, for instance Sikaroudi et al.<sup>36</sup>

Besides these application examples, we utilized a recent development method named PAWS,<sup>4</sup> which shows exceptional classification performance on ImageNet with a drastically reduced amount of labeled data. However, to the authors' knowledge, PAWS has been no application in the medical domain, especially in the analysis of histopathological images.

## Models, material, and methods

This section presents the datasets and the 2 protocols used for training models in the pretraining and finetuning. Both protocols combined form our full experimental pipeline.

In this regard, the pretraining protocol specifies our environment to train models with self- and semi-supervised methods. Such models consist of an encoder that maps (encodes) the input into a high-dimensional space and a method-specific head. The encoder part is further trained

<sup>2</sup> Experiment protocols and implementation of the study can be found in the GitHub repository <https://github.com/bensnajdar/histopathology-ssl>.

using supervised learning on a so-called downstream task, i.e., a specific histopathological problem. The finetune protocol unifies the settings for such downstream training over all experiments and enables us to evaluate the performance of an encoder as a feature extractor. It also ensures the comparability of the learned encoders despite different self- and semi-supervised methods used in the pretraining.

In addition, the self- and semi-supervised methods and their necessary adaptations, which we applied, are briefly reviewed.

### Datasets

All data used are publicly available. We selected 2 datasets with different tissues, classes, and histopathological tasks for pretraining: Kather and PCam. The Lizard dataset was used exclusively to explore and evaluate the resulting encoders. In the following, we provide a brief introduction to all datasets. Additionally, Table 1 shows the training dataset and split sizes used in this work.

**PCam:** The PatchCamelyon is extracted from histopathological slides of lymph node sections provided by the Camelyon16 Challenge.<sup>37</sup> The original hematoxylin and eosin (H&E) stained tissue is digitized using a 40x objective resulting in a pixel resolution of 0.243 microns. This process was performed in 2 different laboratories using different scanners, resulting in color differences in the images, i.e., a potential distributional shift in the data. PCams patch-based dataset is sampled from the WSIs, keeping the initial split into train and test sets. Validation images are drawn from 20% of the training set. It contains 262.144 examples for the training set and 32.768 for each validation and test set. Images are 96x96 pixels and increased to 10x magnification by undersampling. Therefore, the resolution of the images changes to about 0.972 microns/pixel. Patches containing mostly background are filtered out in the sampling process. The resulting publicly available dataset is a balanced binary classification dataset with the positive class indicating the presence of metastatic tissue in the center of an image.

**NCT-CRC-HE-100K:** After its creator, this dataset is often referred to as **Kather**. The training set consists of 100.000 non-overlapping images of  $224 \times 224$  pixels scanned at 0.5  $\mu\text{m}/\text{pixel}$  spatial resolution from H&E stained histological images of human colorectal cancer and normal tissue. It is divided into 9, approximately balanced, tissue classes: Adipose (ADI), background (BACK), debris (DEB), lymphocytes (LYM), mucus (MUC), smooth muscle (MUS), normal colon mucosa (NORM), cancer-associated stroma (STR), and colorectal adenocarcinoma epithelium (TUM). In addition, the dataset is available in an authentic and stain-normalized variant. We used only the latter. One-tenth of the training images were split-off for validation purposes. As a test set, the co-published CRC-HE-7K dataset was utilized as recommended by Kather et al.<sup>8</sup> The 7180 images are also from patients with colorectal adenocarcinoma but do not overlap with the training data. Their resolution is the same at about 0.5 microns/pixel.

**Lizard:** The Lizard dataset is a large-scale histopathological dataset, for instance, segmentation by Graham et al.<sup>9</sup> Images are  $224 \times 224$  pixels with 20x magnification. Originally, the dataset is designed for instance

**Table 1**

Total and average annotated samples per class for each split size used in the downstream tasks, given in %. Data points were uniformly sampled for each split.

Data	Split in %	Samples	Avg. samples per class
Kather	100	90 000	10 000
	8	7200	800
	0.8	720	80
	0.2	180	20
PCam	100	262 144	131 072
	8	20 971	10 485
	0.8	2097	1048
	0.2	524	262
Lizard	100	297 245	74 311
	8	23 780	5945
	0.8	2378	594
	0.2	594	149

segmentation. Based on the publicly available dataset, we created an object classification task by cropping the images such that the label-defining object is positioned in the image center. In addition, we applied zero padding to increase the patch size, if necessary, back to the initial resolution.<sup>3</sup> The unfiltered dataset includes 6 classes: neutrophil, epithelial, lymphocyte, plasma, eosinophil, and connective tissue. In preprocessing, we removed the neutrophil (4116 samples) and eosinophil (2979 samples) classes to balance the dataset since these 2 classes were vastly underrepresented. Our train, validation, and test sets are created from distinct images and contain 29 7245, 64 901, and 62 672 samples, respectively. The resolution of the patches is about 1.167 microns/pixel.

### Models and pretraining protocol

We adapted each of the self- and semi-supervised methods to get the best performance possible in the new data domain. However, we implemented an open protocol for the model training to increase interpretability and comparability in the evaluation process between these methods.

The network architectures of each learning method can be split into an encoder  $f$ , also denoted as feature extractor, followed by a projection head  $h$ , depending on the method (Fig. 1). The ResNet family<sup>38</sup> is used as the base architecture for  $f$ . The networks' multi-layer perceptrons (MLPs) were adjusted to match the structure of the proposed method. We trained the encoder with 3 different architectures to investigate how model complexity affects the quality of the feature extractor later in the experiments. The ResNet50 is studied as a standard, as by most other works. Since the histopathological datasets are often small, ResNet18 is also used since it has fewer parameters and is built on a simpler structure, not using the bottleneck approach like more complex ResNet networks. In addition, we trained models with a Wide\_ResNet28w2, which has slightly more parameters than the ResNet18 and operates with more channels in the convolution blocks than a default architecture. Therefore, in terms of complexity, this results in the following ascending order: ResNet18, Wide\_ResNet28w2, and ResNet50. The projection heads  $h$  used are small MLPs. Each method uses an individual structure for  $h$ , and we left these MLPs as proposed in the original publications.

We performed a broad hyperparameter search for each architecture-method-data triplet to select optimizer parameters, augmentation stack, and method-specific parameters to adapt the methods to the domain. The latter are fixed after generally well-working values were found. Depending on the method, the searches are performed using grid search and more advanced techniques like Adaptive-Asha.<sup>39</sup> We selected the final training settings (Appendix A.2), based on the feature space clustering estimated by UMAP or t-SNE projections<sup>17,40</sup> and model performance measured by method-specific proxy metrics described in the corresponding method sections. In all final training settings, an SGD optimizer, either with or without a LARC/LAMB optimizer, is used, and training is performed using specific random seeds used consistently for all methods, which especially ensures that all methods use the same initialization of the encoders, and thus reduces corresponding variations effect between the methods.

All models were trained with input dimensions of  $96 \times 96$ , corresponding to the smallest image patch size of the utilized datasets. Resulting in a change of the microns/pixel values for datasets based on an image size of 224 by a factor of 7/3. Besides image scaling, several other data augmentation techniques were applied to the images.

We investigated different augmentation stacks for each method in the pretraining process. Starting from the originally published stack, we designed modified variations of these stacks by adjusting parameters and adding transformations, which proved beneficial for the data domain in the past.<sup>41</sup> Appendix A.1 describes each stack in more detail.

In essence, all methods use augmentation techniques to create different views of the same input and then optimize the encoder parameters such

<sup>3</sup> A script to reproduce the lizard classification dataset is also included in the GitHub repository ([https://github.com/bensajdar/histopathology-ssl/blob/main/utis/create\\_lizard\\_dataset.py](https://github.com/bensajdar/histopathology-ssl/blob/main/utis/create_lizard_dataset.py)).



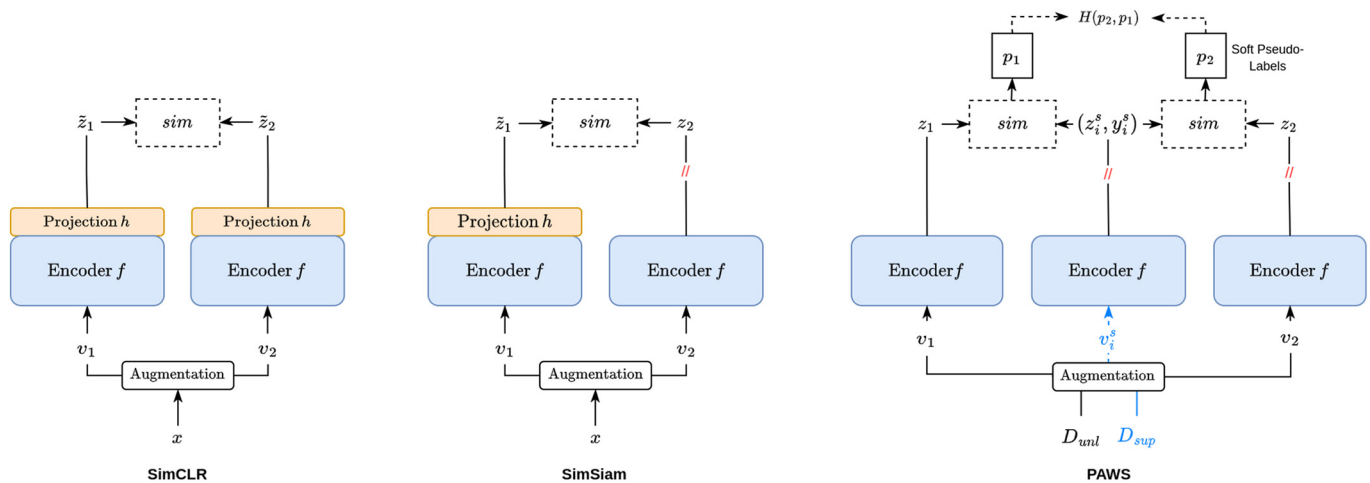


Fig. 1. Semi- and self-supervised methods used for the model training. Each method forwards different views of an image (or images) through a network structure based on a backbone CNN network  $f$ , i.e., the encoder, and an MLP projection head  $h$ . The resulting representations are compared by a similarity function  $sim$  and assessed by an individual loss function; see the related method section for details. Crossed-out paths do not contribute to the parameter update.

that those views are mapped in a meaningful way into an embedding space by either comparing them to labeled data (PAWS), maximizing their distance in embedding space from additional negative examples (SimCLR) or clustering embedding of similar inputs (SimSiam, SimCLR).

#### PAWS

PAWS is a semi-supervised learning method meaning that it utilizes partially labeled data. The training data  $D$  consists of a labeled support set  $D_{sup}$  and a much larger set of unlabeled data, denoted by  $D_{unl}$ . The basic idea of PAWS can be described in 3 major steps; see Fig. 1. First, 2 views are generated for an image drawn from  $D_{unl}$  via an augmentation stack, including random operations, and mapped through the encoder network  $f$  into embedding vectors. Additionally, a balanced batch of annotated support images is sampled, augmented by the same stack, and likewise encoded into embedding vectors. Finally, (soft) pseudo-labels are derived for each view by comparing the views' embeddings with the embeddings of the support set samples using a similarity function  $sim$ . As an optimization task, the difference between these pseudo-labels is minimized so that the encoder network becomes stable against different augmentations and learns important features by exploiting the annotated support set.

On a formal level, for an input  $x \in D_{unl}$ , the pair of views  $(v_1, v_2)$  and the corresponding embeddings  $(z_1, z_2)$  are generated.<sup>4</sup> Similarly for a support set batch  $(x_1^{(s)}, y_1^{(s)}), \dots, (x_{n_s}^{(s)}, y_{n_s}^{(s)}) \in D_{sup}$ , the embeddings  $z_1^{(s)}, \dots, z_{n_s}^{(s)}$  are generated where  $y_i^{(s)} \in \mathbb{R}^K$  is the one-hot encoded label of  $x_i^{(s)}$  with  $K$  being the number of classes. The pseudo-labels  $(p_1, p_2)$  for  $(v_1, v_2)$  are yield by

$$p_i = \frac{\sum_{j=1}^{n_s} sim(z_i, z_j^{(s)})}{\sum_{k=1}^{n_s} sim(z_i, z_k^{(s)})} y_j^{(s)} \quad (1)$$

with  $i = 1, 2$ . We used the exponential temperature scaled cosine similarity as  $sim(z_i, z_j) = \exp(z_i^T z_j / \|z_i\| \|z_j\| \tau)$  with  $\tau$  denoting the temperature parameter, which is 0.1 in all experiments. Since by construction  $p_1, p_2 \in [0, 1]^K$  and the sum of their elements is normalized to 1, they are interpreted as a class probability distribution, providing (soft) pseudo-labels, for the views  $(v_1, v_2)$ . At last,  $p_1, p_2$  are compared by cross-entropy  $H$ , which constitutes the cost function of PAWS. In practice, the 2 training batches are independently sampled from  $D_{unl}$  and  $D_{sup}$ . To optimize the

function of the supporting batch, it should be ensured that each class is equally represented in the support set.

As suggested in Assran et al.,<sup>4</sup> we modified the CNN encoder  $f$  by adding 3 linear layers, while the first 2 layers included batch normalization and ReLU as an activation function. The input and hidden dimensions of the MLP were chosen according to the CNN architecture (128 for the Wide\_ResNet28w2, 2048 for ResNet50, and 512 for ResNet18), and we fixed the output dimension to 128.

We employed PAWS without methodological adaptations on the histopathological data. PAWS models were trained using a LAMB optimizer for about 10 epochs, concerning the size of  $D_{unl}$ . Since the approach outputs soft pseudo-labels, we could compute classification metrics for a validation set to monitor the learning process as for standard supervised learning approaches. We trained multiple models for each architecture–method–data pair exploring distinct sizes of the support set  $D_{sup}$ . Given a training set  $D$  of size  $N$ , we consider support sets of size  $0.08N$ ,  $0.008N$ , and  $0.002N$ . The smallest support set size resulted in about 20 labeled images per class for Kather, which is extremely low considering modern deep learning approaches. Table 1 provides a detailed overview of data split sizes.

#### SimCLR

SimCLR is a self-supervised learning method based on the contrastive approach. The fundamental training objective is to maximize the distance of encodings resulting from fundamentally unequal inputs. This process is controlled by minimizing the embedding similarity of views generated from the same input, while simultaneously maximizing the distance to all other views in a training batch. Again, views are generated by an augmentation stack containing random operations.

On a formal level from the unlabeled dataset  $D$ , a batch  $x_1, \dots, x_n \in D$  is drawn and for each sample  $x_i$ , 2 views  $(v_{2i-1}, v_{2i})$  (forming a positive pair) are generated, resulting in  $2n$  views  $v_1, \dots, v_{2n}$ . The other  $2n - 2$  views act as negative (contrastive) examples for each positive pair of views. We created the views with a reduced augmentation stack as the original SimCLR paper proposed using a strong color distortion and random crop only (Appendix A.1). Each view  $v_i$  is processed through an encoder  $f$  and an additional projection head  $h$  to create a corresponding encoding  $\tilde{z}_i$ , i.e.,  $h(f(v_i)) = \tilde{z}_i$ .<sup>5</sup>

<sup>4</sup> The notation was adapted for consistency. In the original formulation the views were denoted as  $v$  and  $v_+$  with the corresponding embeddings  $z$ ,  $z_+$  and pseudo-labels  $p$  and  $p_+$ .

<sup>5</sup> We refer to the output of the full network  $\tilde{z}_i$  as encoding here as we want to distinguish it from the outputs of encoders, which we usually refer to as embedding.

The loss function is built around encoding  $\tilde{z}_i$ . Therefore, the loss calculates the similarity of encoding  $\tilde{z}_i$  and all other encodings  $\tilde{z}_j$ , which result from views of different inputs, by a function *sim* and is given by

$$L = \frac{1}{2n} \sum_{k=1}^n [l_{2k-1,2k} + l_{2k,2k-1}]$$

$$\text{with } l_{i,j} = -\log \frac{\exp(\text{sim}(\tilde{z}_i, \tilde{z}_j)/\tau)}{\sum_{k=1}^{2n} \exp(\text{sim}(\tilde{z}_i, \tilde{z}_k)/\tau)}. \quad (2)$$

Here,  $\tau$  denotes a temperature parameter. We fixed  $\tau$  to 0.5 for all of our experiments and used the cosine similarity as *sim*. By the definition of the loss, the overall learning objective of SimCLR is to maximize the similarity between positive pairs while minimizing the similarity to all negative examples.

We implemented adaptations as suggested by Chen et al (p. 6).<sup>42</sup> The encoder *f* consists of a base CNN extended by an additional linear layer with variable output dimension, batch normalization, and ReLU activation. Likewise, a deeper projection head *h* was employed by adding 2 additional linear layers, including batch normalization and ReLU. The output dimension value of *f* and the hidden dimensions of *h* resulted from a hyperparameter optimization, but the output dimension of *h* was fixed to 128.

Each SimCLR model was trained using a LAMB optimizer and a 1-cycle cosine decaying learning rate. We investigated other optimizers, but LAMB led to the best results. As large batch sizes are crucial for the SimCLR method, we trained with a batch size of 512. We utilized Adaptive-ASHA to obtain the final parameters of the network dimensions and optimizer settings. We appended an overview in Appendix A.2. A k-nearest-neighbor classifier was regularly calculated on a validation set in the embedding space during the training process. The metric did not influence the training but functioned as a proxy variable to observe the learning process.

### SimSiam

SimSiam uses the self-supervised learning paradigm implementing a non-contrastive approach, i.e., it does not repel the representations of negative pairs in the learning process.

The method implements the learning paradigm by comparing the output vectors of 2 views of the same input. Again, the views are created by an augmentation stack, including random operations. For each view, an embedding (encoder output), an encoding (output of an additional MLP), and a similarity of these 2 vectors are computed. However, the similarity is calculated between the embedding of one view and the encoding of another.

Formally, for an image *x* from an unlabeled dataset *D*, we created a pair of views ( $v_1, v_2$ ). For each view  $v_i$ , we computed the embedding  $z_i = f(v_i)$  and the encoding  $\tilde{z}_i = h(f(v_i))$ . In this study, the encoder *f* consisted of a deep CNN with an additional 3-layer MLP network. The last batch-norm layer of the MLP was dropped to match the network structure of PAWS. The only difference is the output dimension of the encoder, i.e., the embedding size. The projection head  $h^6$  was left unchanged, a 2-layer MLP with a bottleneck structure and a hidden dimension of one-quarter of the embedding size as recommended in Chen and He.<sup>6</sup>

The optimization goal is to maximize the similarity, measured by a function *sim*, between the embeddings and the encodings as represented in the loss function

$$L = -\frac{1}{2} [\text{sim}(\tilde{z}_1, z_2) + \text{sim}(z_1, \tilde{z}_2)], \quad (3)$$

The overall minus sign converts the maximization of the similarity into a minimization of the given loss function, and the *cosine similarity* was used

<sup>6</sup> Notation was adapted for reasons of consistency. In the original publications of SimSiam, the mapping *h* is denoted as a prediction head, and the modification of the encoder MLP as projection head.

as the similarity measure *sim*. An essential aspect of the method is that only the gradients of the encodings  $\tilde{z}_i$  update the network where the embedding  $z_j$  are considered constant during the update step, see Fig. 1.

In the training process, we studied the standard deviation of the  $\ell_2$ -normalized outputs to check if the embeddings were collapsing to a constant vector and regularly calculated the accuracy of a k-nearest-neighbor classifier as a proxy metric to monitor the progress. We employed an SGD optimizer, scaled the learning rate linearly with  $lr * \text{batch\_size}/256$  according to the original implementation, and adjusted with a 1-cycle cosine decay schedule. We performed a hyperparameter optimization using grid search here instead of ASHA. The final network parameters and optimizer settings are reported in Appendix A.2.

### Finetune protocol

To increase the comparability of the pretrained model performance across the methods, we used a fixed protocol unifying the training settings of every finetune process, i.e., the downstream tasks. We recognize that, in general, this approach may result in weaker performance on the task than is individually possible for each classifier, especially if compared to published state-of-the-art results. However, disentangling and inter-comparing the effect of the pretrained models is essentially impossible if each experiment is individually optimized.

In contrast, the fixed finetune process is a reasonable basis for a fair comparison, provided that the results are compared relatively between the evaluated methods. In addition, using this approach, the focus is clearly on the pretrained model in a controlled environment. Therefore, we consider this strategy justified.

Each downstream task was trained using the following training settings and network modifications. Based on the experiment, the model's head *h* was replaced by a reinitialized MLP head consisting of either 1 linear layer (logistic regression) or 3 blocks, each including a linear layer, a batch-norm layer, and ReLU as an activation function. The corresponding downstream task at hand defined the output dimension. The input dimension of the head was chosen according to the embedding dimension of the encoder. In the case of PAWS, where *h* is missing, the head was attached to the encoder instead. The MLP parts of the encoders, described in the related method sections above, are usually kept if not stated otherwise. Since this approach resulted in a stronger, more complex prediction head for SimSiam and SimCLR, we conducted an ablation study to investigate this further, where the MLP part is reduced to the initially proposed one. We report the results in Appendix C.3.

To ensure the performance comparability of the semi- and self-supervised pretraining, we trained a corresponding model in a fully supervised fashion for each encoder finetuning. In this case, the entire model was randomly reinitialized using the proposed strategy of He et al.<sup>38</sup> The training was performed with the same settings as the finetuning of the related encoder. Due to this process, we were able to calculate a benefit for each encoder that is not biased by the slight architectural differences of the encoder and thus ensures comparability.

We studied the encoders on tiny subsets of the training data, Section Downstream task data reduction. Since some datasets contained only a few hundred samples per class in such experiments, a fixed batch size of 32 was used for every training. In the extreme case of a 0.2% subset split of Kather, 1 epoch is about 180 samples only (20 images per class), for example. The training length, the number of batches trained, was chosen to represent 20 epochs on each dataset. In addition, an encoder warmup for roughly 4 epochs was implemented, i.e., parameters of the pretrained model were not updated during that period. The models were trained using an SGD optimizer with a learning rate of 0.005, a momentum value of 0.9, and a weight decay value of 0.0001. The learning rate was adjusted with a cosine decay. We applied a small augmentation stack to the training set suited for histopathological data (Appendix A.1). Despite rescaling to match the input size and data normalization, no test augmentation was employed. As an exception to the protocol, if a training utilized the entire

dataset, we trained with a batch size of 256 instead of 32 and adjusted the total batches trained to be equal to 20 epochs.

If not stated otherwise, we used a fixed set of seeds for each experiment's method/dataset combination to ensure equally random initializations again. Model evaluations are performed on fully trained models only to achieve consistent comparability across experiments, i.e., intermediate model states were discarded even if they performed better on the validation data during training.

## Experiments

We investigated semi- and self-supervised trained models in various experimental setups to gain more insight into the benefit of pretraining in the histopathological environment. In particular, we explored the stability of the training process, which is an essential factor for the reproducibility and applicability of a classifier in the medical field. In addition, we analyzed the encoder performance under the reduction of training data on different downstream tasks and compared them to a fully supervised approach. Finally, we examined the transferability properties of the encoders learned by the different methods upon in-domain shifts.

### Encoder initialization sensitivity

Within this experiment, we analyzed the classifiers for possible performance fluctuations. In critical domains like medical applications of deep learning algorithms, it is essential to be consistent in the results, and also for scientific work, traceability and reproducibility are crucial aspects. We trained 5 encoders for each method on the Kather dataset using different experiment seeds to study the performance fluctuations. Seeds were randomly sampled. After, we finetuned each encoder on the Kather downstream task 5 times, reusing the same seeds, resulting in 25 runs per configuration.

By fixing the seeds, it is ensured that for the encoder training, all CNNs were initialized consistently for all methods, and thus provided a fair and comparable starting ground. Accordingly, on the downstream task, the seeds ensured an equal initialization of the prediction heads and training settings (data sampling, batch sampling, etc.) on a method level.

In this setting, we limited prediction heads to logistic regression since it provides the most information about the performance of each encoder. However, we trained each configuration with and without updating the encoder weights, i.e., freezing the encoder during finetuning. If the encoder weights were updated, we trained only the prediction head for about 4 epochs before updating the entire model weights (encoder warmup). In addition, the finetuning was performed on a fixed 8% data split set to save computational resources. In this experiment, a ResNet50 was studied as a default architecture representative.

### Downstream task data reduction

One purported benefit of training networks based on an encoder is to achieve a competitive performance using less annotated data compared with solely supervised training. We finetuned encoders on subsets to examine this aspect, simulating the scenario of limited annotations being available.

Besides using the entire training set, we randomly sampled 3 notably smaller subsets for every dataset. In the initial investigation of Kather, we could not observe any significant performance changes before reducing the data to 8% of its original size. Hence, we used this breakpoint to create a first subset. Going further, we limited the samples to 0.8% and 0.2% of the corresponding datasets for the other splits. The resulting number of training data and average images per class are given in [Table 1](#). We ensured the subsets were fixed across all experiment runs to maintain comparability.

We trained PCam and Kather encoders with each method for the selected ResNet architectures on the entire datasets. Note that PAWS models have seen labeled data in the process, potentially introducing a bias if the encoder has seen more annotated samples than in finetuning. To counteract this risk, we calculated multiple encoders for PAWS with different sizes of

the support set  $D_{sup}$  matching the split sizes. If the finetuning is performed on an entire training dataset, we used the 8% split encoder version.

Each encoder was then finetuned on the downstream task of every dataset using the 4 training data splits with several configurations. We explored the regression head and 3-layer MLP option as a prediction head. Furthermore, we trained variants in which the encoder weights were either updated or frozen, i.e., finetuning the prediction head only. We used the same warmup strategy as before (4.1) in the configurations where the encoder was updated. Across all datasets, network variations, and configuration settings, we trained 288 classifiers per method. Since we calculated a fully supervised complement for each encoder finetuning, the number effectively doubles.

We monitored the accuracy, the f1-score, and the expected calibration error (ECE) metric to measure the performance on the downstream tasks. In addition, we calculated a benefit for each of those metrics except the ECE. We define the benefit as the difference between the performance of a finetuned encoder and the solely supervised trained counterpart. Accordingly, the benefit reflects the absolute difference (gain or loss) in each experimental setting and allows us to compare the performance gain of different methods.

### Histopathological features and transferability

An essential assumption is that self-supervised methods learn underlying fundamental patterns from non-annotated data.

We investigate this assumption on a macro-level by reexamining the data generated by the experiments conducted in 4.2. Hereby, the experiment runs finetuning an encoder to classification task unrelated to its training data were significant. Hence, their performance metrics provide information about the transferability of the feature space, i.e., to what extent embeddings learned on a specific tissue can be valuable for tasks using other tissue types.

Therefore, we examined specific encoders with basic explainable AI methods to gain a deeper insight into the feature space. Grad-Cam<sup>43</sup> and Guided Backpropagation<sup>44</sup> maps were created for randomly selected images of the Kather test set. In consultation with pathologists, we evaluated maps to determine which morphological structures were relevant for each classifier. However, since we have performed this evaluation exclusively for a few examples, we include the results, even though promising, in [Appendix C.2](#). Finally, we computed t-SNE mappings of these encoders to explore how the datasets were already clustered in feature space, also reported in the [B.1](#).

## Results and discussion

### General observations

Before discussing the experimental results, we report our experience gained in applying the methods in the histopathological domain.

The PAWS approach was most sensitive to the choice of training parameters. To identify suitable parameters, a hyperparameter optimization was needed for every support set split. The ASHA algorithm, covering a larger search space than Grid Search with equal computational resources, worked best with PAWS. In the initialization sensitivity experiment, even with optimized parameters, 2 of 5 encoder trainings collapsed by changing the random seed only, i.e., different initial weights. Thus, we had to change the random seeds twice to train all models. However, overall, we observed that PAWS adapted sufficiently to the domain. But among the tested methods, it was the most computationally intensive.

Discovering a parameter setting for the SimSiam training was initially challenging. When using sophisticated hyperparameter search algorithms such as ASHA, we observed that this approach leads to corrupted training processes regardless of the chosen thresholds. The loss drops close to the minimum within a few iterations, but the proxy metrics did not reflect any improvement. Also, monitoring the standard deviation of the normalized outputs did not help with this problem either. We observed cases

where ASHA would identify parameter settings leading to non-collapsed encoders with no meaningful representation. Thus, we conclude that deficient representation at local minima could be a general issue (even for the class of NCSL methods) and a subject of further research.

Finally, we used Grid Search for SimSiam with a significantly narrowed search space, leading to appropriate parameter settings. However, the determined parameters were stable across the experiments regardless of changing the architecture or dataset. During our exploration of the role of the augmentation stack, we learned that it is another critical factor for learning meaningful embeddings, particularly strong color manipulations were essential.

SimCLR was the only method chosen where we have not encountered any reportable difficulties when applying it to the histopathological domain. However, large batch sizes are required for optimal performance, so the appropriate computational resources are a prerequisite.

#### Encoder initialization sensitivity

We investigated how much the weight initialization affected the encoder training and its performance. Therefore, encoders were trained and finetuned on Kather with a fixed setting but a changing random seed. This yielded a total of 25 data points for each method. The experiment was repeated without updating the encoder weights in the finetune process, i.e., finetuning the prediction head only. We report each method's mean accuracy and f1-score and their respective standard deviations in Table 2.

The results of SimCLR slightly outperform those of SimSiam in the frozen setting. The standard deviation values are consistently low regardless of metric and dataset. Indeed, comparing these methods, this is expected since the contrastive approach is a more stable learning strategy and likely produces a more distinct clustering in the feature space. However, interestingly, this performance advantage is lost when finetuning the encoder weights. As a result, the performance of SimCLR barely changes, while SimSiam improves both metrics and their standard deviations and exceeds the other methods.

In contrast, the metrics of PAWS are predominantly lower, and their standard deviations are consistently above those of the other methods. Especially in the frozen setting, the performance differences on the test data are substantial with a gap of at least 8.17% (f1-score of SimSiam) and standard deviations above 2.15. We conclude that the PAWS encoders learned much weaker embeddings, which was unexpected since it partially exploits annotated data. A probable explanation is the choice of the support set  $D_{sup}$ . When exploring the methods, we observed that the choice of  $D_{sup}$  strongly influences the performance of PAWS. Finetuning the encoder significantly closes the performance gap and elevates the results of PAWS to the ones of SimCLR.

Next to the apparent performance differences, we also noticed that the standard deviation of, for instance, the test set accuracy ranges from 0.29% to 2.60%, hence changes by an order of magnitude when comparing the different methods. We investigated the accuracy standard deviation further to explore the origin of the corresponding values. Hence, we computed for all 5 encoders of each method, i.e., fixing the pretraining seed, the standard deviation of their 5 finetune runs. Averaging these encoder-wise standard deviations for each method gives us an estimate of the fluctuations resulting from the finetuning. Vice versa, we received the pretraining estimate by averaging the finetune-wise standard deviations obtained from

**Table 3**

Summary of the mean standard deviation (std) of the resulting accuracy on the validation (val acc) and test set (test acc), computed for the varying seeds in the pre-training and finetuning where the encoder weights were not updated during finetuning.

Method	Val acc std		Test acc std	
	Finetune	Pretrain	Finetune	Pretrain
PAWS	0.70%	2.44%	1.06%	2.34%
SimCLR	0.18%	0.25%	0.26%	0.21%
SimSiam	0.18%	1.57%	0.18%	0.41%

the 5 distinct encoders sharing the same finetune seed. We restrict ourselves to the frozen encoder setting and report the estimates in Table 3.

In cases with a significant standard deviation, we find that the major contribution resulted from the pretraining. Thus, genuinely different encoders resulted from the different initializations of the pretraining with the semi- or self-supervised method. In contrast, the variations resulting from the downstream task are mostly minor, which post-validates our finetuning scheme as relatively stable. The only exception is the PAWS method, where the finetuning also led to notable variations. This observation likely resulted from the weaker embeddings learned in the pretraining such that more substantial performance variations appeared in the finetuning process.

In summary, PAWS generated the weakest encoders, showing the most considerable performance fluctuations. Moreover, these fluctuations persisted even when such an encoder was finetuned in this experiment. The other methods yield a stable performance all over (SimCLR) or at least after finetuning (SimSiam). However, the findings were somewhat surprising since we expected methods that receive more information in pretraining to be more powerful. Hence, PAWS should be the most reliable approach, followed by SimCLR and SimSiam. Especially with SimSiam, we were concerned that the naive NCSL approach would have difficulties separating the "similar-looking" histopathological data. On the contrary, it is noteworthy that the method achieved almost comparable results to currently published ones<sup>24,30,45</sup>, even though the training conditions were not optimal due to the unified finetuning protocol and the usage of merely 8% of the training data. However, observation indicates that SimCLR is the most stable method concerning the investigated different training initializations.

#### Downstream task data reduction

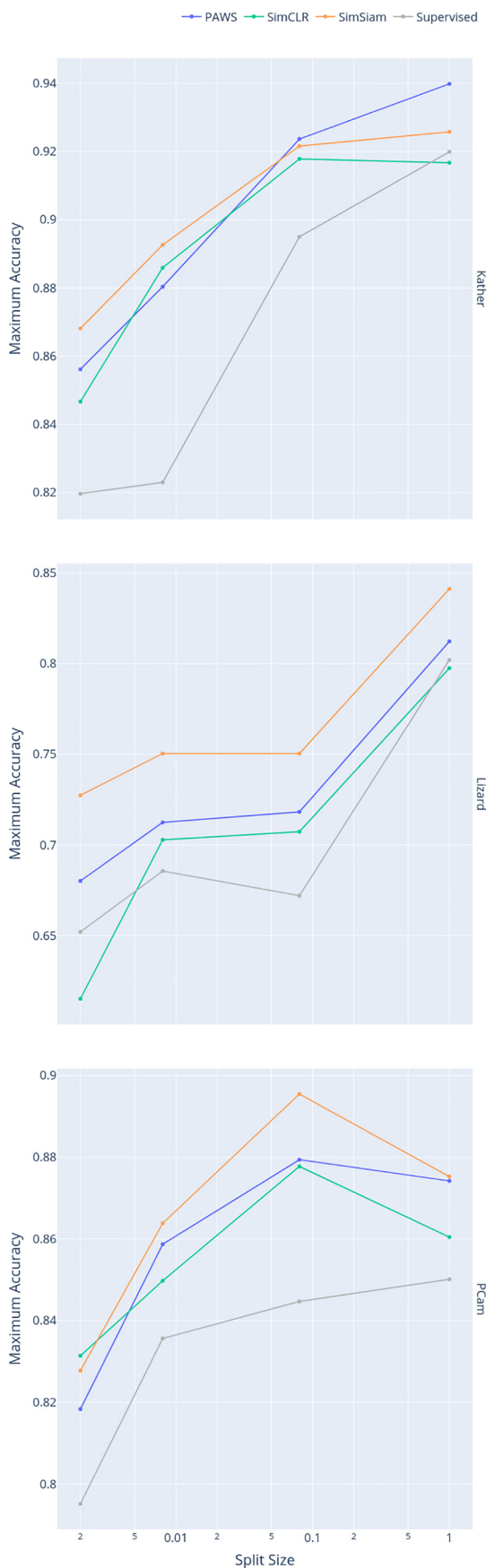
We studied the performance development of the encoders depending on the size of the training dataset. Therefore, PCam and Kather encoders were trained using the proposed methods with different ResNet architectures. Each encoder was finetuned on every dataset classification task using multiple data splits (100%, 8%, 0.8%, and 0.2%) and exploring several training configurations by changing the prediction head complexity and freezing the encoder weights. Furthermore, for each encoder finetuning, a randomly initialized network counterpart was trained complementarily. Since the calculated performance metrics, f1-score and accuracy, demonstrated analogous results, we limit ourselves to discussing the accuracy here. We append all representations using the f1-score in Appendix B.2.

**Table 2**

Accuracies and f1-scores averaged over 25 results on the Kather classification task. Their standard deviation is noted in brackets. Using a logistic regression prediction head, several ResNet50 encoders were finetuned on the 8% data split. The training runs differed only by the initialized weights.

Method	Acc (val)	Acc (test)	f1 (val)	f1 (test)
PAWS (frozen)	82.80% (2.52)	79.25% (2.60)	81.87% (2.68)	75.28% (2.15)
SimCLR (frozen)	92.67% (0.28)	90.45% (0.29)	92.81% (0.28)	87.02% (0.44)
SimSiam (frozen)	85.13% (1.57)	89.42% (0.44)	84.20% (2.09)	83.45% (0.97)
PAWS (finetuned)	91.14% (1.67)	90.52% (1.15)	91.21% (1.72)	87.23% (1.16)
SimCLR (finetuned)	94.10% (0.45)	90.57% (0.70)	94.19% (0.45)	87.01% (0.96)
SimSiam (finetuned)	95.47% (0.27)	91.47% (0.52)	95.54% (0.26)	88.53% (0.60)





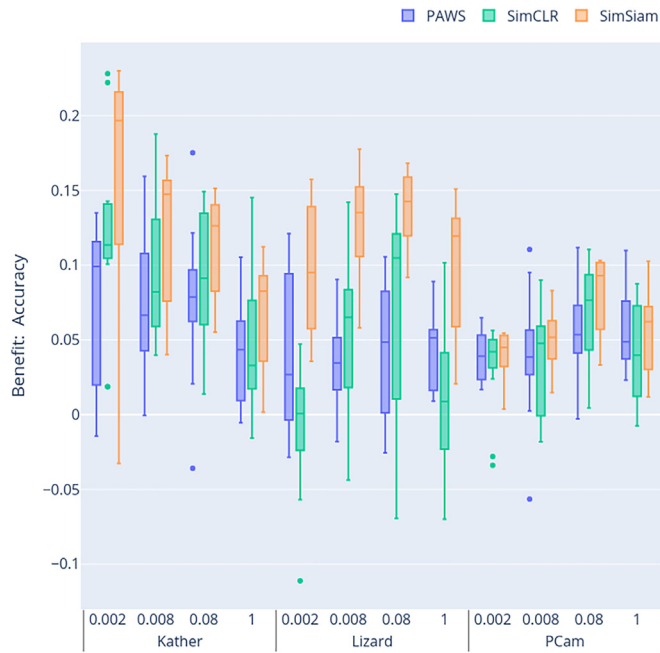
We report each method’s maximum accuracies on different dataset splits in Fig. 2. Note, as we analyzed the completely finetuned networks here, each maximum was picked from a pool of 12 training configurations in which the CNN architecture, MLP complexity, and pretraining dataset were altered. Since complementary supervised runs exist for every method, such maxima were selected from 36 data points. On Kather, the performance consistently decreases if the available training data is reduced. A widening gap between the solely supervised trained models and finetuned encoders is discernible and suggests that the smaller the annotated dataset becomes, the greater the relative advantage of pretraining. The SimCLR performance on the 100% split is an exception to this general behavior, which is not reflected in the f1-score. However, both observations align with our anticipations of the experimental setup and are consistent with recently published results.<sup>30</sup> Overall, the methods yield minor differences in maximum accuracy using lower training splits, and none is consistently superior.

We observe this somewhat differently with the Lizard data. SimSiam performed consistently better than the other methods, while SimCLR did not consistently exceed the supervised runs. Further, an apparent widening of the performance gap between the finetuned encoders and the purely supervised runs is missing. For SimSiam and PAWS, it seems relatively constant. One likely explanation is that most encoder embeddings were irrelevant for the task or tissue since Lizard was absent in the pretraining. Note that this indicates some encoder transferability limitations, which we will investigate further in the following section. The PCam results rank between the observations for Kather and Lizard. The training data reduction aligns mainly with a performance drop but not a widening performance gap. One oddity is that the peak performance was generally achieved at a training split of 8% for each encoder finetuning, rather than for the entire dataset. This observation may indicate that the finetuning could not fully converge on the large PCam dataset within the finetune protocol. Here, too, SimSiam predominantly outperforms the other methods slightly.

By comparing the maximum accuracies in Fig. 2, we simulated a real-world case of running multiple experiments with various architectures and selecting the best performing. However, we compared data resulting from multiple configurations and thus discarded information about performance differences of encoders in single experiment runs. To investigate this further, we calculated the accuracy benefit for the same data as used before, i.e., runs with finetuned encoder weights. As a reminder, we defined the benefit as the performance difference of a specific metric between a finetuned encoder and its purely supervised trained counterpart. Hence, the benefit estimates the gain or loss expected through an encoder. We summarize the accuracy benefits in Fig. 3.

A prominent observation from the benefit evaluation is that SimSiam outperforms the other methods, especially on Kather and Lizard, where the benefit disparity was significant. It was the only method close to purely positive benefits. Yet, on Kather, each method had a considerable benefit, and, like before, we observed the expected behavior, i.e., an increasing benefit with decreasing training data, only here. In contrast, substantial positive benefits were also evident on Lizard, but not following the same pattern and also with apparent method differences. For example, the smallest median of SimSiam was close to 0.1, while SimCLR was barely positive. Ignoring the data points for the entire training set, the benefits consistently decreased with the data reduction. That the performance relies primarily on the amount of data supports the findings in Fig. 2 and strengthens the assumption that embedding transferability is an issue between the tissue types. This effect was also observable in PCam, which is curious since it was used in pretraining. Further, the benefit variance

Fig. 2. Top accuracy of each training data split (0.2%, 0.8%, 8%, and 100%) divided by dataset. As the training data is reduced, the classifier performance generally decreases; an exception is the 100% PCam split. The finetuned encoders mostly attain higher accuracies than purely supervised training.



**Fig. 3.** Test set accuracy benefits for each dataset and split size for runs finetuned encoder weights. Overall, the gain with SimSiam compared to PAWS and SimCLR is significant for Kather and Lizard.

was, in general, significantly more minor compared to the other datasets. Therefore, the finetuned PCam and Kather encoder yielded relatively stable results. We suppose that the Kather encoder embeddings benefit PCam but not Lizard. Overall, the benefits investigation confirms some observations made in Fig. 2 on the individual experiment level.

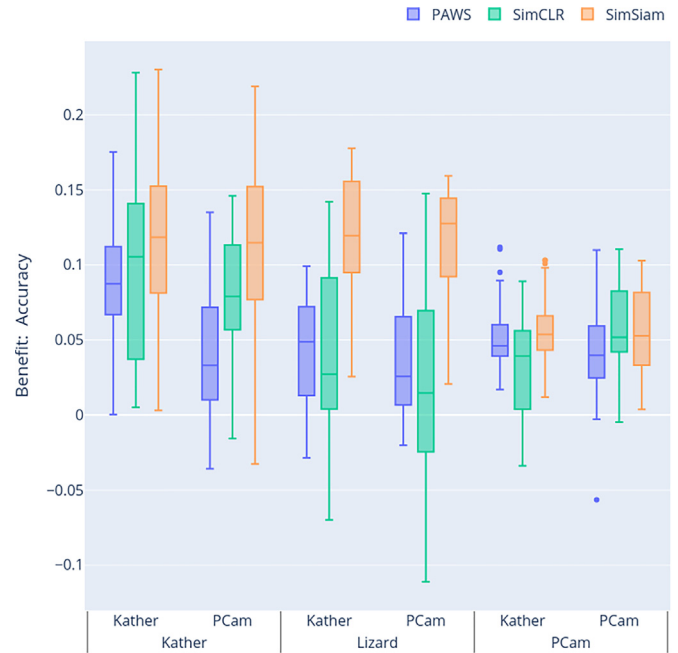
To disentangle the influence of the different finetuning configurations, we analyzed the benefit distributions separated by network architecture, prediction head, and finetuning of the encoder weights. We append a detailed discussion in Appendix C.1, but the findings indicated that the usefulness of pretraining increases with the network complexity, i.e., architectures with more parameters benefit more. Similarly, SimSiam mostly exceeded the other methods in settings where the encoder weights were finetuned, consistent with the previous observations. However, significant instabilities were observed for SimSiam if the encoder was fixed during the finetuning. A closer examination of the apparent oddity revealed that it is an artifact induced by the unified finetune protocol (Appendix C.3). Further investigating runs with fixed encoders, we find that SimCLR produces the most stable, beneficial embeddings, which aligns with previous observations.

In summary, the expected behavior for reducing the training data available was only observable for Kather. This may be because the finetune protocol was initially adjusted to this dataset. But when averaging over the experimental runs, the results showed that building upon a domain encoder was generally advantageous. This strengthens the assumption that all methods generated encoders with beneficial embeddings for the data domain, while SimSiam performed best in the comparison group.

#### Basic histopathological features

To investigate the in-domain transferability of the encoder embeddings, we reexamined the results generated in 4.2. We studied the benefit distribution of the finetuned encoders based on their training data on every downstream task. Fig. 4 summarizes the findings.

Analyzing the Kather finetuning results, we observed that all methods performed consistently better if their encoders were trained on this dataset. We assume an encoder learns the underlying patterns in its pertaining data. Therefore, the observation was rather logical and supports this assumption.



**Fig. 4.** Benefits distribution of finetuned encoders depending on training and downstream task data. The benefit represents the accuracy gained (the mean is noted in the setting label) by comparing the encoder's performance with its supervised counterpart. SimSiam generally outperforms the other methods.

The benefits generally dropped when the finetuning was based on the PCam encoder. However, this was anticipated since the dataset is built upon another tissue type. Though, the methods decreased to a significantly different extent, which was indeed unexpected. Thus, we suppose the encoder embeddings differ considerably, with some learning more features transferable in-domain than others. Method-wise, SimSiam yielded relatively stable results across both encoders while consistently surpassing the other methods. Followed by SimCLR, which was almost on par with SimSiam using the Kather encoder. Even though the median and average benefit of PAWS were positive, the method performed weaker in comparison. However, concluding whether such embeddings represent fundamental histopathological patterns or universal features is challenging. Therefore, we used basic XAI techniques to examine the models on some Kather samples as discussed in Appendix C.2. We found qualitative evidence that the classifiers targeted pathologically relevant tissue structures. Although SimCLR produced the most consistent maps, this was genuine for all methods. Indeed, this conveys the presence of domain-specific embeddings.

Examining the PCam downstream task, we observed a somewhat equivalent behavior, with SimSiam exceeding SimCLR, which in turn outperformed PAWS. Interestingly, the median and the average benefit were higher for PAWS using an encoder trained on Kather. Nevertheless, the differences were minor since the second and third quantiles strongly overlap, and the higher average benefit could result from some positive outliers. Overall, the differences were minor between the methods regarding their general performance and the influence of the encoder training data. This finding contrasts with the one made before on the Kather downstream. We suspect this is a direct consequence of the datasets and can be explained accordingly. The Kather dataset is built upon colorectal cancer and normal tissue. The evenly distributed 9 classes represent almost distinguishable tissue structures from which the classification task is derived. This structural distinction is not reflected in the design of PCam, which consists of randomly selected samples of sentinel lymph node sections that may or may not contain metastatic tissue. Thus, the apparent benefit decrease using a PCam encoder on Kather is explainable. In contrast, the Kather dataset may include helpful information for the PCam task, i.e., cancer and normal tissue, even if from another tissue type.

We chose the Lizard dataset since the intersection of the tissue sections included in the dataset with PCam, and Kather is small. In addition, its classification task, detection of cell nuclei, is, in fact, different from that of the other datasets. Therefore, Lizard represents a proxy metric for learning fundamental histopathological features or at least valuable universal features that work across tissues. Under this assumption, we observed that the feature transfer was not optimal for PAWS or SimCLR, regardless of the encoder training data. SimSiam yielded remarkably better performance, significantly surpassing the other methods. This observation aligns with the findings of Fig. 2 and Fig. 3. However, across all experimental setups, the general performance achieved and benefits gained on the Lizard dataset were mainly inferior to the other datasets. This is strongly supported when studying the experiment runs with a frozen encoder (e.g., Fig. C.2) in more detail. Note that the influence of the initially learned embeddings is maximized in such runs. The benefit distribution was either decreasing or relatively constant, depending on the method, compared to finetuned encoder runs. Hence, the learned embedding was as valuable as those resulting from a freshly initialized network for Lizard. The difference between Lizard and the pretraining datasets seems too pronounced. Accordingly, training encoders on specific datasets seem insufficient to represent the entire histopathological domain.

In summary, we found evidence that the encoders produced embeddings transferable in-domain but with limitations only. The observations suggest that the embeddings lose value while the downstream task data shifts from the pretraining data. Hence, training with a diverse dataset, including various tissue types and structures, is likely a requirement to produce encoder models applicable across the domain. We identify the training of such models as a further research topic, including a closer investigation of the embeddings.

## Conclusions

We have compared a cross-selection of state-of-art semi- and self-supervised learning methods in the histopathological domain. Therefore, we trained encoders with PAWS, SimCLR, and SimSiam on public datasets and evaluated them within a unified framework on several domain tasks.

The findings suggest that pretraining generally has a positive effect, measured by the average benefit and absolute performance, compared to a purely supervised training approach. This effect is more significant for complex network architectures and persists in insufficient training data regimes. Thus, less annotated data is needed to achieve comparable performance if building upon a domain encoder. Furthermore, there is evidence that such models learn underlying domain-specific features applicable across tissue types. Regarding this, initial explorations of encoder models using GradCam are promising, revealing that classifiers targeted structures of interest from a pathological perspective. Yet, according to observations, feature transferability is limited and dependent on encoder training and downstream task data divergence. Accordingly, we identify the relation between training data and resulting feature space as a critical topic of further research to build a general domain encoder. We recommend building a neural network classifier on an in-domain encoder model, especially if less annotated data is available.

Therefore, we notice a necessity to expand the sharing of domain models, i.e., a public histopathological model zoo.

As a semi-supervised approach, PAWS is the only method that uses partially labeled data in the training process. This additional information showed no advantage in this work since PAWS could keep up on average but mainly achieved significantly lower values than the other methods. Also, we experienced PAWS results as the most sensitive regarding hyperparameter settings and network initialization, which is potentially unsuitable in the medical field. Across the methods, PAWS has the highest computational resource consumption.

SimCLR is also a relatively computationally expensive method. It yielded the most consistent performance concerning all experimental settings, a valuable property in this domain. Moreover, it mainly achieved comparable results in its absolute metrics and benefit distributions, close to the best-performing method. SimCLR was straightforwardly applicable to the new data domain, i.e., identifying appropriate training settings was uncomplicated.

SimSiam demonstrated the lowest sensitivity regarding weight initialization in the finetuned encoder setting. In comparison, it achieved top evaluation metrics and gained the most relative benefit overall. Additionally, the method consumes, by far, the least computational resources. However, it was the most challenging method to train a performant encoder.

Note that a unified evaluation strategy limited the performance of every method. Though, it enhanced the inter-comparability of the encoders and, therefore, the investigated learning strategies, which is the primary aspect of this work. We assume each method will improve relatively if optimized for a downstream task. Under this assumption and based on the experimental result, we recommend trying SimSiam as a promising alternative or addition to the more established SimCLR, which seems the primary choice for consistent results in the histopathological domain.

## Funding

This work was supported by the Federal Ministry of Education and Research of Germany (BMBF, <https://www.bmbf.de/>) in the project deep. Health [project number 13FH770IX6]. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

## Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

We thank Tim-Rasmus Kiehl and Sebastian Lohmann from Charite - Universitätsmedizin Berlin for assistance with the medical interpretation of the results and for providing the perspective of pathologists.

We thank our colleagues from CBMI, Jonas Annuscheit and Patrick Baumann, for their input and discussions about our work.



## Appendix A. Encoder training information

### A.1. Augmentation stacks

All methods utilized augmentation strategies to create distinct views of a given image. In addition, to the augmentation stacks published with the associated methods, we implemented custom augmentation stacks suited for the data domain in a supervised setting, based on the findings of Tellez et al,<sup>41</sup> and Annuscheit et al.<sup>46</sup> Fig. A.1 illustrates transformations of a datapoint from the Kather dataset using all different augmentation stacks.

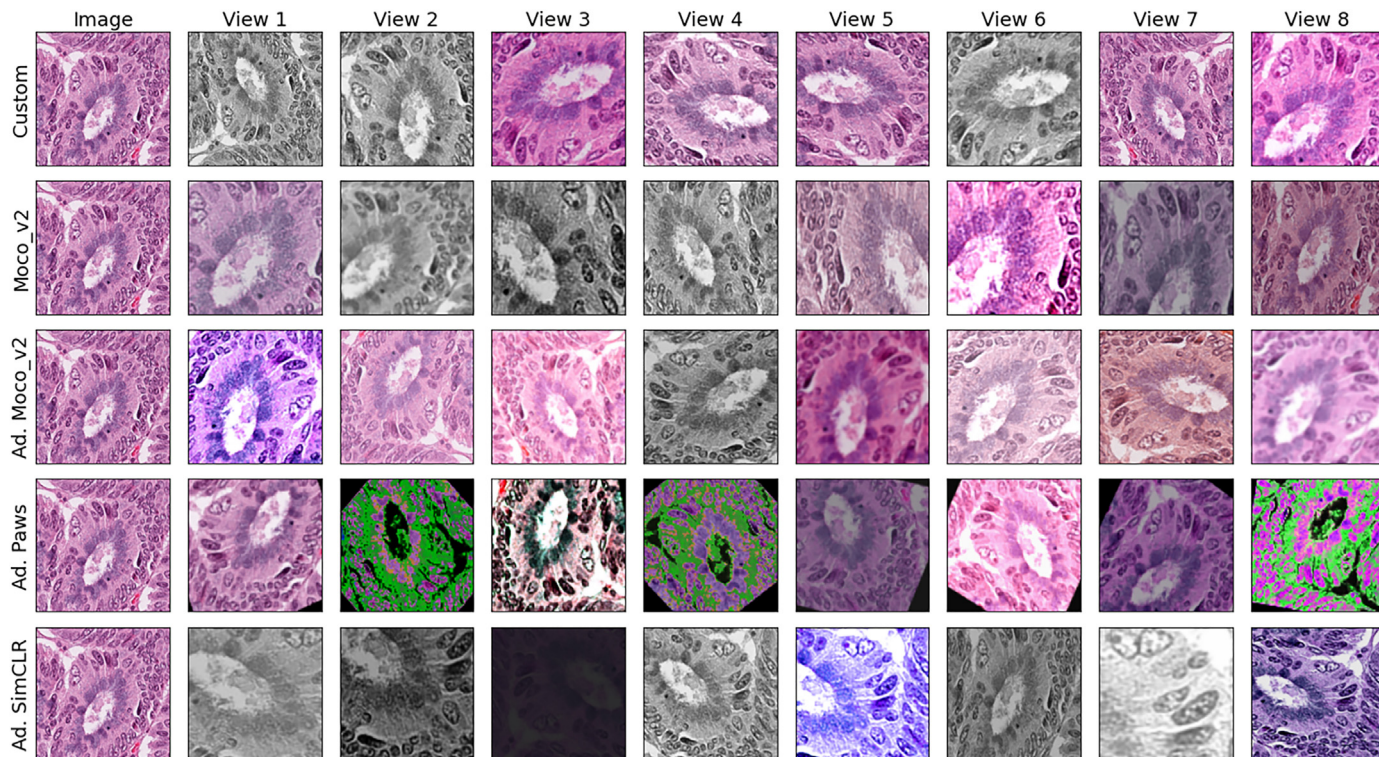


Fig. A.1. A Kather sample transformed with the different augmentation stacks used.

First, we trained SimSiam models with the simpler of our adapted histopathological stacks. The stack utilized only a slight color jitter because we expected too much distributional shift and problems in the learning process. Usually, color differences in the input data are a problem when modeling a classifier for histological tasks, e.g., stain normalization problems. Furthermore, gray scaling of the image was applied as a color transformation. As geometric manipulations random cropping, rotating ( $360^\circ$ ), and flipping (horizontal/vertical) are parts of the augmentation stack. The size of an image is correctly adjusted before and after the rotation transformation to avoid black borders. Each transformation is applied with a probability  $p$  with  $p \in [0.2; 0.8]$ .

We observed that models trained with the stack struggled to learn meaningful representations even though they did not collapse during the process. Hence, we attempted the moco\_v2 stack initially used for SimSiam training, leading to considerably better results. However, after further investigations, we found that the parameterization of the color jitter caused the problem. Therefore, we modified our stack to use the moco\_v2's heavier color jitter parameterization and also added the Gaussian blur to this method. This new custom stack, adapted moco\_v2, achieves similar to better results with the same training settings of SimSiam and SimTriplet learning approaches. For the other learning methods, we also adapted their augmentation stacks to our observations by adding geometric transformations or using different parameterizations but keeping the color manipulations of the respective method.

### A.2. Hyperparameter settings

Table Appendix A.2 reports the final hyperparameter used for the encoder training. In addition to the noted optimizer parameter, each one was wrapped in a 1-cycle cosine scheduling to decrease the learning rate. For the PAWS encoder trained on the Kather dataset, the same setting was used across the different split sizes.

**Table A.1**

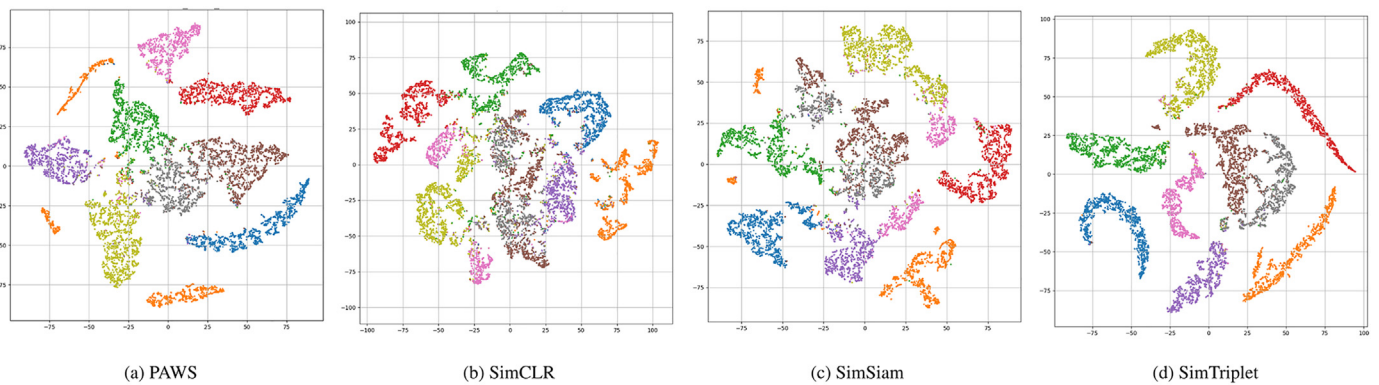
Summary of final hyperparameter settings used to train the different encoders. PAWS networks are divided by the size of the support set  $D_{sup}$  used. For brevity some column headings are shortened: ‘Size’ represents the batch size used; the optimizer column includes values for learning rate, momentum, and weight decay; column ‘Emb./MLP’ reports the values used for the encoder embedding size and the MLP hidden layer size.

Method	Data	Network	Epoch/Batch	Size	Optimizer	Emb./MLP	
PAWS	PCam	R18/.08	10/40940	64	LARS (0.467, 0.647, 72e-6)	128/-	
		R18/.008	10/40940	64	LARS (1.17, 0.364, 29e-6)	128/-	
		R18/.002	10/40940	64	LARS (2.41, 0.317, 31e-7)	128/-	
		R50/.08	10/40940	64	LARS (0.88, 0.48, 25e-6)	128/-	
		R50/.008	10/40940	64	LARS (1.39, 0.57, 18e-6)	128/-	
		R50/.002	10/40940	64	LARS (1.57, 0.6, 19e-6)	128/-	
		W28/.08	10/40940	64	LARS (2.231, 0.364, 87e-6)	128/-	
		W28/.008	10/40940	64	LARS (2.379, 0.49, 82e-6)	128/-	
		W28/.002	10/40940	64	LARS (0.784, 0.81, 32e-6)	128/-	
		Kather	R18	10/28120	32	LARS (1.772, 0.499, 28e-6)	128/-
			R50	10/28120	32	LARS (2.0, 0.5, 3e-4)	128/-
			Wide	10/28120	32	LARS (1.5, 0.6, 5e-4)	128/-
SimCLR	PCam	R18	23/12000	512	LAMB (6e-3, -, 1e-5)	1024/64	
		R50	23/12000	512	LAMB (6e-3, -, 1e-5)	2048/256	
		Wide	23/12000	512	LAMB (17e-4, -, 1e-5)	2048/128	
	Kather	R18	77/15000	512	LAMB (7e-4, -, 1e-5)	2048/256	
		R50	77/15000	512	LAMB (3e-3, -, 1e-5)	1024/256	
		Wide	77/15000	512	LAMB (3e-3, -, 1e-5)	1024/64	
SimSiam	PCam	R18	50/51200	256	SGD (0.5, 0.5, 1e-4)	512/128	
		R50	50/51200	256	SGD (0.5, 0.5, 1e-4)	2048/512	
		Wide	50/204800	64	SGD (0.5, 0.5, 1e-4)	512/128	
	Kather	R18	115/45000	256	SGD (1.0, 0.5, 1e-3)	512/128	
		R50	29/45000	64	SGD (0.5, 0.9, 1e-4)	2048/512	
		Wide	29/45000	64	SGD (0.5, 0.9, 1e-4)	512/128	

## Appendix B. Additional results

### B.1. t-SNE examples

Fig. B.1 shows t-SNE plots of the Kather validation dataset after an encoder training using a ResNet50. To what extent the investigated semi- and self-supervised methods can separate the validation data after training is remarkable. The plots indicate that the separation of some classes is difficult. Especially for the classes MUS (smooth muscle tissue, brown) and STR (cancer-associated stroma, gray), the separate clustering seems to be challenging. Both SimTriplet and PAWS achieve the sharpest separation. This results probably from the direct label information of the PAWS method and the close-patch strategy implementation of the SimTriplet method used here, which gives indirect information about the class affiliation. Also, the background class (BACK, orange), which combines quite diverse structures, appears problematic to unite in one cluster. With this mapping, SimTriplet and SimCLR manage to assign most data points to a joint region.



**Fig. B.1.** Example ResNet50 encoder t-SNE representations of each method on the Kather validation dataset. The representations suggest that the MUS (brown) and STR (gray) classes are the most difficult to separate in the learning process. The most structured clusters create the SimTriplet method (d) on the dataset.

### B.2. F1-score and ECE results

We append the corresponding f1-score representations (Figs. B.2, B.3 and B.4), of the accuracy ones reported in the main text. Although performance differences were observed for each method when comparing f1 score/accuracy metrics, they were small and non-significant. In particular, there was no difference in terms of methodological relations. This effect could be expected since the Kather, and PCam datasets are nearly balanced. Only the Lizard dataset contains some imbalances, which could have led to major differences in those metrics. However, these were also not observed, compare Fig. 4 with Fig. B.4 and Fig. 3 with Fig. B.3.

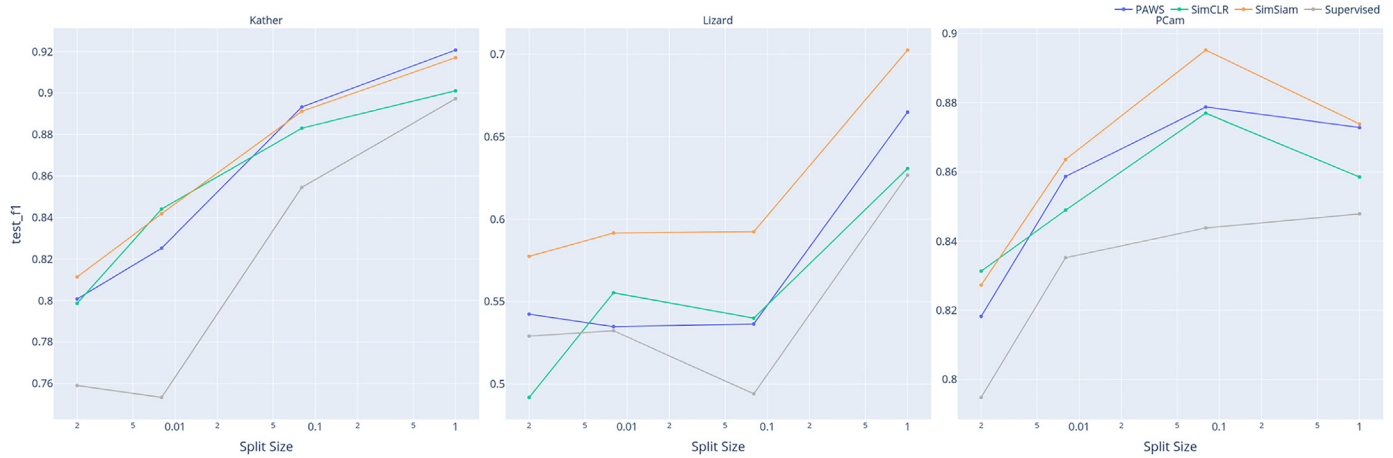
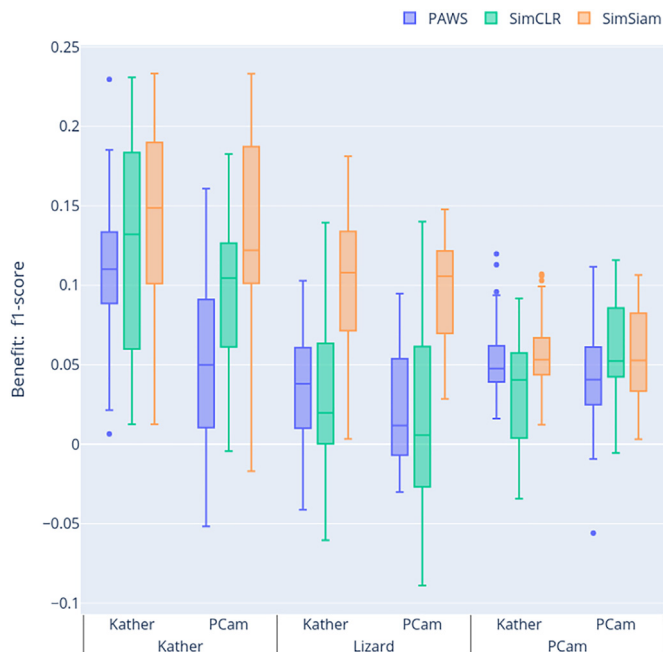


Fig. B.2. Top F1-score of each training data split (0.2%, 0.8%, 8%, and 100%) divided by dataset. Corresponding representation to 2.

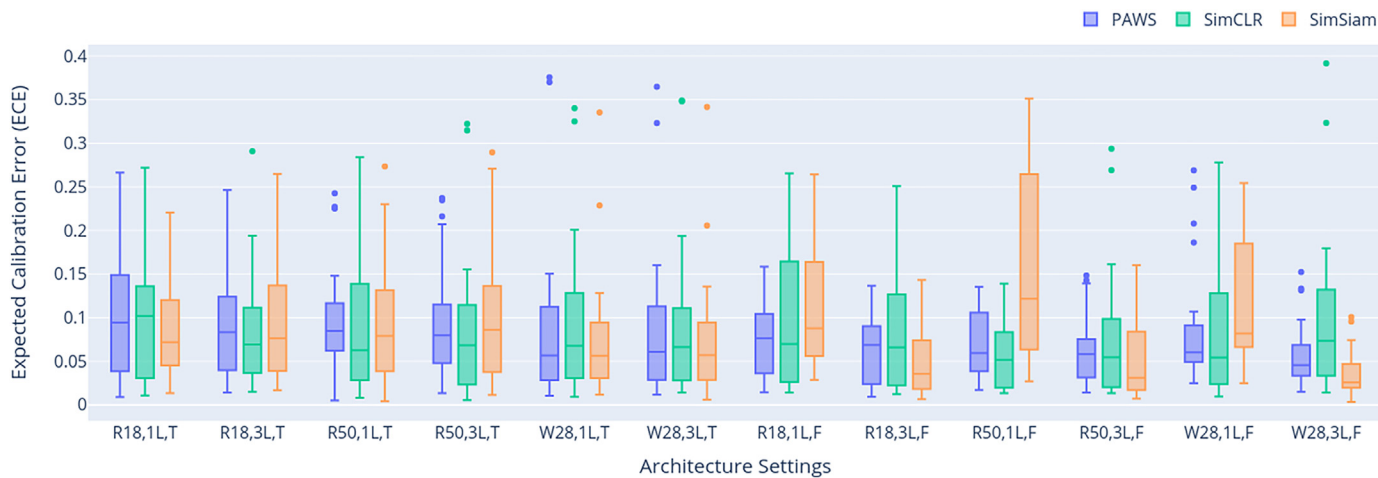


Fig. B.3. Test set f1-score benefits for each dataset and split size for runs finetuned encoder weights. Corresponding representation to 3.



**Fig. B.4.** F1-score benefit distribution of finetuned encoders depending on training and downstream task data. The benefit represents the accuracy gained (the mean is noted in the setting label) by comparing the encoder’s performance with its supervised counterpart. Corresponding representation to 4.

In addition, Fig. B.5 reports the distribution of the ECE (expected calibration error) across the different architectural settings. The models are usually relatively calibrated for experiments that finetune the encoder, but some outliers exist. The medians are between 0.05 and 0.1, which is reasonable for networks with that many parameters.



**Fig. B.5.**



### Appendix C. Additional analysis

#### C.1. Benefit of architectural settings and frozen encoder

In Sections Encoder initialization sensitivity and Downstream task data reduction, we have investigated the experiments finetuning encoders across the different datasets since this view is comparable to real-world training procedures and, thus, reflects practical usage. However, such finetunings blur the effective contribution of the encoder to the results, and inferences about learned domain-specific features are impaired. To disentangle some of those effects, we analyzed the benefits regarding the finetuning configurations more in-depth. Therefore, we calculated the benefit distribution separated by network architecture, prediction head, and finetuning of the encoder weights and summarized the findings in Fig. C.1.

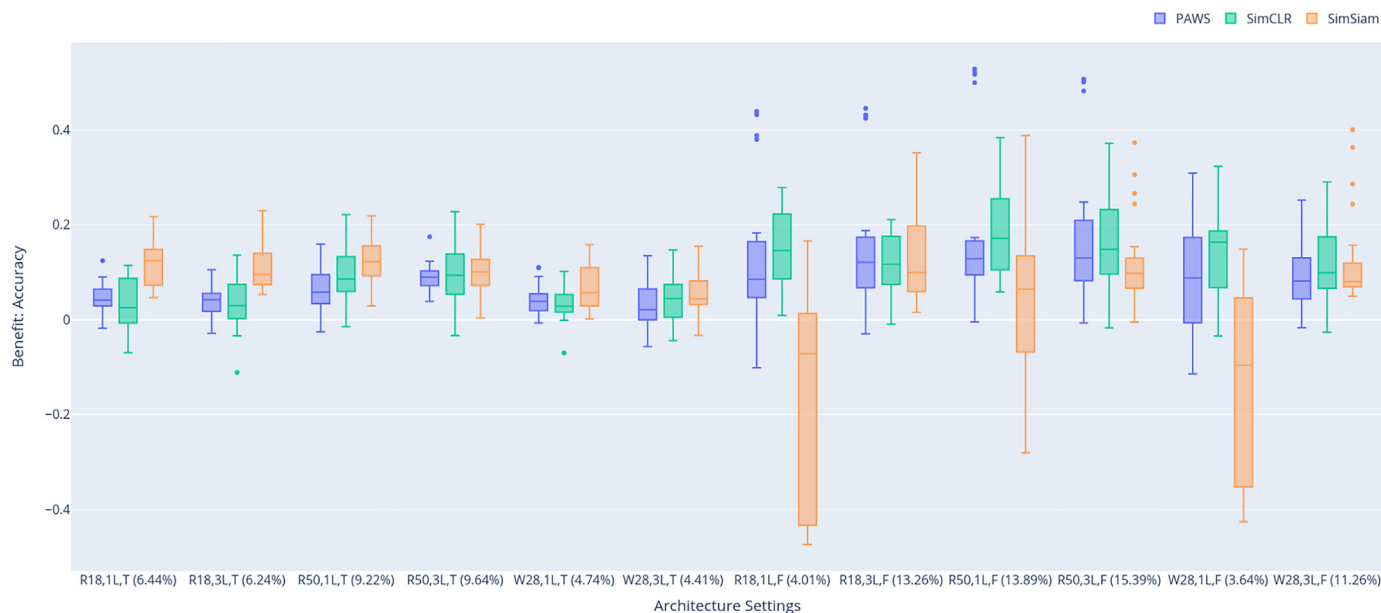


Fig. C.1. Benefit distribution over different network configurations. The benefit represents the accuracy gained/lost using a pretrained model instead of a randomly initialized network. The mean accuracy gain is noted in the label of each setting. For brevity, the labels are shortened (R18: ResNet18, R50: ResNet50, W28w2: Wide\_ResNet28w2, 1L: logistic regression, 3L: 3-layer MLP, T: encoder weights finetuned, F: encoder weights frozen).

We found only minor influences of the network settings in case the encoder weights were finetuned (indicated by a ‘T’ in the figure label). There is almost no difference when using logistic regression or a 3-layer MLP prediction head. However, we observed more minor benefits for all methods when using the Wide\_ResNet28w2 and for PAWS/SimCLR using the ResNet18. The most consistent positive benefit occurs for the ResNet50 across all methods, although SimSiam showed a similar positive benefit for the ResNet18. We expected an increased benefit for the ResNet50 since networks with increasing complexity also require more data to tune their many parameters appropriately. Therefore, pretraining should be more helpful for more complex architectures.

For finetunings with fixed encoder weights (indicated by an ‘F’ in the figure label), we observed evident changes. First, some SimSiam configurations were immediately conspicuous. The methods showed tremendous benefit fluctuations in the case of a logistic regression head. The median was negative in 2 of these settings, suggesting that the encoder impaired the classification performance. Even if this is possible, e.g., a (nearly) collapsed feature space, this observation appeared curiously based on the previous results. Therefore, we investigated these runs in more detail and found that adapting the learning rate was sufficient to improve these seemingly corrupted runs. Further, by dropping the encoder MLP part, this issue could be fixed, see Appendix C.3. The MLP encoder part represented an adaption of SimSiam to match the architecture baseline of PAWS. Hence, this observation was not caused by methodological issues but represents an artifact resulting from the unified finetune protocol. Notably, this clearly showed that only the CNN embeddings, as intended by the method, should be used as feature extractors to create performant embedding.

Besides the artifacts, we found the benefits generally increased but registered larger fluctuations in the results. Especially PAWS and SimCLR significantly improves in particular. On the one hand, this seems reasonable since it cannot be assumed that a randomly initialized encoder, like in the supervised runs, will produce linearly separable features without finetuning its weights, and the performance gap measured by the benefit, therefore, should be increased in such settings. On the other hand, the findings could also indicate that the encoder captured beneficial domain properties, resulting in a gain. More detailedly, we also notice that the average benefit increases and its fluctuations decrease if the logistic regression is replaced with a 3-layer MLP here. Again, the ResNet50 gained the most average benefit in each setting, supporting the previous observations that pretraining is more profitable for complex networks.

In Fig. C.2, we reported the corresponding representation to the finetuned encoder runs (Fig. 4), i.e., same settings but without updating the encoder weights during finetuning. SimSiam’s results here suffered from the same corrupt experiment runs, leading to tremendous fluctuations. However, we noticed that the benefit median generally increases, even for SimSiam, including the faulty runs. Again, this is expected compared to a supervised run but also indicates the presence of valuable features. Especially if the difference is substantial, like in some observations for PAWS or SimCLR. There was a notable distinction between the encoder of those two methods, i.e., in the case of the Kather-encoder, PAWS benefited more, while for the PCam-encoder, SimCLR surpassed the other methods. SimCLR repeatedly showed minor differences between the different encoders and overall fluctuations, supporting the finding that this method is the most consistent.



The drop behavior between the encoder training set and the downstream task did not differ in the case of the frozen encoder from the previous observations in [Section Downstream task data reduction](#). Hence, it supports the conclusion regarding the transferability of the encoder embeddings, which is an important one.



**Fig. C.2.** Accuracy benefit distribution of frozen encoders depending on training and downstream task data. The benefit represents the accuracy gained (the mean is noted in the setting label) by comparing the encoder's performance with its supervised counterpart. Corresponding representation to 4.

## C.2. Encoder GradCam and Guided Backprop illustrations

We investigated the topic qualitatively by examining a part of the resulting models from [Section Downstream task data reduction](#) using explainable AI methods, i.e., Grad-Cam<sup>43</sup> and Guided Backpropagation<sup>44</sup> on randomly selected examples from all downstream test datasets. Here, we only used encoders with the Wide\_ResNet28w2 for the experiment because it produced the largest feature map (14x14) of the 3 architectures before entering the global average pooling layer, using an input size of 96x96 pixels. The feature map size is crucial for the quality of resulting activation maps of the applied technics and consequently allows a more significant analysis. Furthermore, to maximize the impact of the encoder, we only evaluated models that used logistic regression as the prediction head and only considered models that were finetuned on an 8% split of the corresponding training dataset with a frozen encoder. Besides the resulting activation maps, we rendered the normalized version of the original image with a threshold of 0.6 to highlight the relevant content. Parts below the threshold are whitened, and higher activated areas are more color-intense, see [Fig. C.3](#).

We observed a fragmented selection from the purely supervised models. On the *normal colon mucosa* image ([Fig. C.3.a](#)), the activation reflected some structures of the original image, which was nevertheless remarkable for a randomly initialized network with a regression head only. However, the higher activations in [Fig. C.3.e](#) were somewhat arbitrary from a pathologist's point of view.<sup>47</sup> Crypts can be recognized (besides the shape) by the goblet cells. Typically, the nucleus is close to the basement membrane and the cytoplasm of the goblet cells towards the intestine. Thus, only marking the cell nuclei is problematic because these would still have to be distinguished from other cell nuclei, e.g., those of basal cells. However, selecting the cytoplasm of the goblet cells is more meaningful. From a pathological point of view, marking both aspects is the most significant. We observed all gradations in the activation maps of the pretrained models. Almost all models pretrained on the Kather dataset considerably recognized the cytoplasm of the goblet cells. SimSiam ([Fig. C.3.d](#)) also marked the cell nucleus structure, albeit slightly. For the 2 encoders pretrained on Kather and PCam, respectively, SimCLR showed the best generalization performance. Both variants identified parts of the cytoplasm and partially marked the same image content, comparing [Fig. C.3.c](#) and [Fig. C.3.g](#).

The second example ([Fig. C.3.i](#)) represented *adipose tissue*. Detecting fatty tissue can be tricky. The fat cells can quickly be perceived as background and useless information due to larger uniform areas. The supervised approach ([Fig. C.3.m](#)) targeted some nuclei with no discernible structure. The surrounding area had a much lower activation, and no explicit marking of relevant adipose tissue was noticeable. In comparison, all pretrained models consider this tissue type to varying degrees. One strategy is to mark the cell membrane and the adjacent adipose tissue. Good results were achieved here by several methods. However, SimCLR had consistent results for both encoders again, comparing images [Fig. C.3.k](#) and [Fig. C.3.o](#). The PAWS encoder, trained on the PCam dataset, followed a different approach and targeted the fat tissue directly. The PAWS results also contain a part of the cell membrane and other unimportant tissue in the context, see [Fig. C.3.n](#).

Even though it is a qualitative exploration, the results indicate that the evaluated learning approaches produce models already adapted to the histopathological data domain. Therefore, such models can generate reasonable representations from a pathologist's view to a certain degree. The observation also suggests that encoders trained on a particular tissue type can produce underlying histopathological features usable for another tissue type, i.e., downstream task. Both encoders occasionally selected the same image content or used the same histopathological features across the encoders, regardless of the chosen method.

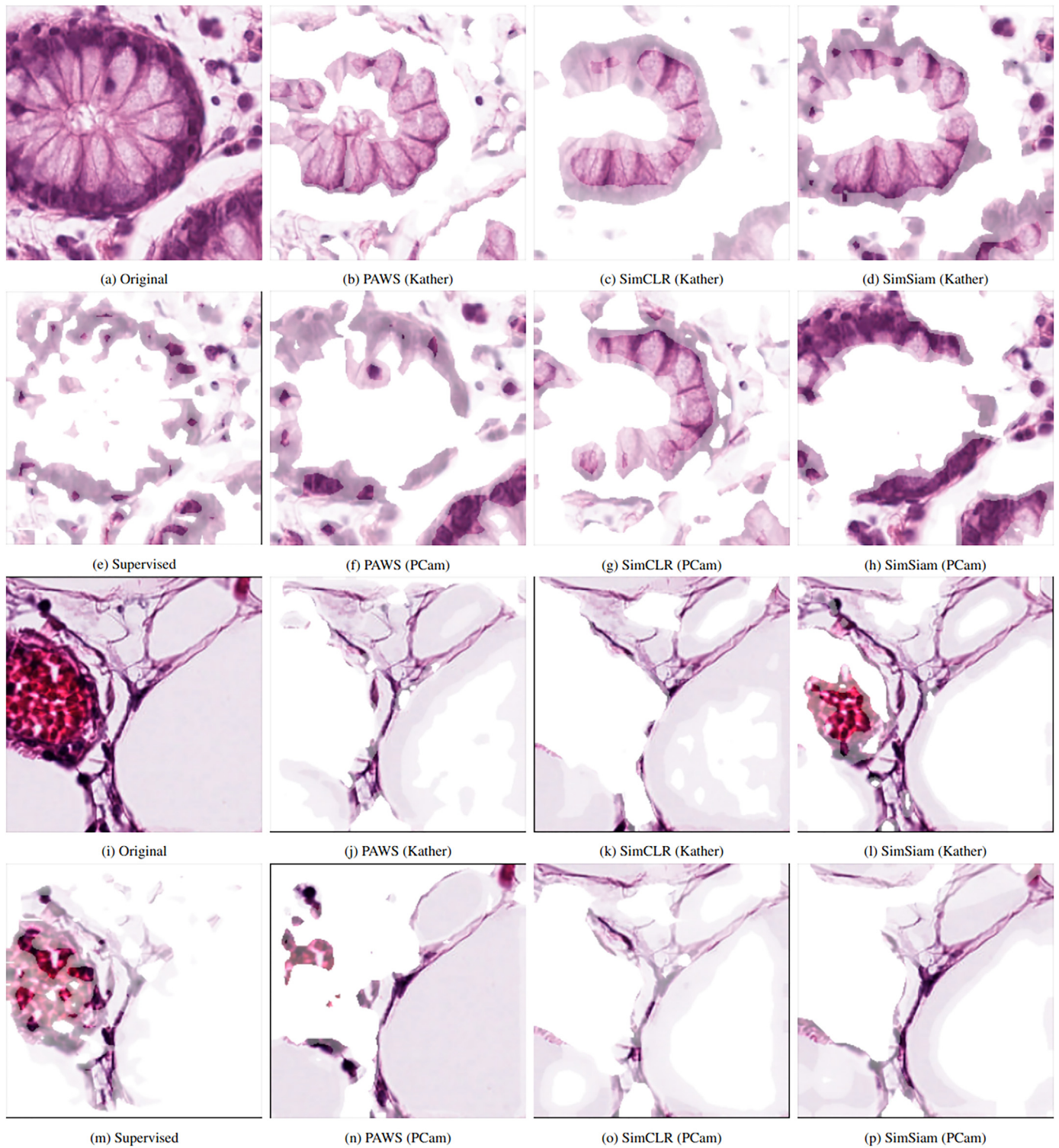


Fig. C.3. Exploration of tissue samples with the Grad-Cam method. Images (a) and (i) illustrate the original samples from the Kather data. All other images merge the original image with the activation map produced by Grad-Cam on a particular encoder, revealing the relevant parts for the classification decision. The encoder training data is noted in brackets after the method. Pretrained models make use of domain-specific structures but to a different degree, e.g., selecting goblet cells (b)–(h) or targeting fat tissue (j)–(p).

Overall, the experience applying both methods was that GradCam marked medium-sized image regions of interest based on feature activation maps from the CNNs' last convolutional layer, while Guided Backprop marked relatively small local pixel regions, given the nature of the method. Therefore, GradCam worked more closely with a pathologist's workflow when making classification decisions and showed a decision-making process based on contextual regional information. For Guided Backprop, it seemed that instead of medium-sized regions, strongly H&E colored pixels are marked, which might not always be of interest for a specific classification. Especially for fully supervised experiments, meaning experiments with no feature-based pretraining, this appeared to be the case. Even some Guided Backprop maps that looked like noise were no exception in this setup. However, this problem softens for semi-supervised experiments, and Guided Backprop showed far better maps that looked reasonably congruent to GradCam maps, underlying the importance of feature-based pretraining. We suggest that, when using XAI methods to analyze the feature space of encoder models, multiple strategies should be used and discussed with domain knowledge experts.

### C.3. MLP ablation study

In the original publications of SimCLR and SimSiam, only the base network of the encoder  $f$ , i.e., without the modified MLP part, is used for further downstream training. To preserve the comparison with PAWS, this was changed in the study. Hence, we also performed all experiment runs of the data reduction experiment with the intended implementation of the methods to examine the influence of these adaptations. We limit ourselves here to report one representation,<sup>7</sup> which can be used to summarize the results of the ablation study. Fig. C.4 illustrates the distribution of accuracy benefits for the different training configurations (corresponding to Fig. C.1).

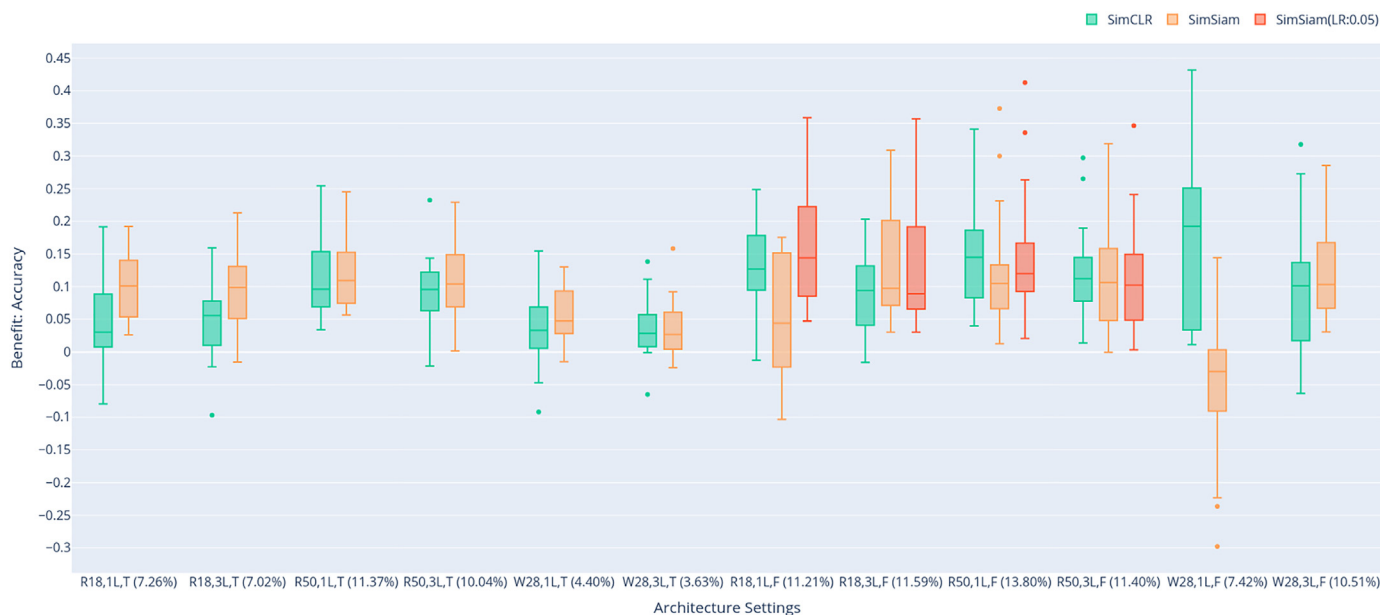


Fig. C.4. Accuracy benefit distribution over different network configurations of SimCLR and SimSiam omitting the additional MLP. The mean accuracy gain is noted in the label of each setting. For brevity, the labels are shortened (R18: ResNet18, R50: ResNet50, W28w2: Wide ResNet28w2, 1L: logistic regression, 3L: 3-layer MLP, T: encoder weights finetuned, F: encoder weights frozen).

In the case of the finetuned encoder runs, we observed that the performance is relatively constant when dropping the MLP. The additional parameters were, therefore, not very profitable in this scenario, which corresponds more to the realistic use case. Even if the MLP learned helpful information in the pretraining. Even though the MLP had contained helpful representations learned in pretraining, this could be negated solely based on the encoder CNN feature and a simple prediction head.

However, the frozen encoder settings were more interesting in this context. We found that dropping the MLP significantly influenced the performance of SimSiam here. In all settings with logistic regression, this leads to an apparent reduction of corrupted runs. Indeed, omitting the encoder MLP when using the ResNet50 architecture fixed the problem completely. In contrast, using a 3-layer prediction head, no significant performance change was observable. We suppose that during pretraining, the MLP learned a poor mapping, which a single linear transformation could not correct in the finetune process, which resulted in corrupted runs. Additional experimental runs without the added MLP were executed to investigate this further. We repeated all ResNet18 and ResNet50 architecture experiments with one exception to the finetune protocol, which was an increased learning rate. This modification, furthermore, evoked a significant performance gain in the case of the ResNet18 but only barely changed the results of the ResNet50, supporting the idea of harmful mapping. Overall, these are strong indications that this problem was induced by the design of the pretraining and finetuning protocol, not the method itself. Thus, we recommend using SimSiam as initially proposed, i.e., using only the CNN structure as a feature extractor.

<sup>7</sup> However, all presented diagrams and the corresponding f1-score charts can be generated from the raw experimental data using the reproducibility script found in the git repository.



## References

- Russakovsky O, Deng J, Su H, et al. ImageNet large scale visual recognition challenge. *Int J Comput Vision (IJCV)* 2015;115:211–252. <https://doi.org/10.1007/s11263-015-0816-y>.
- Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. *Adv Neural Inform Process Syst* 2012;25.
- Azizi S, Mustafa B, Ryan F, et al. Big self-supervised models advance medical image classification. *Proceedings of the IEEE/CVF International Conference on Computer Vision*; 2021. p. 3478–3488.
- Mahmoud ASSRAN, et al. Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples. *Proceedings of the IEEE/CVF International Conference on Computer Vision*. In. 2021. p. 8443–8452.
- Chen T, Kornblith S, Norouzi M, Hinton G. A simple framework for contrastive learning of visual representations. *International Conference on Machine Learning*. PMLR; 2020. p. 1597–1607.
- Chen X, He K. Exploring simple siamese representation learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*; 2021. p. 15750–15758.
- Veeling BS, Linmans J, Winkens J, Cohen T, Welling M. Rotation equivariant cnns for digital pathology. *International Conference on Medical image computing and computer-assisted intervention*. Springer; 2018. p. 210–218.
- Kather JN, Halama N, Marx A. *100,000 histological images of human colorectal cancer and healthy tissue*. 2018. <https://doi.org/10.5281/zenodo.1214456>.
- Graham S, Jahanifar M, Azam A, et al. Lizard: a large-scale dataset for colonic nuclear instance segmentation and classification. *Proceedings of the IEEE/CVF International Conference on Computer Vision*; 2021. p. 684–693.
- McAlpine ED, Michelow P, Celik T. The utility of unsupervised machine learning in anatomic pathology. *Am J Clin Pathol* 2021;157:5–14. <https://doi.org/10.1093/ajcp/aqab085>.
- Raina R, Battle A, Lee H, Packer B, Ng AY. Self-taught learning: transfer learning from unlabeled data. *Proceedings of the 24th International Conference on Machine Learning*; 2007. p. 759–766.
- Van den Oord A, Li Y, Vinyals O. Representation learning with contrastive predictive coding. *arXiv e-prints* 2018.arXiv:1807.
- Chen X, Fan H, Girshick RB, He K. Improved baselines with momentum contrastive learning. *ArXiv* 2020.abs/2003.04297.
- Saillard C, Dehaene O, Marchand T, et al. Self supervised learning improves dmmr/msi detection from histology slides across multiple cancers. *arXiv preprint* 2021. arXiv: 2109.05819.
- Kather JN. Histological images for MSI vs. MSS classification in gastrointestinal cancer. *FFPE Samples* 2019. <https://doi.org/10.5281/zenodo.2530835>.
- Baykaner K, Xu M, Bordeaux L, et al. Image model embeddings for digital pathology and drug development via self-supervised learning. *bioRxiv* 2021.
- McInnes Leland, John Healy, Saul Nathaniel, Lukas Großberger. UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software* 2018;3(29):861. <https://doi.org/10.21105/joss.00861>.
- Lu Ming Y, Chen Richard J, Mahmood Faisal. Semi-supervised breast cancer histology classification using deep multiple instance learning and contrast predictive coding (Conference Presentation). In: *Tomaszewski John E, Ward Aaron D, eds. Proc. SPIE* 11320, *Medical Imaging 2020: Digital Pathology*; 2020, 11320. <https://doi.org/10.1117/12.2549627>.
- Aresta G, Araújo T, Kwok S, et al. Bach: grand challenge on breast cancer histology images. *Med Image Anal* 2019;56:122–139.
- Stacke K, Lundström C, Unger J, Eilertsen G. Evaluation of contrastive predictive coding for histopathology applications. *Machine Learning for Health*. PMLR; 2020. p. 328–340.
- Chen Richard J, Lu Ming Y, Wang Jingwen, Williamson Drew FK, Rodig Scott J, Lindeman Neal I, Mahmood Faisal. Pathomic fusion: an integrated framework for fusing histopathology and genomic features for cancer diagnosis and prognosis. *IEEE Transactions on Medical Imaging*, *IEEE* 2020;41(4):757–770.
- Dehaene O, Camara A, Moindrot O, de Lavergne A, Courtiol P. Self-supervision closes the gap between weak and strong supervision in histology. *arXiv preprint* 2020.arXiv: 2012.03583.
- Koohbanani NA, Unnikrishnan B, Khurram SA, Krishnaswamy P, Rajpoot N. Self-path: self-supervision for classification of pathology images with limited annotations. *IEEE Trans Med Imaging* 2021;40:2845–2856.
- Srinidhi CL, Kim SW, Chen F-D, Martel AL. Self-supervised driven consistency training for annotation efficient histopathology image analysis. *Med Image Anal* 2022;75, 102256.
- Grill J-B, Strub F, Altché F, et al. Bootstrap your own latent: a new approach to self-supervised learning. *Adv Neural Inform Process Syst* 2020;33:21271–21284.
- Bromley J, Guyon I, LeCun Y, Säckinger E, Shah R. Signature verification using a "siamese" time delay neural network. *Adv Neural Inform Process Syst* 1993;6.
- Li T, Feng M, Wang Y, Xu K. Whole slide images based cervical cancer classification using self-supervised learning and multiple instance learning. *2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*; 2021. p. 192–195. <https://doi.org/10.1109/ICBAIE52039.2021.9389824>.
- Yang P, Hong Z, Yin X, Zhu C, Jiang R. Self-supervised visual representation learning for histopathological images. *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer; 2021. p. 47–57.
- Zhang J, Hua Z, Yan K, et al. Joint fully convolutional and graph convolutional networks for weakly-supervised segmentation of pathology images. *Med Image Anal* 2021;73, 102183.
- Ciga O, Xu T, Martel AL. Self supervised contrastive learning for digital histopathology. *Mach Learn Appl* 2022;7, 100198. URL: <https://www.sciencedirect.com/science/article/pii/S2666827021000992>, <https://doi.org/10.1016/j.mlwa.2021.100198>.
- Wang H, Zheng H, Chen J, Yang L, Zhang Y, Chen DZ. Unlabeled data guided semi-supervised histopathology image segmentation. *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. *IEEE*; 2020. p. 815–820.
- Wei J, Suriawinata A, Vaickus L, et al. Generative image translation for data augmentation in colorectal histopathology images. *Proc Mach Learn Res* 2019;116:10.
- Wei J, Suriawinata A, Vaickus L, et al. Generative image translation for data augmentation in colorectal histopathology images. In: *Dalca AV, McDermott MB, Alsentzer E, et al, eds. Proceedings of the Machine Learning for Health NeurIPS Workshop*. Volume 116 of *Proceedings of Machine Learning Research*. PMLR; 2020. p. 10–24. <https://proceedings.mlr.press/v116/wei20a.html>.
- de Vulpian A, di Proietto V, Roy G, Hadji SB, Fick RR. A semi-supervised deep learning approach for multi-stain foreground segmentation in digital pathology. *Medical Imaging with Deep Learning*; 2022. <https://openreview.net/forum?id=6uw53DAsjNG>.
- Liu Kun, Liu Zhuolin, Liu Sidong. Semi-supervised breast histopathological image classification with self-training based on non-linear distance metric. *IET Image Processing* 2022;16(12):3164–3176.
- Sikaroudi M, Safarpoor A, Ghojogh B, Shafiei S, Crowley M, Tizhoosh HR. Supervision and source domain impact on representation learning: A histopathology case study. *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. *IEEE*; 2020. p. 1400–1403.
- Bejnordi BE, Veta M, Van Diest PJ, et al. Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. *Jama* 2017;318:2199–2210.
- He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*; 2016. p. 770–778.
- Li L, Jamieson K, Rostamizadeh A, et al. A system for massively parallel hyperparameter tuning. *Proc Mach Learn Syst* 2020;2:230–246.
- van der Maaten L, Hinton GE. Visualizing data using t-sne. *J Mach Learn Res* 2008;9: 2579–2605.
- Tellez D, Litjens G, Bándi P, et al. Quantifying the effects of data augmentation and stain color normalization in convolutional neural networks for computational pathology. *Med Image Anal* 2019;58, 101544.
- Chen T, Kornblith S, Swersky K, Norouzi M, Hinton GE. Big self-supervised models are strong semi-supervised learners. *Adv Neural Inform Process Syst* 2020;33:22243–22255.
- Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-cam: visual explanations from deep networks via gradient-based localization. *Proceedings of the IEEE International Conference on Computer Vision*; 2017. p. 618–626.
- Springenberg Jost Tobias, Dosovitskiy Alexey, Brox Thomas, Riedmiller Martin A. Striving for Simplicity: The All Convolutional Net. *CoRR* 2014.abs/1412.6806.
- Kather JN, Krisam J, Charoentong P, et al. Predicting survival from colorectal cancer histology slides using deep learning: a retrospective multicenter study. *PLoS Med* 2019;16, e1002730.
- Annuschein J, Voigt B, Fischer O, et al. Systematic investigation of basic data augmentation strategies on histopathology images. *Artificial Intelligence – Application in Life Sciences and Beyond. The Upper Rhine Artificial Intelligence Symposium UR-AI 2021*, arXiv; 2021. p. 39–48. <https://doi.org/10.108550/ARXIV.2112.05657>.
- Kiehl T-R. Private Communication 2022.

## **Curriculum Vitae**

Mein Lebenslauf wird aus datenschutzrechtlichen Gründen in der elektronischen Version meiner Arbeit nicht veröffentlicht.



## Publication list

1. **Voigt B**, Fischer O, Schilling B, Krumnow C, Herta C. Investigation of semi- and self-supervised learning methods in the histopathological domain. *Journal of Pathology Informatics*. 2023;100305.
2. **Voigt B**, Fischer O, Krumnow C, Herta C, Dabrowski PW. NGS read classification using AI. *PLOS ONE*. 2021 Dec;16(12):1–13.
3. Annuschein J, **Voigt B**, Fischer O, Baumann P, Lohmann S, Krumnow C, et al. Systematic investigation of Basic Data Augmentation Strategies on Histopathology Images. In: *Artificial Intelligence – Application in Life Sciences and Beyond The Upper Rhine Artificial Intelligence Symposium UR-AI 2021*. arXiv; 2021. p. 39–48.
4. Strohmenger K, Annuschein J, Klempert I, **Voigt B**, Herta C, Hufnagl P. Convolutional neural networks and random forests for detection and classification of metastasis in histological slides. Submission results Camelyon17 challenge <https://camelyon17.grand-challenge.org/evaluation/results/> Accessed. 2019;18.
5. Herta C, **Voigt B**, Baumann PW, Strohmenger K, Jansen C, Fischer O, et al. Deep Teaching: Materials for Teaching Machine and Deep Learning. 5th International Conference on Higher Education Advances (HEAd'19). 2019;
6. Herta C, **Voigt B**. Radial Prediction Layer. ArXiv. 2019;abs/1905.11150.
7. Göppert T, **Voigt B**, Stege A, Bettig U, Hufnagl P. Der wissenschaftliche Wert von Bioproben - eine qualitative Expert\_innenbefragung. In: *Nationales Biobanken-Symposium – Jahresbericht Aktuelle Herausforderungen und Chancen im Biobanking 6 Nationales Biobanken-Symposium 2017 – Tagungsband*. Berlin: Akademische Verlagsgesellschaft AKA GmbH; 2017. p. 161–5.
8. Bruun A, Hahn C, **Voigt B**, Schultz M. Touch, and You Will Gaze: Elderly and Youngers' Use of Remote Controls in Interacting with a Healthcare Portal. *Studies in health technology and informatics*. 2013;190:135–7.
9. Bruun A, Hahn C, **Voigt B**, Schultz M. Digging Wide and Narrow: An Exploratory Study of Senior and Younger Users' Strategies for Retrieving Information from a Healthcare Portal. In 2012.
10. **Voigt B**, Cornils M, Pilgermann S, Schultz M. Entwurf und Implementierung einer standardbasierten Telemedizinplattform am Beispiel eines Szenarios im Rahmen des SmartSenior-Projektes. In: *Deutscher AAL-Kongress: Demographischer Wandel-Assistenzsysteme aus der Forschung in den Markt*, Berlin, Deutschland. 2011.
11. Ehret M, Fritsch T, **Voigt B**. The Profi League Continues - An Online Gaming Community for Professional Players. In: *GAMEON*. 2009.
12. Fritsch T, **Voigt B**, Schiller JH. Next Generation In-game Message Interfaces. In 2007.
13. Fritsch T, Schiller JH, **Voigt B**. Personal behavior and virtual fragmentation. In: *ACE '07*. 2007.
14. Fritsch T, **Voigt B**, Schiller JH. Distribution of online hardcore player behavior: (how hardcore are you?). In: *Network and System Support for Games*. 2006.

## Acknowledgments

Thank you Christian for sharing your knowledge and expertise.

Thank you, Chris, your leadership has brought this work to a successful conclusion.

Thank you, Peter, for your support before and during the PhD, your faith in me and your friendship.

Finally, special thanks to you Sanne! You shared all the moments with me during this time. Your backing and support gave me the strength to finish this personal project. I am unable to express the gratitude I feel for this.