





## Article

# A Systematic Comparison of Task Adaptation Techniques for Digital Histopathology

Daniel Sauter <sup>1,\*</sup>, Georg Lodde <sup>2</sup>, Felix Nensa <sup>3,4</sup>, Dirk Schadendorf <sup>2</sup>, Elisabeth Livingstone <sup>2</sup> and Markus Kukuk <sup>1</sup>

<sup>1</sup> Department of Computer Science, Fachhochschule Dortmund, 44227 Dortmund, Germany; markus.kukuk@fh-dortmund.de

<sup>2</sup> Department of Dermatology, University Hospital Essen, 45147 Essen, Germany; georg.lodde@uk-essen.de (G.L.); dirk.schadendorf@uk-essen.de (D.S.); elisabeth.livingstone@uk-essen.de (E.L.)

<sup>3</sup> Institute for AI in Medicine (IKIM), University Hospital Essen, 45131 Essen, Germany; felix.nensa@uk-essen.de

<sup>4</sup> Institute of Diagnostic and Interventional Radiology and Neuroradiology, University Hospital Essen, 45147 Essen, Germany

\* Correspondence: daniel.sauter@fh-dortmund.de

**Abstract:** Due to an insufficient amount of image annotation, artificial intelligence in computational histopathology usually relies on fine-tuning pre-trained neural networks. While vanilla fine-tuning has shown to be effective, research on computer vision has recently proposed improved algorithms, promising better accuracy. While initial studies have demonstrated the benefits of these algorithms for medical AI, in particular for radiology, there is no empirical evidence for improved accuracy in histopathology. Therefore, based on the ConvNeXt architecture, our study performs a systematic comparison of nine task adaptation techniques, namely, DELTA, L<sup>2</sup>-SP, MARS-PGM, Bi-Tuning, BSS, MultiTune, SpotTune, Co-Tuning, and vanilla fine-tuning, on five histopathological classification tasks using eight datasets. The results are based on external testing and statistical validation and reveal a multifaceted picture: some techniques are better suited for histopathology than others, but depending on the classification task, a significant relative improvement in accuracy was observed for five advanced task adaptation techniques over the control method, i.e., vanilla fine-tuning (e.g., Co-Tuning:  $P(\gg) = 0.942$ ,  $d = 2.623$ ). Furthermore, we studied the classification accuracy for three of the nine methods with respect to the training set size (e.g., Co-Tuning:  $P(\gg) = 0.951$ ,  $\gamma = 0.748$ ). Overall, our results show that the performance of advanced task adaptation techniques in histopathology is affected by influencing factors such as the specific classification task or the size of the training dataset.

**Keywords:** transfer learning; fine-tuning; computer vision; CNN; whole-slide imaging; cancer



**Citation:** Sauter, D.; Lodde, G.; Nensa, F.; Schadendorf, D.; Livingstone, E.; Kukuk, M. A Systematic Comparison of Task Adaptation Techniques for Digital Histopathology. *Bioengineering* **2024**, *11*, 19. <https://doi.org/10.3390/bioengineering11010019>

Academic Editors: Andrea Cataldo and Giuseppe Baselli

Received: 20 November 2023

Revised: 20 December 2023

Accepted: 21 December 2023

Published: 24 December 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

The need for improved tools in clinical histopathology has been repeatedly emphasized. Commonly cited arguments include an increasing need for accuracy, a lack of specialized histopathologists, subjectivity, and a lack of reproducibility [1,2]. Furthermore, the situation is expected to worsen due to increasing cancer incidence and more screening campaigns [3]. Fortunately, deep neural networks are expected to significantly change the study of cancer tissue in histopathology [1]. In fact, deep learning (DL) in computational histopathology promises various benefits. These include time and cost savings, lower error rates, better accessibility, and the learning of more accurate representations [2]. Furthermore, it can improve clinical decision making by providing access to high-level information [4]. It also plays an important role in mining big data [5]. As a result, the yearly number of research papers on DL-based histopathology is steadily increasing [3,6].

However, DL models must achieve a sufficient level of accuracy to be approved for clinical practice, while digital histopathology poses several challenges for the application of DL, including large image size, insufficient data annotation, varying magnification levels of the microscope, staining artifacts, and color variation [2]. Transfer learning, including the fine-tuning of pre-trained models, is a standard technical solution for insufficient annotation. Given limited training data, fine-tuning can significantly improve the performance compared to training from scratch [7]. Kornblith et al. [8] also found that, on average, fine-tuning achieved a 17-fold increase in the training speed compared to training from scratch. This was confirmed by He et al. [9]. In histopathology, fine-tuning appears to be generally superior to using off-the-shelf features [10,11].

The exact procedure for fine-tuning, however, is still an active field of research. Early studies attempted to systematically evaluate the relationship between model parameters and transferability in convolutional neural networks (CNNs) [12]. Recently, several studies have tried to systematically investigate the impact of pre-training [13,14], algorithms [15,16], and parameters [8,17]. Raghu et al. [18] reported findings on model size in the medical domain. A recent literature review for computer vision (CV) with further information can be found in the dissertation by Plested [19], among others.

Especially the technical approach to transfer learning in histopathology leaves room for improvement. The most straightforward algorithm consists of copying the pre-trained weights of the initial layers, reinitializing the subsequent layers, and then training them together. This algorithm dates back to the early 2010s [7,20,21]. At present, researchers using pre-trained DL models for digital histopathology can choose from various recently proposed algorithms from CV research [22–24]. Some initially successful applications of advanced transfer learning techniques in the medical domain can be found with respect to radiological images [25–28] or lung sound analysis [29]. However, the wider adoption of these techniques in medicine has not occurred to date. For example, “task adaptation,” which transfers a pre-trained model into a supervised target domain [30], has not been widely explored for histopathology. In general, Sauter et al. [31] recently noticed a lack of transfer of state-of-the-art DL methodology into the histopathology of malignant melanoma.

On the other hand, choosing a transfer learning algorithm for histopathology can be challenging. It is known that no machine learning (ML) algorithm is generally superior in every conceivable use case [32]. More specifically, Zhang et al. [33] noted that, in addition to other factors, the benefit of transfer learning depends on the algorithm due to assumptions made and specific application scenarios. Accordingly, the question arises of whether transfer learning algorithms that perform well on datasets like ImageNet are also suitable for histopathology. In addition, there is the related question of how much each algorithm benefits from more labeled training data. From a practical point of view, gaining only a small increase in accuracy would perhaps speak against the labor- and cost-intensive effort of labeling additional training data by pathology experts [2]. Systematic evaluations of other technical aspects, such as pre-training [34,35] and data augmentation [36], exist. However, there is—to our knowledge—no systematic comparison of task adaptation strategies applied to histopathological data.

The present study intends to advance the development of DL models for histopathology via improved transfer learning techniques from CV. Based on our findings, follow-up research can enhance their models addressing specific clinical questions and achieve accuracy levels sufficient for clinical approval. This will hopefully help to realize the potential benefits of DL in clinical patient care. Therefore, in our study, we would like to know which task adaptation technique(s) should be used when developing DL models for medical use cases to achieve the best possible accuracy. Consequently, our research question is as follows: which task adaptation technique achieves the highest accuracy (in terms of AUC) for classification in digital histopathology tasks?

Our study has several unique contributions, which can be summarized as follows:

- It is the first systematic evaluation of up-to-date task adaptation techniques for classification tasks in digital histopathology.

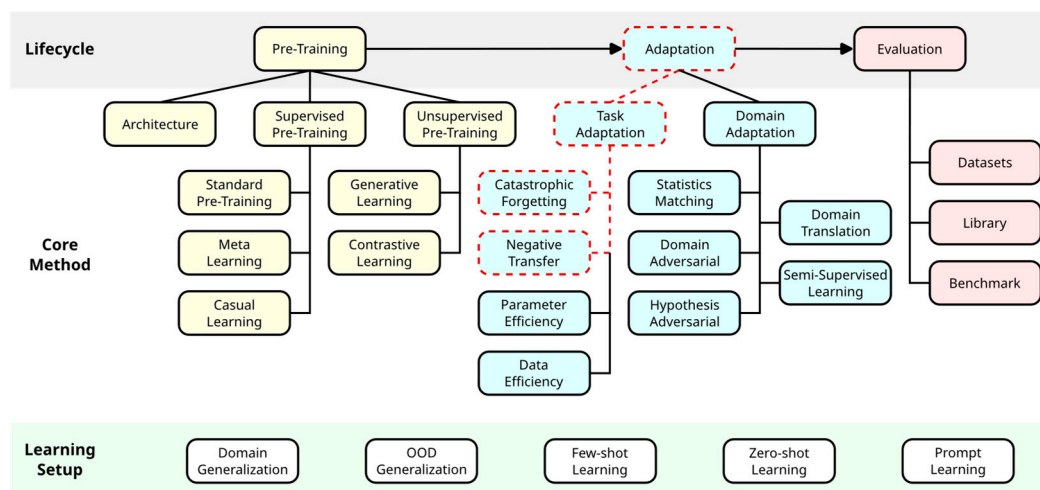
- We comprehensively carried out evaluations based on five histopathological classification tasks (mitotic figure detection, tumor metastasis detection, tumor-infiltrating lymphocytes detection, colorectal cancer tissue-type classification, and skin cancer tissue-type classification) and eight datasets. Our evaluations include external testing and statistical validation.
- We show that the standard fine-tuning procedure can be outperformed by more advanced task adaptation techniques depending on the task at hand.
- Furthermore, the impact of dataset size is investigated. We show that the Co-Tuning technique can offer further improvements in large-scale settings.

## 2. Research Background

The following section that presents the research background is divided into two parts. First, the section introduces some categorization schemas of transfer learning. Second, techniques from the literature on CV are described and categorized from a technical perspective.

### 2.1. Transfer Learning

Transfer learning is a technique in the field of ML that has been studied for some time. Its goal is to improve ML models by transferring knowledge from a source domain to the target domain [37]. Jiang et al. [30] recently structured research on transfer learning for DL from a lifecycle-oriented perspective (see Figure 1). The main training steps are pre-training a model in the source domain to learn transferable knowledge and then adapting it in the target domain. In the adaptation step, a further distinction is made between whether labels are available in the target domain (“task adaptation”) or not (“domain adaptation”) [30]. Task adaptation is further divided into four subcategories. “Catastrophic forgetting” refers to the loss of information that was learned from previous tasks when subsequently learning a new task [38,39]. “Negative transfer” describes a decrease in performance in the target domain caused by preceding learning processes in the source domain [33,40]. “Parameter efficiency” reduces the amount of computation in the target domain [41,42]. “Data efficiency” minimizes the amount of data required for adaptation in the target domain [30].

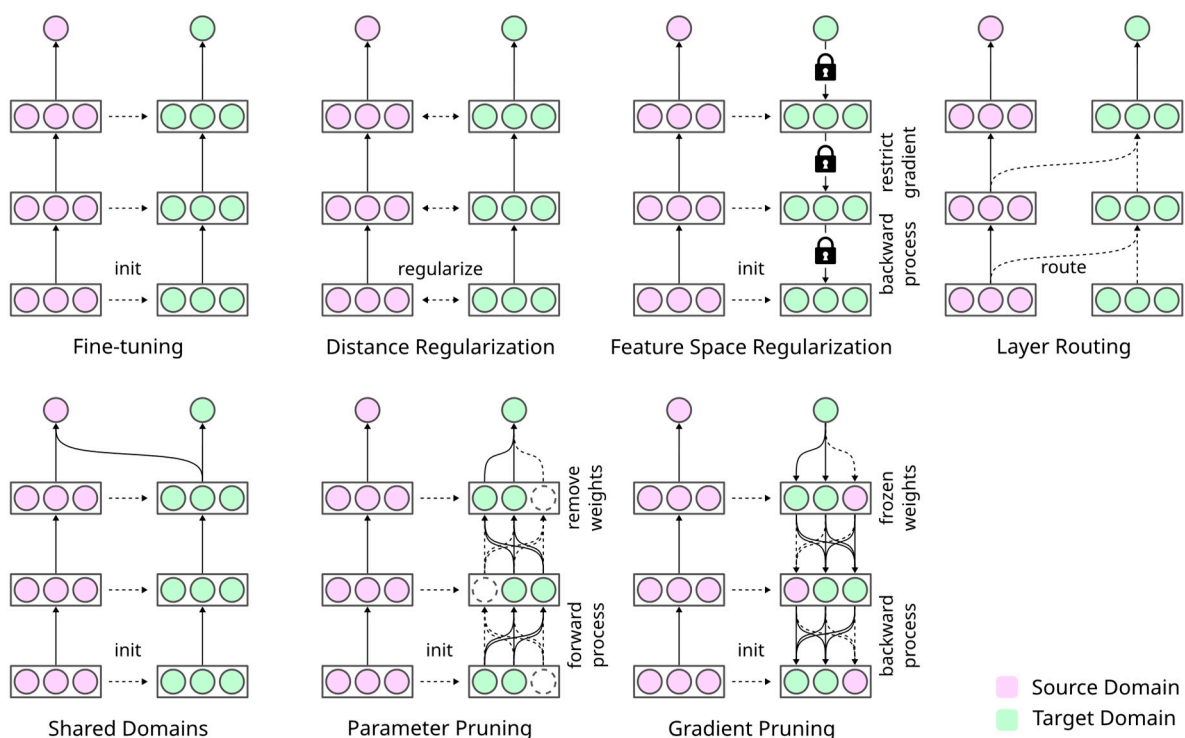


**Figure 1.** Lifecycle-oriented perspective on transfer learning. Transfer learning includes pre-training on an upstream task in the source domain and adaptation to a downstream task in the target domain. The adaptation process can be divided based on whether labels are available in the target domain (“task adaptation”) or not (“domain adaptation”). Red dashed lines highlight the subcategories examined in our study. Adapted with permission from Jiang et al. [30].

There are several related fields of research. “Domain generalization” tries to learn a predictor in a set of source domains so that it is invariant relative to distribution shifts. The predictor thus generalizes well in an unseen target domain [43,44]. “Few-shot learning” aims to learn ML models using only a small amount of training data for supervised training [45]. Research on “zero-shot learning” tries to train a model without supervised labeling in the target domain [46]. Recently, “prompt learning” from the field of natural language processing has been employed to attempt to reduce or avoid labeled data in the target domain. For this, it embeds the input into a prompting function and searches for the highest-scoring solution [47]. Some approaches transfer knowledge from a set of multiple neural networks in the source domain (“model zoo”) [48]. In “model selection,” different models are compared with respect to their suitability for a specific transfer learning problem [49]. “Continual learning” is a technique used for training neural networks with respect to multiple tasks in succession without unlearning previously learned tasks [50].

## 2.2. Task Adaptation Techniques

From another perspective, task adaptation strategies can be broadly categorized into seven categories of technical approaches (see Figure 2) [51]. In the following, we describe task adaptation techniques from the literature relative to image classification (see Table 1). In each case, we classify the techniques in a technical scheme.



**Figure 2.** Technical perspective on different forms of task adaptation. Adapted with permission from Ding et al. [51] and extended by introducing the categories “feature space regularization” and “shared domains”.

### 2.2.1. Vanilla Fine-Tuning

Regular “vanilla” fine-tuning is relatively simple and usually includes three steps [7,20,21]. First, one trains a neural network on a source task. Second, the last layers of the network are usually replaced by newly initialized layers. Third, single layers are either frozen or adapted to a target task using additional supervised training.

### 2.2.2. Distance Regularization

“Distance regularization” attempts to minimize the distance between fine-tuned weights and pre-trained weights [51]. It thereby preserves knowledge from the source domain. Most techniques rely on the loss function to regulate the feature space. “Explicit Inductive Bias” (L<sup>2</sup>-SP) [24] is based on the standard L<sup>2</sup> regularization called “weight decay”. New layers that are added to the network and that are then reinitialized use this regularization. The procedure adds a second loss term, which penalizes large distances between the fine-tuned and pre-trained weights. The strength of both loss components is controlled using two parameters:  $\alpha$  and  $\beta$ . “MARS-SP” [52] penalizes the maximum absolute row sum (MARS) distance between the weights of the source and the target domain. “DEep Learning Transfer using Feature Map with Attention” (DELTA) [53] is similar to L<sup>2</sup>-SP. However, it uses the distance between the feature maps of the source and the target model (instead of the layer weights). The distance is weighted by the performance reduction caused by turning off filters one at a time.

Instead of limiting the deviations using the loss function, hard constraints can be used. Gouk et al. [52] studied two variants. “L<sup>2</sup>-PGM” [52] uses a variant of the projected stochastic sub-gradient method with a constraint on the Frobenius distance. “MARS-PGM” [52] uses a constraint on the MARS distance instead.

**Table 1.** Overview of task adaptation techniques from the literature on image classification and categorized from a technical perspective.

Technique	Reference	Category	Ranking	Source Code
<b>Distance Regularization</b>				
DELTA	[53]	CF	A*	✓
L <sup>2</sup> -PGM	[52]	CF	A*	✓
L <sup>2</sup> -SP	[24]	CF	A*	✓
MARS-PGM	[52]	CF	A*	✓
MARS-SP	[52]	CF	A*	✓
<b>Feature Space Regularization</b>				
Bi-Tuning	[54]	DE	A	✓
BSS	[55]	NT	A*	✓
DTNH	[15]	NT	A*	✗
StochNorm	[56]	DE	A*	✓
<b>Layer Routing</b>				
AdaFilter	[57]	CF	A*	✗
DEFT	[58]	CF	Q1	✗
DKL	[59]	CF	Q2	✓
Flex-Tuning	[60]	CF	A	✗
MultiTune	[61]	CF	B	✓
PTU	[62]	CF	A	✗
SpotTune	[23]	CF	A*	✓
PathNet	[63]	CF	B	✗
Stepwise PathNet	[64]	CF	Q1	✗
<b>Shared Domains</b>				
Co-Tuning	[22]	CF	A*	✓
LwF	[65]	CF	Q1	✓
Selective Joint Fine-tuning	[66]	DE	A*	✓
<b>Parameter Pruning</b>				
Ticket Transfer	[67]	PE	A*	✓
Winning Lottery Tickets	[68]	PE	B	✓

Category descriptors include: catastrophic forgetting (CF), negative transfer (NT), parameter efficiency (PE), or data efficiency (DE). Ranking according to SCImago Journal Rank (SJR) indicator (Q1, Q2, Q3, or Q4) or CORE Conference Ranking score (A\*, A, B, or C).

### 2.2.3. Feature Space Regularization

The “feature space regularization” techniques apply some restrictions on backpropagation to ensure the desired properties of the fine-tuned feature space. Two techniques belong



to the category of catastrophic forgetting. “Stochastic Normalization” (StochNorm) [56] replaces standard batch normalization with a two-branch version. One branch uses means and mini-batch variances; the other uses moving statistics. The branches are selected stochastically. In this manner, over-fitting is penalized more, and more knowledge is also transferred. “Bi-Tuning” [54] adds a newly designed categorical contrastive learning loss to the loss term to better make use of the intrinsic structure of pre-trained feature representations.

Two other methods try to reduce negative transfer. “Batch Spectral Shrinkage” (BSS) [55] applies singular value decomposition (SVD) on feature matrices. It assumes that spectral components with small singular values are not transferable. In order to suppress negative transfer, BSS penalizes the smallest singular values by using an additional loss term. Wan et al. [15] developed “Descent Direction Estimation Strategy” (DTNH), which evaluates the gradients of empirical loss and regularization separately. In the case of an obtuse angle between those two, it decomposes the regularization gradient into two orthogonal vectors. One of these two vectors, which is parallel to the empirical gradient, is then truncated. This prevents the empirical loss descent from slowing down.

#### 2.2.4. Layer Routing

For “layer routing,” some of the pre-trained layers are either frozen or disconnected [51]. The selection of layer subsets is carried out using different approaches. Routing can be performed using modifications of the model architecture itself. “MultiTune” [61] applies L<sup>2</sup>-SP on two parallel ResNet models with different parameters each. Then, a single fully connected layer called “MultiTune layer” is trained to combine the output of these two models. “Parameter Transfer Unit” (PTU) [62] uses two parallel neural networks. One is trained in the source domain and then frozen. The other is trained from scratch in the target domain. Between the layers, two neurons called “fine-tune gate” and “update gate” learn to combine the activations of both models.

Some methods identify layers for fine-tuning using simple algorithmic approaches: “Flex-tuning” [60] decomposes a neural network into subcomponents. It identifies one component that should be fine-tuned (while the others are frozen). The fastest variant, “even faster flex-tuning,” carries out a single fine-tuning epoch. Then, each fine-tuned block is copied into a separate network with pre-trained weights to obtain a proxy measurement of accuracy. After selecting the best block, it is fully fine-tuned. “DKL” [59] uses the Kullback–Leibler divergence on weight correlations to identify the best layers for fine-tuning.

Some techniques use an additional policy network for routing prediction: “Spot-Tune” [23] is based on the view that ResNet is an ensemble of shallow networks [69]. The approach uses two copies of the pre-trained model: a frozen and a trainable one. For each image, a policy network decides whether a pre-trained or fine-tuned block of ResNet should be used. As the binary policy vector itself is discrete and thus non-differentiable, the Gumbel–SoftMax trick is used during the backward pass. “AdaFilter” [57] uses pre-trained and fine-tuned filters. A binary vector called “fine-tuning policy” decides which filters to use on a per-image basis. These fine-tuning policies are predicted using a recurrent gated network. The network is trained using the “straight-through estimator” because of the binary values during the forward pass. Gated batch normalization replaces standard batch normalization to handle domain shifts.

Some approaches make use of evolutionary algorithms for routing: “PathNet” [63] finds binary fine-tuning genotypes (fine-tune/do not fine-tune specific layers) using a genetic algorithm. The best genotypes are combined into an ensemble. In “Stepwise PathNet” [64], a tournament selection algorithm based on a microbiological genetic algorithm carries out selections between frozen pre-trained and fine-tunable pre-trained layers. Similarly to PathNet, “Differential Evolution based Fine-Tuning” (DEFT) [58] uses an evolutionary algorithm (differential evolution) to select layers for fine-tuning. However, real-valued genotypes (instead of binary values) are used. For training, phenotypes

are binarized using a simple threshold. In the end, a final model is fine-tuned based on the results.

### 2.2.5. Shared Domains

The category “shared domains” uses information from the source domain for additional training supervision during the backpropagation step. “Selective Joint Fine-tuning” [66] searches for training images with visually similar low-level characteristics. Two output layers and two cost functions are used to simultaneously train the network on the source and target datasets. “Learning without Forgetting” (LwF) [65] uses the pre-trained classifier to predict labels for new images in the source label space. Then, new output layers are initially trained (warm-up step), while previous layers are kept frozen. Finally, with all layers unfrozen, the model is jointly trained to predict the pre-computed source labels and the new layers using a combined loss term. “Co-Tuning” [22] translates target labels into probability distributions in the source domain. During training, the translated probability distributions combined with the source classifier layers are used to add an additional element to the loss function.

### 2.2.6. Parameter Pruning

“Parameter pruning” leverages the structural information of the pre-trained model from the source domain [51]. van Soelen and Sheppard [68] carried out training on a source dataset. Based on the “Winning Lottery Tickets hypothesis” [70], they identified winning tickets using one-shot pruning. The authors then retrained them on the target dataset. Each layer was pruned individually, and bias terms were excluded. Mehta [67] also adopted the procedure by Frankle and Carbin [70] and validated what they called the “Ticket Transfer Hypothesis”. However, they applied iterative pruning. The weights of the winning ticket were then transferred to the target domain.

## 3. Methodology

This section describes the methodology of our study with respect to task adaptation and its various techniques. As shown in Section 2, task adaptation comprises several closely related research areas as well as a variety of algorithms. Therefore, in this section, we begin with a precise delimitation of our object of investigation based on various criteria. Secondly, we describe an appropriate experimental design to answer our research question for the previously delimited algorithms. Thirdly, our research question itself does not imply a specific DL pipeline architecture. To collect empirical results, however, we must restrict ourselves to a single representative architecture. We therefore define and describe the exemplary DL CNN pipeline used in this study. Finally, we lay out the details regarding its hardware and the software implementation.

### 3.1. Transfer Learning Techniques

Our study defined a specific application scenario regarding the availability of labeled data: we assumed that there was a small amount of labeled data available. While the amount was enough for supervised model training, it was not sufficient for “training from scratch”. Therefore, we subsequently disregarded the application areas of data efficiency, domain adaptation, and the related areas of few-shot/zero-shot learning. Furthermore, we were interested in training a model with high performance. Thus, we were not interested in runtime performance and disregarded parameter efficiency as well. Our focus was, therefore, on task adaptation, especially catastrophic forgetting and negative transfer. We further narrowed down the use case scenario. We assumed that there was only one model in the source domain (no model zoo) and that the time-consuming pre-training was already carried out (no model selection). The focus was on the target domain; thus, accuracy in the source domain was irrelevant (no continual learning).

The task adaptation techniques to be included in our study were selected according to several criteria. One constraint was to keep the total computation time of our experimental

setup feasible with respect to our available computational resources. Model training in DL takes significant time and resources. More specifically, we had to train  $m \times n \times k$  models, where  $m$  is the number of techniques to be evaluated,  $n$  is the number of tasks, and  $k$  is the number of training repetitions [71]. Another motivation was to restrict the findings to procedures that are relevant to practical applications.

For the reasons mentioned above, we defined the following theoretical and practical inclusion criteria:

- We required the code availability of a PyTorch/TensorFlow/Keras implementation.
- The task adaptation technique was designed for/was suitable to image classification models.
- The task adaptation technique was compatible with the ConvNeXt architecture.
- The task adaptation technique was designed for high accuracy/AUC (catastrophic forgetting and negative transfer).
- The original publication was published in a peer-reviewed journal/conference and holds a good ranking in the SJR (Q1) or CORE ranking (A\*, A, B).

Furthermore, the following restrictions were used to reduce the computational complexity of our evaluation:

- We dropped L2-PGM and MARS-SP, as MARS-PGM seems to be favored by Gouk et al. [52].
- We dropped LwF in favor of Co-Tuning as the approaches are similar, and You et al. [22] found that “Co-Tuning fits better for transfer learning than LwF because Co-Tuning explicitly models the category relationship” [22].

We finally included DELTA, L<sup>2</sup>-SP, and MARS-PGM from distance regularization; Bi-Tuning and BSS from feature space regularization; MultiTune and SpotTune from layer routing; and Co-Tuning from shared domains. As a baseline, we chose to train a model using the standard vanilla fine-tuning approach [7].

### 3.2. Experimental Design

In our experimental design, we limited ourselves to a subset of histopathological classification tasks and their corresponding public datasets. Then, we distinguished between two aspects: comparing task adaptation algorithms and comparing training dataset sizes, each with respect to their achieved accuracy. The respective evaluation procedures follow the latest best practices for the comparison of ML algorithms to ensure generalizable findings [71,72].

#### 3.2.1. Classification Tasks and Datasets

The study focused on histopathology in general and therefore addressed exemplary clinical tasks based on previously published datasets. To cover a wider variety of application scenarios in histopathology, we included five classification tasks from multiple cancer types. We also collected multiple public image datasets for each task. The following describes the classification tasks and the corresponding datasets in detail.

*Mitotic Figure Detection.* Our first classification task was partly adopted from Tellez et al.’s study [36]. In this case, the task was to detect mitotic figures in the center of image patches from breast cancer. The predicted variable is binary, namely, “mitotic figure present” vs. “no mitotic figure present” in the center. As our first dataset, we used the “breast cancer histopathological annotation and diagnosis dataset” (BreCaHAD) [73]. Hematoxylin and eosin (H&E)-stained images with a size of  $1360 \times 1024$  pixels were collected at the University of Calgary. Our second dataset comes from the “Tumor Proliferation Assessment Challenge” in 2016 (TUPAC16) [74]. More specifically, the “mitosis detection dataset” consists of H&E-stained square images of different sizes. It was gathered from three pathology centers in The Netherlands: the University Medical Center in Utrecht, the Symbiant Pathology Expert Center in Alkmaar, and the Symbiant Pathology Expert Center in Zaandam.



The images from the TUPAC16 and the BreCaHAD datasets had a spatial resolution of 0.25  $\mu\text{m}/\text{pixel}$ .

*Tumor Metastasis Detection.* The second classification task was partly adopted from Tellez et al.'s study [36] as well. The purpose of the classifier was to detect patches containing metastatic tumor cells of breast cancer. Therefore, the binary classes were "metastatic tumor cells present" and "others". Both the training and test data came from the "Cancer MEtastases in LYmph nOdes challeNge" in 2017 (CAMELYON17) [75]. The annotated data included H&E-stained whole-slide images (WSIs) from the Radboud University Medical Center (Radboudumc), the Utrecht University Medical Center, the Rijnstate Hospital, the Canisius-Wilhelmina Hospital, and the LabPON in The Netherlands. Model training and external testing were performed on the data from the first, second, third, and fourth centers. Image data from all centers in the CAMELYON17 dataset had a spatial resolution of 0.23–0.25  $\mu\text{m}/\text{pixel}$ .

*Tumor-infiltrating Lymphocytes (TIL) Detection.* The third task was to classify TIL-positive/-negative image patches. In this case, TIL-positive meant that at least two TILs were present in the image [76–78]. The dataset included H&E-stained square images with  $100 \times 100$  pixels from 22 cancer types in "The Cancer Genome Atlas" (TCGA) program. We excluded four cancer types (CESC, LUSC, READ, and STAD) as they were annotated using DL. This resulted in a dataset of 18 remaining TCGA projects (ACC, BRCA, COAD, ESCA, HNSC, KIRC, LIHC, LUAD, MESO, OV, PAAD, PRAD, SARC, SKCM, TGCT, THYM, UCEC, and UVM). We obtained two independent, non-overlapping mitosis datasets by randomly dividing cancer types into two splits. The spatial resolution of all image patches in TCGA-TIL was 0.50  $\mu\text{m}/\text{pixel}$ .

*Colorectal Cancer Tissue-type Classification.* Again, based on the study by Tellez et al. [36], the goal of colorectal cancer (CRC) tissue type classification was to distinguish among six different CRC tissue classes. The classes for classification were "tumor epithelium," "simple stroma," "immune cells," "debris and mucus," "normal mucosal glands," and "adipose tissue". As the first dataset, CRC-5000 by Kather et al. [79] was used. They provided 5000 patches (625 per class) of archived H&E-stained WSIs from the University Medical Center Mannheim. Each patch had a resolution of  $150 \times 150$  pixels. As the second dataset, the colon cancer dataset (DRCO) from the DROID project was used [80]. DRCO provided H&E-stained WSIs with annotation masks. To align the classes for CRC-5000 and DRCO, we dropped the "complex stroma" and "background" classes from CRC-5000. The severe imbalance of some classes made it difficult to divide the classes into training and test splits without data leakage. Thus, we excluded two "immune cells" and "debris and mucus" classes for the task "DRCO Small". The spatial resolution of CRC-5000 was 0.50  $\mu\text{m}/\text{pixel}$ , and DRCO used a resolution of 0.45–0.50  $\mu\text{m}/\text{pixel}$ .

*Skin Cancer Tissue-type Classification.* For skin cancer classification, the goal of the classifier was to distinguish among eight different skin tissue types. The tissue classes were "skin appendage" (including hair follicles and sweat glands), "inflammation," "hypodermis," "dermis," "epidermis," "basal cell carcinoma" (BCC), "squamous cell carcinoma" (SCC), and "intraepidermal carcinoma" (IEC). We used a dataset that we subsequently referred to as "Queensland" as our first dataset. The dataset was published by Thomas et al. [81] and provided by MyLab Pathology in Australia. It included segmented H&E-stained WSIs of BCC, SCC, and IEC. As our second dataset, the skin cancer dataset (DRSK) from the DROID project was used [80]. DRSK provided H&E-stained WSIs with annotation masks. To align the classes of Queensland and DRSK, we dropped the "keratin" and "background" classes with respect to Queensland. Furthermore, "reticular dermis" and "papillary dermis" were combined into "dermis". The classes "sweat glands" and "hair follicles" were combined into "skin appendage structure". The severe imbalance of some classes made dividing these classes into training and test splits difficult without causing data leakage. Thus, we dropped three classes, BCC, SCC, and IEC, for the task "DRSK Small". The spatial resolution of Queensland was 0.67  $\mu\text{m}/\text{pixel}$ , while the images in DRSK had a resolution of 0.45–0.50  $\mu\text{m}/\text{pixel}$ .

### 3.2.2. Comparison of the Task Adaptation Techniques

The experimental design for comparing task adaptation techniques was based on the statistical comparison of multiple ML algorithms across multiple datasets, as described by Benavoli et al. [71]. Generally, for the comparison of  $m$  techniques on  $n$  tasks, model training is repeated  $k$  times for each combination of  $m$  and  $n$ . The mean values of the  $k$  repetitions form the set of observations per algorithm. These means are then statistically compared. For our case with  $m = 9$  techniques,  $n = 12$  tasks, and  $k = 5$  repetitions, this resulted in a set of  $9 \times 12 \times 5 = 540$  models in total.

Our experimental design was robust. Multiple tasks were used to evaluate the classifier (see Table 2). We carried out stratified splitting with respect to the training and validation datasets at the case level to avoid patient-specific data leakage. For a more realistic task adaptation scenario, only a fraction of all data was randomly sampled for training and validation. The proportions were chosen so that the smaller class sizes were roughly in the lower three-digit range, similarly to other experiments on task adaptation [22,24,53]. However, achieving equal dataset sizes among the different tasks was challenging due to varying class imbalances. Training and subsequent testing were repeated five times. We kept the stratified random samples constant across the evaluated algorithms [72]. For every task, we tested them on independent external test datasets. For computational efficiency, a stratified random subset with a maximum size of 100,000 patches was chosen for all datasets.

**Table 2.** Number of image patches per class for the training, validation, and test datasets. Training and validation data were split at the case level, while an external dataset was used for testing. Training and validation splits were extracted from the same image dataset.

Task	No.	Classes	Train	Validation	Test
Mitotic figure detection	#1	TUPAC16		"	BreCaHAD
		no mitosis	14,742	14,726	12,631
	#2	mitosis	100	100	115
		BreCaHAD		"	TUPAC16
Tumor metastasis detection	#3	no mitosis	5042	6041	99,326
		mitosis	46	55	674
	#4	Camelyon17, center 1		"	Camelyon17, center 2
		no metastasis	5040	5400	99,421
	#5	metastasis	100	100	579
		Camelyon17, center 2		"	Camelyon17, center 1
	#6	no metastasis	17,151	17,152	98,055
		metastasis	100	100	1945
	#7	Camelyon17, center 3		"	Camelyon17, center 4
		no metastasis	9473	9372	96,888
Tumor-infiltrating lymphocyte detection	#8	metastasis	100	100	3112
		Camelyon17, center 4		"	Camelyon17, center 3
	#9	no metastasis	3112	3113	98,945
		metastasis	100	100	1055
	#10	TCGA TILs, center 1		"	TCGA TILs, center 2
		TIL-negative	656	656	73,687
	#11	TIL-positive	100	100	18,030
		TCGA TILs, center 2		"	TCGA TILs, center 1
	#12	TIL-negative	408	408	86,778
		TIL-positive	100	100	13,222

Table 2. Cont.

Task	No.	Classes	Train	Validation	Test
Colorectal cancer tissue-type classification	#9	CRC5000		"	DRCO
		tumor	100	100	29,201
		stroma	100	100	32,166
		lympho	100	100	175
		debris	100	100	193
		mucosa	100	100	9565
		adipose	100	100	28,700
	#10	DRCO Small		"	CRC5000 Small
		tumor	305	305	625
		stroma	336	336	625
		mucosa	100	100	625
		adipose	300	300	625
Skin cancer tissue-type classification	#11	Queensland		"	DRSK
		SAS	56	56	5970
		INF	331	331	16,523
		HYP	1735	1735	41,060
		DER	4011	4010	30,845
		EPI	5	5	499
		BCC	149	149	3867
		SCC	416	416	1097
		IEC	100	100	139
	#12	DRSK Small		"	Queensland Small
		SAS	100	100	928
		INF	276	276	5403
		HYP	687	687	28,264
		DER	516	516	65,311
		EPI	8	8	94

TIL: tumor-infiltrating lymphocyte; SAS: skin appendage structure; INF: inflammation; HYP: hypodermis; DER: dermis; EPI: epidermis; BCC: basal cell carcinoma; SCC: squamous cell carcinoma; IEC: intraepidermal carcinoma.

The experimental design includes solid statistical validation. The area under the curve (AUC) was chosen as the final measure. It is a standard metric for comparing classification algorithms [72]. For each combination of tasks and algorithms, the mean and standard deviation of the AUC values were calculated and reported in the Results Section. We used statistical tests to verify the generalizability of our results beyond the tasks in our experimental setup [82]. The classical null hypothesis significance testing (NHST) procedure was recently criticized for being inappropriate for ML algorithm comparisons. We therefore used Bayesian hypothesis testing instead [71].

Bayesian testing is preferable because it avoids several shortcomings of classical NHST [71]. With respect to NHST, decisions are made based on the probability of observing an effect when the actual mean difference of two classifiers,  $A$  and  $B$ , is assumed to be 0 ( $H_0$  hypothesis). Bayesian testing, instead, can directly estimate the probability of  $A$  being better than  $B$  (and vice versa) based on some observations. Furthermore, NHST is based on the unrealistic assumption that two algorithms can potentially have equal performance. Thus, trivial effect sizes can become significant by increasing the number of observations. In Bayesian testing, one can define a region of practical equivalence (ROPE) to take this problem into account beforehand. NHST does not allow drawing conclusions from non-significant results. However, Bayesian testing can show the equivalence of two algorithms based on the ROPE.

In addition to Bayesian testing, we additionally studied effect sizes [83]. Given means  $m_A$  and  $m_B$  and the pooled standard deviation,  $\sigma_{pooled}$ , of two normally distributed populations  $A$  and  $B$ , Cohen's  $d$  is calculated as follows [84]:

$$d = \frac{m_A - m_B}{\sigma_{pooled}} \quad (1)$$

We omitted effect sizes in the pairwise comparison based on multiple datasets in Section 4.1. In this case, the denominator in Equation (1) reflects the classification tasks' difficulty and not the model's performance dispersion. As a nonparametric variant of Cohen's  $d$  for populations without a normal distribution, we adopted Akinshin's gamma from the Python package "Autorank" [85].

### 3.2.3. Comparison of Training Dataset Sizes

We were also interested in how the performance of the techniques under investigation would behave with a larger amount of data. The performance gain of vanilla fine-tuning is known to become saturated with the increase in the dataset size [8]. Some recent studies on advanced task adaptation techniques carried out comparisons with respect to the training datasets' size. Some techniques performed better in a large-scale setting with more than 1000 images per class, while others did not [22,54,56]. To evaluate the impact of dataset sizes in histopathology, we chose to re-evaluate some of the task adaptation techniques using additional experiments.

The experimental design for comparing multiple dataset sizes was based on the statistical comparison of two algorithms using one dataset, as described by Benavoli et al. [71]. In this case, we compared an algorithm trained on a small training split vs. trained on a larger training split of the same dataset. Therefore, for each combination of  $m$  techniques and  $n$  training splits,  $k$  repetitions of the model training were performed. We chose three techniques ( $L^2$ -SP, fine-tuning, and Co-Tuning) from our set of task adaptation algorithms. We re-used dataset #10 from the colorectal cancer tissue-type classification (see Table 2). Three variants of the training split were generated using a varying subsampling factor: "Base," "Large," and "XLarge" (see Table 3). Base is equivalent to the size shown in Table 2. For every combination of split size and learning technique, we trained  $k = 20$  models. With  $m = 3$  techniques,  $n = 3$  training split variants, and  $k = 20$  repetitions, this resulted in a set of  $3 \times 3 \times 20 = 180$  models in total. Again, the AUC was chosen as our metric for comparisons. We verified the results using the Bayesian correlated  $t$ -test. We further report effect sizes in the Results Section.

**Table 3.** Number of image patches per class for three different dataset sizes. The factor of the dataset size is provided in brackets.

Classes	Base ( $\times 1$ )		Large ( $\times 2.5$ )		XLarge ( $\times 10$ )	
	Train	Validation	Train	Validation	Train	Validation
tumor	305	305	763	763	3052	3052
stroma	336	336	840	840	3362	3362
mucosa	100	100	250	250	1000	1000
adipose	300	300	750	750	3000	3000

### 3.3. Exemplary Image Classification Pipeline

A DL pipeline for image classification typically includes multiple sequential components. Each component comes with its own set of design choices. Here, we limited ourselves to a single exemplary DL pipeline, used for all experiments in this study. The core building blocks are typically image preprocessing and the neural network itself. We first started with the definition of our image preprocessing. We then described the state-of-the-art CNN architecture. Finally, we addressed the hyperparameters of our pipeline.

#### 3.3.1. Image Preprocessing

Neural networks for image processing typically operate on smaller image patches. Therefore, we used square image patches with a default edge length of 128 pixels. The spatial resolution of the training and testing images were identical for all tasks, except for the skin cancer tissue-type classification. To account for the different spatial resolutions in the latter case, we used patch sizes with an edge length of  $128 \times (0.67/0.5) = 171$  pixels for

the DRSK dataset. Subsequently, all patches were scaled uniformly to  $224 \times 224$  pixels, the default input image size of ConvNeXt.

Inequalities in class distribution must be considered when training representation learning algorithms. An imbalance of varying magnitudes is present in all datasets, except for CRC5000. Oversampled minority classes were therefore used during training time. This prevents the classifier from becoming biased towards one or more specific class(es). For validation and testing, we kept the original class distribution of the datasets.

When training neural networks for image processing, increasing the training set size using various image augmentations is common. We used random horizontal flipping. Following the advice of Tellez et al. [36], we further used HSV color augmentation. As commonly carried out in PyTorch, the mean and standard deviation were normalized to [0.485, 0.456, 0.406] and [0.229, 0.224, 0.225] in the RGB color space, respectively.

### 3.3.2. CNN Architecture

Since the development of AlexNet in 2012 [86], neural networks based on convolution have been widely used as a standard approach for CV. Research has constantly improved network architecture around this fundamental principle, including architectural milestones like VGG [87], ResNet [88], Xception [89], or EfficientNet [90]. Recently, transformer-based architectures from natural language processing (NLP) challenged the use of convolution operations [91–94]. Other than CNNs, they are based on multi-head self-attention.

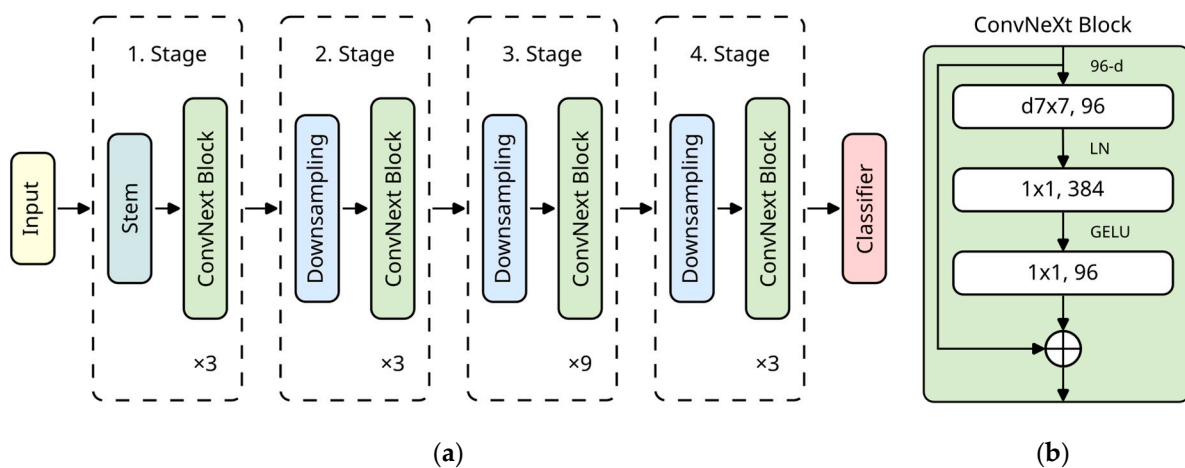
Whether to use convolution- or attention-based architectures for CV remains an open research question. Liu et al. [95] found that transformers gain importance compared to CNNs mainly because of their better scaling behavior. However, recent studies showed that CNNs can still achieve an equivalent or better performance than transformers after optimizing several design choices. One of these options is larger kernel sizes [95–99]. This raises the question of whether the recent performance of transformer models is really a result of multi-head self-attention. At the same time, the established convolution approach was designed around efficiency [95]. The new design principle of choosing large kernels also seems to push the networks to shift towards a shape bias [97]. This effect resolves a previous issue of pre-trained CNNs being biased due to texture [100].

For this study, we chose a state-of-the-art CNN architecture. As described above, the newest architectures achieved cutting-edge performance for image classification. Furthermore, most task adaptation techniques were explicitly designed for CNNs. Therefore, we selected ConvNeXt [95] to represent novel CNN architectures. We chose the smallest one, “ConvNeXt Tiny”, as we assumed it had enough capacity for our tasks while keeping training time manageable. The building blocks are shown in Figure 3. The pre-trained ImageNet weights originate from PyTorch [101]. We transferred these weights to all used models. Furthermore, we ensured that all implementations (both in PyTorch and Keras) produced the same model results (excluding rounding errors) so that different model implementations do not distort the results.

### 3.3.3. Hyperparameters

For our experiments, we preferred the stochastic gradient descent (SGD) optimizer over Adam, another common choice, based on two reasons. First, SGD generalizes better for image classification [102]. The performance of Adam seems to depend on the hyperparameter optimization of decay rates  $\beta_1$  and  $\beta_2$ . Second, Adam was only used in the study performed by Gouk et al. [52]. The authors of the other studies used SGD. From a practical perspective, this results in less implementation effort when using SGD as well.





**Figure 3.** (a) Overall architecture of ConvNeXt Tiny [95]. The macro-level architecture with a ratio of 1:1:3:1 was adopted from Swin Transformers [103]. (b) Detailed structure of the ConvNeXt block [95]. The micro-level architecture is based on the inverted bottleneck design [104]. Inspired by transformer architectures, the depthwise convolution uses a kernel size of  $7 \times 7$ . For computational efficiency, it is computed before the expansion of channels. The Gaussian error linear unit (GELU) [105] and layer normalization [106] were also adopted from recent transformer-based architectures [95]. Adapted from Jiang et al. [107], © 2023, Frontiers Media S.A., CC BY 4.0 DEED.

Broadly consistent parameters for SGD were used across the algorithms. In line with recent recommendations on fine-tuning in dissimilar source and target domains [17], we used SGD with an initial learning rate of 0.01 and a momentum of 0.9 and activated Nesterov. We adopted step decay from the majority of the evaluated studies [7,23,24,52–54,61]. We consistently set a milestone at the 10th epoch with a gamma value of 0.1. We used a lower learning rate for the pre-trained layers in the fine-tuning setting. We also adopted lower learning rates for the policy network of SpotTune and the last block in MultiTune from the original studies [23,61]. To avoid exploding gradients, we applied gradient value clipping with a value of 1.0. We chose to use a batch size of 64. During training, iterations per epoch were fixed to 128. Models were trained with early stopping, patience of 15, and a maximum number of 100 epochs.

Various hyperparameters must be specified for the algorithms under study. However, conducting hyperparameter optimization to identify optimal parameters for each task on each dataset was not practical due to the substantial computation time required. Instead, we utilized the suggested values from the literature. Li et al. [108] recently found that  $L^2$ -SP is robust across different parameter ranges. However,  $\alpha > \beta$  is generally preferable. Therefore, we extracted  $\alpha = 0.1$  and  $\beta = 0.01$  from the original study [24]. For MARS-PGM, we used the average values of  $\gamma_j = 9.32$  and  $\gamma_L = 16.89$  [52]. For DELTA, Li et al. [53] fixed  $\beta = 0.01$  and generally used it for all conditions [53]. We further set  $\alpha = 0.04$ . For Co-Tuning, You et al. [22] found that  $\lambda = 2.3$  is robust across all tested datasets and sampling rates. For BSS, Chen et al. [55] found that  $\eta = 0.001$  and  $k = 1$  are generally adequate settings under varying conditions. Zhong et al. [54] universally applied  $\tau = 0.07$  and  $m = 0.999$  for Bi-Tuning. For a 100% sampling rate, which is comparable to our class size, they found that a high number of sampling keys,  $K$ , is beneficial. In line with this, we therefore used the following defaults:  $K = 40$  and  $L = 128$ . Wang et al. [61] used  $\alpha = 0.01$  and  $\beta = 0.01$  for all their experiments.

For our baseline, vanilla fine-tuning, the model weights were copied from the ImageNet domain. The pre-trained classification head was replaced by a new one with randomly initialized weights. The new model was then fine-tuned on the histopathology tasks. For the pre-trained feature extractor, a smaller learning rate (10 times smaller) was chosen.

### 3.4. Hardware and Software Implementation

Our server had two AMD EPYC 7402 24-core processors, one terabyte of random-access memory, NVIDIA RTX A6000 graphics cards, and an Ubuntu 22.04 LTS operating system. Various freely available software packages were used. All experiments were implemented in Python (ver. 3.9.16). WSI data were loaded using the Python package tiffslide (ver. 2.1.2) [109]. Model training was implemented using PyTorch (ver. 2.0.0) [110], Keras (ver. 2.11.0), and TensorFlow (ver. 2.11.1) [111] libraries. For PyTorch training, image augmentations used the Transforms module from Torchvision (ver. 0.15.1). For Keras training, the same augmentations were implemented using the Albumentations (ver. 1.3.0) package [112]. For the calculation of general sample statistics and the Bayesian signed-rank test, we used the implementation of the Python package Autorank (ver. 1.2.0) [85]. The Bayesian correlated *t*-test was calculated using baycomp (ver. 1.0.3) [113].

## 4. Results

In the following, the experimental results are presented. As defined in Section 3.2.2, we begin by comparing different task adaptation techniques. We then compare the accuracy of several algorithms for different training data sizes, as described in Section 3.2.3.

### 4.1. Comparison of the Task Adaptation Techniques

As described in Section 3.2.2, we trained 540 models for our first experiment on task adaptation techniques across different tasks. Table 4 shows the aggregated results. Table 5 shows some general statistics (median, median absolute deviation from the median, and confidence interval for the median) of the results. The statistics were ranked using the median AUC in descending order. Furthermore, the effect size was provided for the difference from the best algorithm using Akinshin's gamma. Based on these results, we carried out a Bayesian signed-rank test for nine algorithms with paired samples ( $n = 12$ ) and significance levels of  $P(\cdot) \geq 0.9, 0.95$ , and  $0.99$ . Using the Shapiro–Wilk test for normality with Bonferroni correction, Autorank failed to reject the null hypothesis of normal distributions for MARS-PGM ( $p = 0.002$ ). Therefore, a normal distribution of the results was not given for all algorithms. Based on the recommendations of Kruschke and Liddell [114], Autorank used  $0.1 \times \text{MAD}$  for the region of practical equivalence (ROPE) around the median.

Autorank reported significant and practically relevant differences between the algorithms  $L^2$ -SP (MD =  $0.930 \pm 0.096$ ; MAD = 0.021), fine-tuning (MD =  $0.911 \pm 0.105$ ; MAD = 0.042), Co-Tuning (MD =  $0.898 \pm 0.082$ ; MAD = 0.038), DELTA (MD =  $0.893 \pm 0.136$ ; MAD = 0.040), SpotTune (MD =  $0.889 \pm 0.093$ ; MAD = 0.020), BSS (MD =  $0.881 \pm 0.100$ ; MAD = 0.044), Bi-Tuning (MD =  $0.878 \pm 0.086$ ; MAD = 0.033), MultiTune (MD =  $0.849 \pm 0.086$ ; MAD = 0.051), and MARS-PGM (MD =  $0.802 \pm 0.215$ ; MAD = 0.037). Compared to the top-ranking algorithm,  $L^2$ -SP, the mean difference was inconclusive for fine-tuning and Co-Tuning. DELTA was significantly smaller ( $P(\ll) = 0.949$ ) than  $L^2$ -SP with a medium magnitude. There was a significant difference with a large magnitude for SpotTune ( $P(\ll) = 0.962$ ), Bi-Tuning ( $P(\ll) = 0.934$ ), MultiTune ( $P(\ll) = 1.0$ ), and MARS-PGM ( $P(\ll) = 1.0$ ).

Table 6 shows the results of the Bayesian signed-rank test for all possible pair-wise comparisons. Figure A1 in Appendix A shows the simplex plots from Monte Carlo sampling. Table A1 in Appendix B lists the exact aggregated probability values for all comparisons. The pair-wise mean differences were inconclusive for the first three algorithms ( $L^2$ -SP, fine-tuning, and Co-Tuning). With some exceptions, at least one of these algorithms achieved significantly better results than the following two-thirds. This first group was followed by a second group of algorithms (DELTA, SpotTune, BSS, and Bi-Tuning) with inconclusive pair-wise differences. Finally, the remaining two algorithms (MultiTune and SpotTune) showed significant negative mean differences in almost all cases. At this point, we emphasize that inconclusive results do not necessarily constitute evidence of equal performance.

**Table 4.** Mean and standard deviation of AUC values and per-task rank (in parentheses) for different task adaptation procedures.

Task	Baseline	Distance Regularization			Feature Space Regularization		Layer Routing		Shared Domains
	Fine-Tuning	DELTA	L <sup>2</sup> -SP	MARS-PGM	Bi-Tuning	BSS	MultiTune	SpotTune	Co-Tuning
#1	0.954 ± 0.014 (1)	0.926 ± 0.020 (5)	0.952 ± 0.014 (3)	0.780 ± 0.035 (9)	0.918 ± 0.023 (6)	0.939 ± 0.012 (4)	0.811 ± 0.044 (8)	0.901 ± 0.020 (7)	0.954 ± 0.019 (2)
#2	0.937 ± 0.028 (3)	0.676 ± 0.130 (9)	0.941 ± 0.016 (2)	0.770 ± 0.129 (8)	0.878 ± 0.071 (6)	0.919 ± 0.030 (4)	0.784 ± 0.139 (7)	0.887 ± 0.033 (5)	0.948 ± 0.009 (1)
#3	0.903 ± 0.029 (3)	0.897 ± 0.061 (4)	0.926 ± 0.023 (1)	0.763 ± 0.125 (9)	0.849 ± 0.031 (8)	0.893 ± 0.024 (5)	0.853 ± 0.044 (7)	0.866 ± 0.021 (6)	0.910 ± 0.034 (2)
#4	0.919 ± 0.024 (3)	0.943 ± 0.014 (1)	0.934 ± 0.019 (2)	0.804 ± 0.090 (8)	0.882 ± 0.049 (6)	0.832 ± 0.044 (7)	0.767 ± 0.107 (9)	0.906 ± 0.021 (4)	0.886 ± 0.035 (5)
#5	0.949 ± 0.010 (2)	0.948 ± 0.009 (3)	0.949 ± 0.009 (1)	0.837 ± 0.075 (9)	0.932 ± 0.007 (6)	0.940 ± 0.012 (4)	0.913 ± 0.014 (7)	0.891 ± 0.070 (8)	0.936 ± 0.010 (5)
#6	0.861 ± 0.051 (5)	0.848 ± 0.073 (8)	0.890 ± 0.042 (2)	0.473 ± 0.442 (9)	0.865 ± 0.050 (3)	0.864 ± 0.055 (4)	0.853 ± 0.036 (7)	0.890 ± 0.035 (1)	0.855 ± 0.076 (6)
#7	0.937 ± 0.007 (2)	0.899 ± 0.049 (9)	0.934 ± 0.012 (4)	0.902 ± 0.025 (8)	0.934 ± 0.012 (5)	0.938 ± 0.007 (1)	0.915 ± 0.015 (7)	0.916 ± 0.008 (6)	0.936 ± 0.010 (3)
#8	0.870 ± 0.023 (3)	0.888 ± 0.018 (1)	0.861 ± 0.024 (6)	0.800 ± 0.086 (9)	0.874 ± 0.017 (2)	0.870 ± 0.026 (4)	0.818 ± 0.117 (8)	0.849 ± 0.028 (7)	0.867 ± 0.021 (5)
#9	0.865 ± 0.011 (7)	0.874 ± 0.007 (3)	0.868 ± 0.011 (4)	0.831 ± 0.022 (9)	0.879 ± 0.008 (1)	0.866 ± 0.012 (5)	0.848 ± 0.019 (8)	0.865 ± 0.006 (6)	0.877 ± 0.005 (2)
#10	0.960 ± 0.009 (1)	0.928 ± 0.046 (8)	0.942 ± 0.032 (6)	0.856 ± 0.110 (9)	0.948 ± 0.028 (5)	0.953 ± 0.023 (4)	0.939 ± 0.025 (7)	0.959 ± 0.021 (2)	0.957 ± 0.015 (3)
#11	0.751 ± 0.036 (8)	0.793 ± 0.048 (1)	0.761 ± 0.037 (6)	0.737 ± 0.085 (9)	0.775 ± 0.011 (3)	0.753 ± 0.017 (7)	0.766 ± 0.010 (5)	0.773 ± 0.030 (4)	0.793 ± 0.022 (2)
#12	0.826 ± 0.020 (8)	0.808 ± 0.056 (9)	0.836 ± 0.036 (7)	0.845 ± 0.023 (4)	0.842 ± 0.079 (5)	0.842 ± 0.038 (6)	0.851 ± 0.048 (3)	0.875 ± 0.033 (1)	0.872 ± 0.015 (2)
Median (Avg.)	0.911 (3.83)	0.893 (5.08)	0.930 (3.67)	0.802 (8.33)	0.878 (4.67)	0.881 (4.58)	0.849 (6.92)	0.889 (4.75)	0.898 (3.17)

**Table 5.** Descriptive statistics for the ranked algorithms.

Algorithm	Median	MAD	CI	$\gamma$	Magnitude	$P(\ll)$	$P(=)$
L <sup>2</sup> -SP	0.930	0.021	[0.761, 0.952]	–	–	–	–
Fine-Tuning	0.911	0.042	[0.751, 0.960]	0.391	small	0.647	0.332
Co-Tuning	0.898	0.038	[0.793, 0.957]	0.699	medium	0.408	0.021
DELTA	0.893	0.040	[0.676, 0.948]	0.788	medium	0.948	0.003
SpotTune	0.889	0.020	[0.773, 0.959]	1.365	large	0.963	0.001
BSS	0.881	0.044	[0.753, 0.953]	0.947	large	0.899	0.093
Bi-Tuning	0.878	0.033	[0.775, 0.948]	1.276	large	0.934	0.001
MultiTune	0.849	0.051	[0.766, 0.939]	1.390	large	1.000	0.000
MARS-PGM	0.802	0.037	[0.473, 0.902]	2.898	large	1.000	0.000

MAD: mean absolute deviation of the median; CI: confidence interval;  $\gamma$ : Akinshin’s gamma.

**Table 6.** Decision matrix for the pair-wise comparison of the algorithms. The algorithms are sorted by descending median performance.

Algorithm	L <sup>2</sup> -SP	Fine-Tuning	Co-Tuning	DELTA	Spot-Tune	BSS	Bi-Tuning	MultiTune	MARS-PGM
L <sup>2</sup> -SP	—	ns	ns	≤ *	≤ **	ns	≤ *	≤ ***	≤ ***
Fine-Tuning	ns	—	ns	ns	≤ *	ns	≤ *	≤ ***	≤ ***
Co-Tuning	ns	ns	—	≤ *	≤ **	≤ *	ns	≤ ***	≤ ***
DELTA	≥ *	ns	≥ *	—	ns	ns	ns	ns	≤ ***
SpotTune	≥ **	≥ *	≥ **	ns	—	ns	ns	≤ ***	≤ ***
BSS	ns	ns	≥ *	ns	ns	—	ns	≤ ***	≤ ***
Bi-Tuning	≥ *	≥ *	ns	ns	ns	ns	—	≤ ***	≤ ***
MultiTune	≥ ***	≥ ***	≥ ***	ns	≥ ***	≥ ***	≥ ***	—	≤ ***
MARS-PGM	≥ ***	≥ ***	≥ ***	≥ ***	≥ ***	≥ ***	≥ ***	≥ ***	—

≤: negative mean difference; ≥: positive mean difference; \*:  $P(\cdot) \geq 0.9$ ; \*\*:  $P(\cdot) \geq 0.95$ ; \*\*\*:  $P(\cdot) \geq 0.99$ ; ns: not significant.

While our baseline (fine-tuning) performed comparatively well overall, it only ranked 3.83 on average across all datasets with respect to the descriptive statistics shown in Table 4. The question is whether it is still the best choice for all histopathological tasks. To better understand the influencing factors, we compared it with all other task adaptation techniques at the task level using the Bayesian correlated  $t$ -test. The aggregated results are shown in Table 7. The exact probabilities and effect sizes are also listed in Appendix C. For 5 of the 12 tasks, significantly better results than vanilla fine-tuning with medium or large effects were obtained by one or more task adaptation techniques. Co-Tuning and DELTA were the most frequently represented here, with three significant results each. SpotTune was significantly superior in two cases. Both Bi-Tuning and L<sup>2</sup>-SP achieved significantly better results than using the fine-tuning process once. In Table A2, one can further find a large positive effect ( $P(\geq) = 0.896$ ,  $\gamma = 1.692$ ) for DELTA and a medium positive effect ( $P(\geq) = 0.890$ ,  $d = 0.626$ ) for L<sup>2</sup>-SP. In both cases, Bayesian testing narrowly missed the threshold for automatic decisions. Thus, none of the algorithms consistently performed better than the baseline across histopathology. However, the indifference in results is partly explained by the differences in classification tasks. The positive findings include both binary and multiclass classification tasks.

**Table 7.** Decision matrix for the pair-wise comparison of vanilla fine-tuning with all other algorithms on the task level.

Task	BSS	Bi-Tuning	Co-Tuning	DELTA	L <sup>2</sup> -SP	MARS-PGM	MultiTune	Spot-Tune
#1	≤ **, l	≤ *, l	ns	ns	ns	≤ ***, l	≤ ***, l	≤ ***, l
#2	ns	ns	ns	≤ **, l	ns	≤ **, l	≤ *, l	ns
#3	ns	≤ ***, l	ns	ns	≥ *, l	≤ *, l	≤ *, l	≤ **, l
#4	≤ **, l	ns	ns	≥ *, l	ns	ns	≤ *, l	ns
#5	≤ **, m	≤ *, l	≤ **, l	ns	ns	≤ **, l	≤ **, l	ns
#6	ns	ns	ns	ns	ns	ns	ns	ns
#7	ns	ns	ns	ns	ns	≤ *, l	≤ **, l	≤ ***, l
#8	ns	ns	ns	ns	ns	ns	ns	ns
#9	ns	≥ ***, l	≥ *, l	≥ *, l	ns	≤ *, l	≤ *, l	ns
#10	ns	ns	ns	ns	ns	ns	ns	ns
#11	ns	ns	≥ *, l	≥ *, l	ns	ns	ns	≥ *, m
#12	ns	ns	≥ *, l	ns	ns	ns	ns	≥ **, l

≤: negative mean difference; ≥: positive mean difference; \*:  $P(\cdot) \geq 0.9$ ; \*\*:  $P(\cdot) \geq 0.95$ ; \*\*\*:  $P(\cdot) \geq 0.99$ ; ns: not significant; m: medium effect; l: large effect. Significant positive mean differences ( $P(\geq) \geq 0.9$ ) are additionally highlighted in bold.

#### 4.2. Comparison of Training Dataset Size

As described in Section 3.2.3, we trained 180 models for the comparison of the dataset's size. The aggregated training results based on AUC, balanced accuracy, and F1 score for different dataset sizes are presented in Tables 8–10, respectively. For the dataset size XLarge, Co-Tuning achieved the highest absolute increase in performance in terms of AUC (+0.013), balanced accuracy (+0.045), and F1 score (+0.048).

**Table 8.** Mean and standard deviation of the AUC for fine-tuning, L<sup>2</sup>-SP, and Co-Tuning relative to different dataset size settings.

Dataset	Baseline	Distance Regularization	Shared Domains
	Fine-Tuning	L <sup>2</sup> -SP	Co-Tuning
Base	0.957 ± 0.016	0.955 ± 0.022	0.958 ± 0.014
Large	0.961 ± 0.009	0.960 ± 0.011	0.966 ± 0.010
XLarge	0.962 ± 0.013	0.966 ± 0.009	0.971 ± 0.007
Median	0.961	0.960	0.966

**Table 9.** Mean and standard deviation of the balanced accuracy for fine-tuning, L<sup>2</sup>-SP, and Co-Tuning relative to different dataset size settings.

Dataset	Baseline	Distance Regularization	Shared Domains
	Fine-Tuning	L <sup>2</sup> -SP	Co-Tuning
Base	0.782 ± 0.050	0.796 ± 0.065	0.787 ± 0.048
Large	0.802 ± 0.032	0.797 ± 0.035	0.818 ± 0.032
XLarge	0.807 ± 0.036	0.819 ± 0.028	0.832 ± 0.023
Median	0.802	0.797	0.818

**Table 10.** Mean and standard deviation of the F1 score for fine-tuning, L<sup>2</sup>-SP, and Co-Tuning relative to different dataset size settings.

Dataset	Baseline	Distance Regularization	Shared Domains
	Fine-Tuning	L <sup>2</sup> -SP	Co-Tuning
Base	0.776 ± 0.055	0.791 ± 0.072	0.781 ± 0.055
Large	0.798 ± 0.034	0.794 ± 0.037	0.815 ± 0.036
XLarge	0.803 ± 0.037	0.817 ± 0.029	0.829 ± 0.023
Median	0.798	0.794	0.815

Using the Bayesian correlated *t*-test, we carried out pair-wise comparisons between the algorithms for all three dataset sizes. Tables 11, 13 and 15 show descriptive sample statistics and Tables 12, 14 and 16 show the decision matrices for the settings Base, Large, and XLarge, respectively. The posterior distributions and probability values are presented in Figure A2 and Table A3–A5, respectively.

We started with the dataset size “Base”. According to the Shapiro–Wilk test for normality, the results follow a normal distribution. Therefore, we used parametric statistics to describe the samples (Table 11). The mean differences between Co-Tuning and the other two techniques were negligible in magnitude. For Bayesian testing, we used  $0.1 \times \text{STD}$  for the ROPE. Similarly to our experiments in Section 4.1, Bayesian testing produced inconclusive results (Table 12).

**Table 11.** Descriptive statistics for the ranked algorithms using the dataset size “Base”.

Algorithm	Mean	STD	CI	<i>d</i>	Magnitude	<i>P</i> (≪)	<i>P</i> (=)
Co-Tuning	0.958	0.014	[0.949, 0.967]	–	–	–	–
Fine-Tuning	0.957	0.016	[0.947, 0.967]	0.079	negligible	0.473	0.241
L <sup>2</sup> -SP	0.955	0.022	[0.941, 0.969]	0.197	negligible	0.596	0.173

STD: standard deviation; CI: confidence interval; *d*: Cohen’s *d*.



**Table 12.** Decision matrix for the pair-wise Bayesian correlated *t*-test of algorithms using the dataset size “Base”.

Algorithm	Fine-Tuning	L <sup>2</sup> -SP	Co-Tuning
Fine-Tuning	—	ns	ns
L <sup>2</sup> -SP	ns	—	ns
Co-Tuning	ns	ns	—

<sup>ns</sup>: not significant.

We continued with the dataset size “Large”. According to the Shapiro–Wilk test for normality, the results follow a normal distribution. Therefore, we used parametric statistics to describe the samples (Table 13). There were medium differences between Co-Tuning and the other two algorithms. For Bayesian testing, we used  $0.1 \times \text{STD}$  for the ROPE. The differences mentioned above were inconclusive in Bayesian testing (Table 14).

**Table 13.** Descriptive statistics for the ranked algorithms using the dataset size “Large”.

Algorithm	Mean	STD	CI	<i>d</i>	Magnitude	<i>P</i> ( $\ll$ )	<i>P</i> (=)
Co-Tuning	0.966	0.010	[0.959, 0.972]	—	—	—	—
Fine-Tuning	0.961	0.009	[0.955, 0.966]	0.518	medium	0.830	0.088
L <sup>2</sup> -SP	0.960	0.011	[0.953, 0.967]	0.544	medium	0.849	0.081

STD: standard deviation; CI: confidence interval; *d*: Cohen’s *d*.

**Table 14.** Decision matrix for the pair-wise Bayesian correlated *t*-test of algorithms using the dataset size “Large”.

Algorithm	Fine-Tuning	L <sup>2</sup> -SP	Co-Tuning
Fine-Tuning	—	ns	ns
L <sup>2</sup> -SP	ns	—	ns
Co-Tuning	ns	ns	—

<sup>ns</sup>: not significant.

We further continued with the dataset size “XLarge”. According to the Shapiro–Wilk test for normality, the results do not follow a normal distribution. Therefore, we used nonparametric statistics to describe the samples (Table 15). For Bayesian testing, we used  $0.1 \times \text{MAD}$  for the ROPE. There was a significant medium increase ( $P(\gg) = 0.951$ ,  $\gamma = 0.748$ ) for Co-Tuning over vanilla fine-tuning (Table 16). Overall, the results of the Bayesian testing across all three settings of our comparison of the datasets’ size thus confirm the increase in accuracy for the size XLarge.

**Table 15.** Descriptive statistics for the ranked algorithms using the dataset size “XLarge”.

Algorithm	Median	MAD	CI	$\gamma$	Magnitude	<i>P</i> ( $\ll$ )	<i>P</i> (=)
Co-Tuning	0.972	0.006	[0.964, 0.981]	—	—	—	—
L <sup>2</sup> -SP	0.967	0.005	[0.958, 0.976]	0.619	medium	0.876	0.063
Fine-Tuning	0.965	0.006	[0.951, 0.979]	0.748	medium	0.951	0.025

MAD: mean absolute deviation of the median; CI: confidence interval;  $\gamma$ : Akinshin’s gamma.

**Table 16.** Decision matrix for the pair-wise Bayesian correlated *t*-test of algorithms using the dataset size “XLarge”.

Algorithm	Fine-Tuning	L <sup>2</sup> -SP	Co-Tuning
Fine-Tuning	—	ns	$\gg$ **, m
L <sup>2</sup> -SP	ns	—	ns
Co-Tuning	$\ll$ **, m	ns	—

$\ll$ : negative mean difference;  $\gg$ : positive mean difference; \*\*:  $P(\cdot) \geq 0.95$ ; <sup>ns</sup>: not significant; m: medium effect.

## 5. Discussion

To the best of our knowledge, we are the first to provide extensive empirical evidence on advanced task adaptation techniques in histopathology. We showed that specific techniques are, on average, better suited for histopathology than others. Interestingly, the authors of all methods investigated in this paper observed, in their original studies, an improvement of their method over the baseline fine-tuning technique [23,52,54]. However, as shown in Table 6, this is not confirmed by our results for the histopathology setting. Vanilla fine-tuning significantly outperformed SpotTune, Bi-Tuning, MultiTune, and MARS-PGM. Three task adaptation techniques combined, namely,  $L^2$ -SP, vanilla fine-tuning, and Co-Tuning, outperformed the others regarding classification tasks in digital histopathology. Among the three top-performing algorithms were distance regularization ( $L^2$ -SP), shared domains (Co-Tuning), and the baseline technique (fine-tuning). Moreover, the worst-performing algorithms (MultiTune and MARS-PGM) were significantly outperformed by almost all other techniques, whereas MARS-PGM was significantly worse than MultiTune.

However, our results reveal that the superiority of an algorithm in histopathology depends on the task. Our systematic comparison, summarized in Table 6, did not show a general superiority of one task adaptation technique over the others. Also, none of the evaluated algorithms showed performances that were significantly superior to our baseline. However, the comparison between the three highest-ranking algorithms ( $L^2$ -SP, fine-tuning, and Co-Tuning) was inconclusive as well. At first glance, this might be either due to our experimental setup, or performances might further depend on other influencing factors. Zhang et al. [33] already mentioned that, in addition to domain divergence and source/target data quality, the fit between the TL algorithm and the task is essential. Our task-level analysis confirmed this for histopathology (see Table 7). At least one technique obtained significantly better results than the baseline in about half of the cases.

Advanced task adaptation algorithms can also be helpful when a large amount of data are available. Our comparison of the dataset's size showed that, by increasing the dataset size by a factor of ten, Co-Tuning was able to outperform the baseline technique significantly. This is in line with the results of You et al. [22]. They found that additional supervision by Co-Tuning helps to further increase performances in large-scale classification settings where regularization techniques, like  $L^2$ -SP, do not help. Two other methods, which we did not examine in this large-scale setting, reported related results: the study on Bi-Tuning also reported a superior performance [54], while the study on StochNorm [56] obtained mixed findings.

The architecture of a neural network might be an influencing factor for the effectiveness of task adaptation. By choosing ConvNeXt, our investigation was based on an up-to-date CNN architecture from the field of CV. In general, ImageNet performance is a good indicator of the accuracy of CNN architectures relative to a downstream task [8]. However, Ding et al. [97] recently showed that CNNs with larger kernel sizes (like ConvNeXt) have higher shape biases compared to texture biases. In general research on CV, this is generally seen as beneficial due to its similarity with human perception [97,100,115]. While this might be true for photograph-like image data, histopathological analysis at a high magnification relies on relatively small cell structures and the visual texture of the tissue for decisions.

The algorithm-specific influence of histopathology pre-training needs to be further clarified. Image features from ImageNet and histopathology are likely to be different, and domain divergence is assumed to affect the result negatively [33]. For vanilla fine-tuning, three studies found that histopathology benefits from domain-specific pre-training [34,35,116], while two did not [117,118]. The question arises with respect to how advanced task adaptation techniques benefit from domain-specific pretraining and whether this effect diverges between algorithms.

Our results are in line with other medical studies. Several authors already provided evidence for the usefulness of advanced task adaptation techniques in a broader medical context. Most examples can be found in radiology. Liao et al. [25] used  $L^2$ -SP to avoid catastrophic forgetting in a multitask setting called "MUSCLE". They found that  $L^2$ -SP

was beneficial in all tested use cases. Sagie et al. [26] evaluated  $L^2$ -SP in a supervised task adaptation setting for segmentation.  $L^2$ -SP was not beneficial for ImageNet pre-training. However, it improved performances with respect to medical pre-training. Su et al. [27] successfully used  $L^2$ -SP to carry out the supervised adaptation of an MRI-pre-trained GAN relative to CT image generation. Unfortunately, they did not compare their results to vanilla fine-tuning. An et al. [28] evaluated SpotTune in a supervised task adaptation setting for segmentation, which was pre-trained on a related medical dataset. SpotTune performed better than vanilla fine-tuning. Another example can be found in lung sound classification. Nguyen and Pernkopf [29] evaluated Co-Tuning, Stochastic Normalization, and a combination of both for supervised task adaptation with ImageNet pre-training. While Co-Tuning and StochNorm were superior to vanilla fine-tuning, their performance varied between classification tasks.

Our results add empirical evidence to CV research for the general relationship between domain divergence and negative transfer. The findings are based on eight techniques applied to the medical domain. We showed that the relative effectiveness of task adaptation techniques in histopathology diverges from general CV research. This observation confirms the assumption that performance depends on the domain [33]. Our findings might have implications for evaluating new transfer learning techniques. Increasing the variety in benchmark tasks has been proposed before [14,19]. In this paper, we confirmed that the evaluation of commonly used datasets containing animals, cars, aircraft, plants, etc., is insufficient.

However, the findings on some techniques also deviate from earlier findings in CV research in some ways. First, Li et al. [17] reported that distance regularization does not perform well for dissimilar source and target domains. Plested et al. [16] thus suggested that distance regularization should be limited to earlier layers, and the number of layers depends on the downstream dataset. Surprisingly, regular  $L^2$ -SP distance regularization performed comparatively well in histopathology. Second, our results also contradict the conclusion of Gouk et al. [52], who reported that hard constraints perform better than distance constraints using the penalty term. MARS-PGM performed the worst of the algorithms compared (see Table 6).

## 6. Conclusions

This study is the first to provide empirical evidence on the suitability of up-to-date task adaptation techniques for histopathological classification tasks using CNNs. Analysis at the task level showed that, in half of the cases, one or multiple techniques, including Bi-Tuning, Co-Tuning, DELTA,  $L^2$ -SP, and SpotTune, could outperform vanilla fine-tuning, which is the standard procedure for task adaptation. Co-Tuning can further increase performance when a large amount of data in the target domain are available. Our results align with earlier studies mainly from the radiological field, showing that medical AI can benefit from advanced task adaptation techniques. We further provided evidence on the general relationship between domain divergence and negative transfer. In this paper, we found some deviations from the literature on distance regularization.

### 6.1. Limitations

Our study was limited to supervised task adaptation techniques and excluded related research areas, like domain adaptation and zero-, one-, and few-shot learning. As we narrowed down the evaluated task adaptation techniques to the most promising ones in Section 3.1 for theoretical and practical reasons, our study did not cover 100% of all available methods from the literature. Furthermore, we limited our study to techniques that were initially designed for image classification. Our experiments did not combine BSS with distance regularization using  $L^2$ -SP or DELTA. Although the classification tasks in Section 3.2 were based on 12 different datasets covering various cancer types and classification tasks, not all possible scenarios from histopathology were fully covered.

While we demonstrated relative improvements over the baseline approach (fine-tuning) for several tasks, our results do not show the best possible performance improvement due to the substantial computational costs required in our experimental setup for extensive hyperparameter optimization. We did not further investigate performances under varying influencing factors beyond dataset size—like different source domains, unsupervised pre-training, and the network’s architecture and size—or other downstream ML tasks—like instance segmentation. In our experiments, the number of layers using distance regularization was not adjusted relative to downstream tasks.

## 6.2. Outlook

This relationship between the histopathological task and the suitability of task adaptation techniques needs to be better understood in the future for task adaptation to be used in a targeted manner. More generally, the differentiated evaluation of transfer learning algorithms should continue so that more domains beyond ImageNet-like photographs can benefit more from task adaptation. Future studies should examine the impact of pre-training in pathology on task adaptation outcomes. More techniques, including Bi-Tuning [54] and StochNorm [56], should be investigated concerning their behavior in a large-scale setting.

**Author Contributions:** Conceptualization, D.S. (Daniel Sauter) and M.K.; methodology, D.S. (Daniel Sauter); software, D.S. (Daniel Sauter); validation, D.S. (Daniel Sauter), investigation, D.S. (Daniel Sauter), G.L., F.N., D.S. (Dirk Schadendorf), E.L. and M.K.; data curation, D.S. (Daniel Sauter); writing—original draft preparation, D.S. (Daniel Sauter) and M.K.; writing—review and editing, D.S. (Daniel Sauter), G.L., F.N., D.S. (Dirk Schadendorf), E.L. and M.K.; visualization, D.S. (Daniel Sauter); supervision, G.L., F.N., D.S. (Dirk Schadendorf), E.L. and M.K. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was funded by a Ph.D. grant from the DFG Research Training Group 2535 Knowledge- and data-based personalization of medicine at the point of care (WisPerMed), University of Duisburg-Essen, Germany.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Publicly available datasets were analyzed in this study. The BreCaHAD dataset is available from Figshare at <https://doi.org/10.6084/m9.figshare.7379186> (accessed on 9 March 2023). The TUPAC16 dataset can be found at: <https://tupac.grand-challenge.org/> (accessed on 9 March 2023). The CAMELYON17 dataset can be found at: <https://camelyon17.grand-challenge.org/> (accessed on 7 March 2023). The dataset for TIL classification is available from Zenodo at <https://zenodo.org/record/6604094> (accessed on 28 June 2023). The CRC-5000 dataset is available from Zenodo at <https://zenodo.org/record/53169> (accessed on 9 March 2023). The DRCO dataset (K. Lindman, M. Lindvall, C. B. Stadler, C. Lundstrom, and D. Treanor, 2019, Colon data from the Visual Sweden project DROID) is available upon request at <https://datahub.aida.scilifelab.se/10.23698/aida/drco> (accessed on 4 April 2023). The Queensland dataset is available from UQ eSpace at <https://espace.library.uq.edu.au/view/UQ:8be4bd0> (accessed on 8 March 2023). The DRSK dataset (K. Lindman, J. F. Rose, M. Lindvall, and C. B. Stadler, 2019, Skin data from the Visual Sweden project DROID) is available upon request at <https://datahub.aida.scilifelab.se/10.23698/aida/drsk> (accessed on 4 April 2023).

**Conflicts of Interest:** The authors declare no conflicts of interest.

# Appendix A

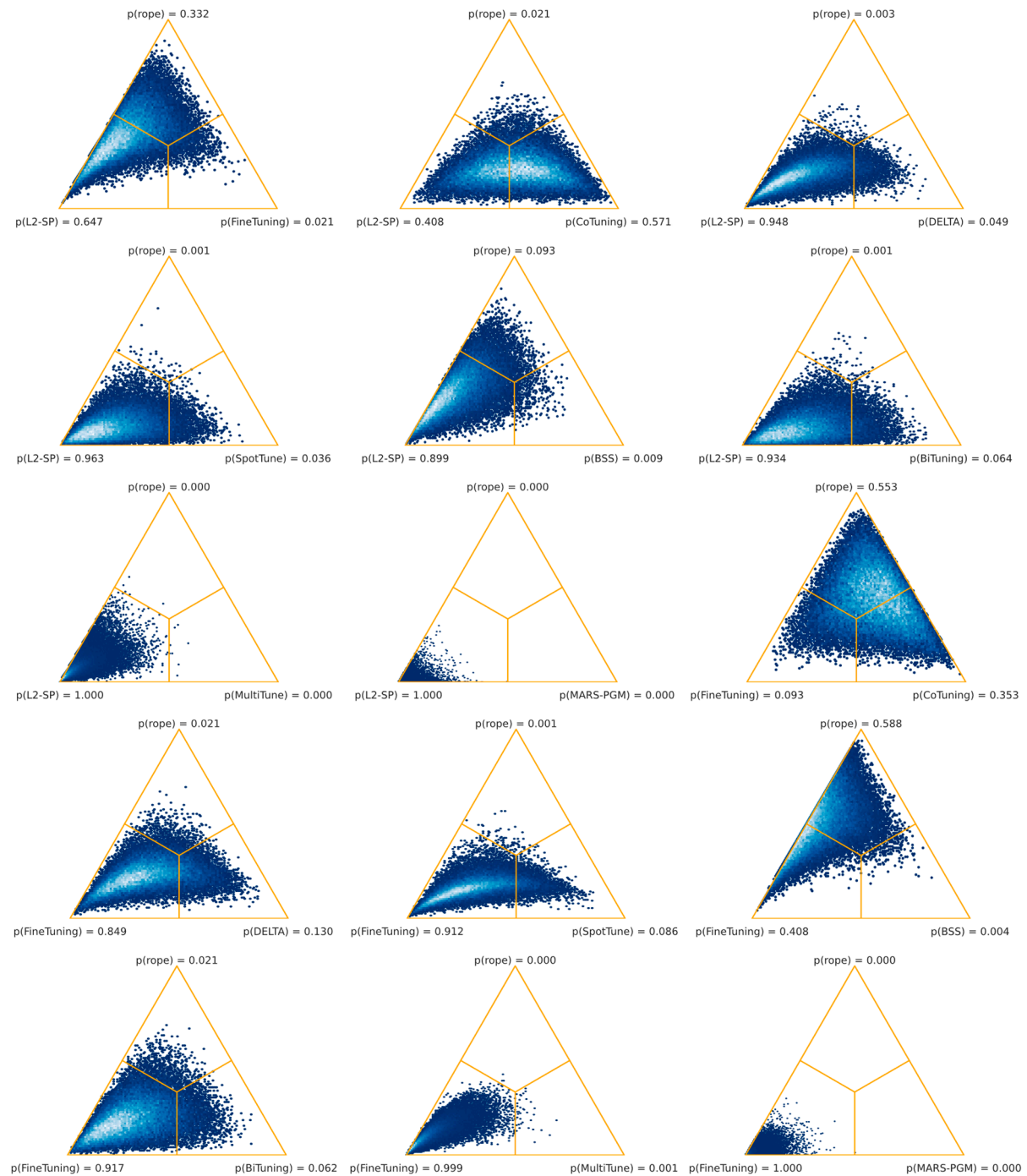


Figure A1. Cont.



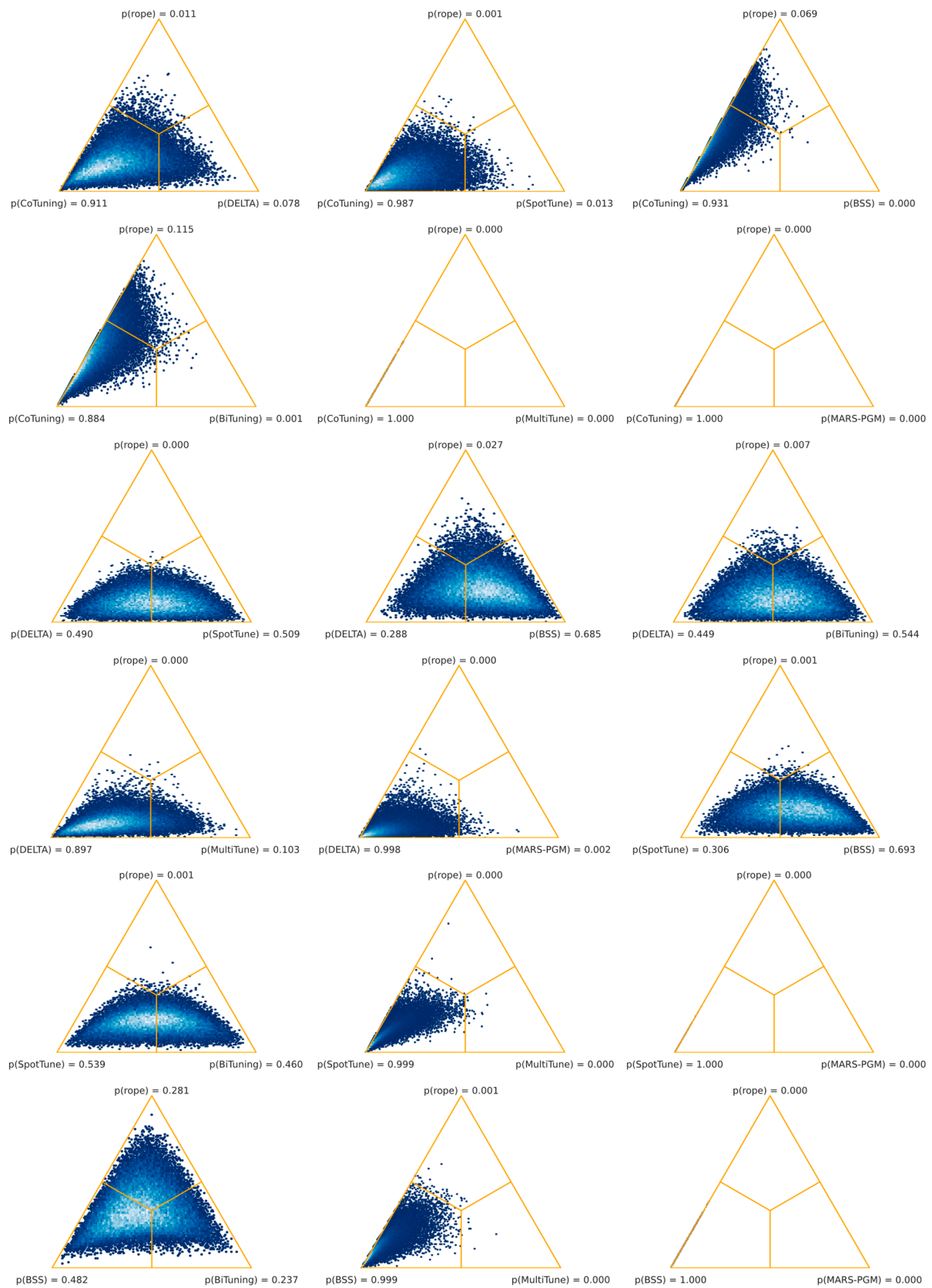
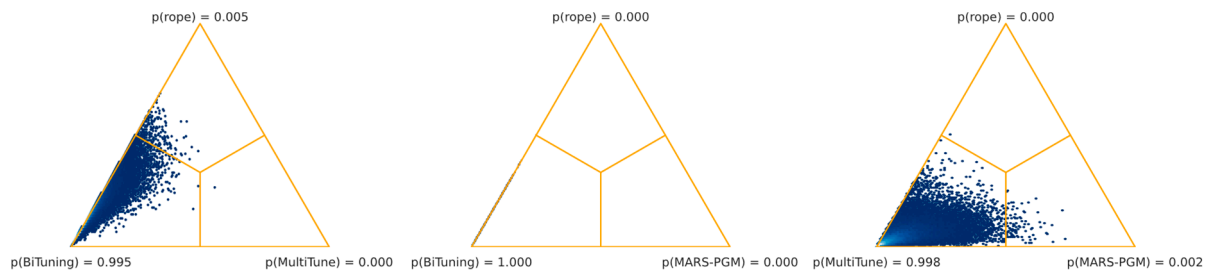


Figure A1. Cont.



**Figure A1.** Simplex plots for the Monte Carlo sampling results of the pair-wise Bayesian signed-rank test.

## Appendix B

**Table A1.** Posterior matrix with probabilities from the pair-wise Bayesian signed-rank test for comparisons across all histopathological tasks.

Algorithm	L <sup>2</sup> -SP			Fine-Tuning			Co-Tuning		
	$P(\ll)$	$P(=)$	$P(\gg)$	$P(\ll)$	$P(=)$	$P(\gg)$	$P(\ll)$	$P(=)$	$P(\gg)$
L <sup>2</sup> -SP	—	—	—	0.647	0.332	0.021	0.408	0.021	0.571
Fine-Tuning	0.021	0.332	0.647	0.647	0.332	0.021	0.093	0.553	0.353
Co-Tuning	0.571	0.021	0.408	—	—	—	—	—	—
DELTA	0.049	0.003	<b>0.948</b>	0.353	0.553	0.093	0.078	0.011	<b>0.911</b>
SpotTune	0.036	0.001	<b>0.963</b>	0.130	0.021	0.849	0.013	0.001	<b>0.987</b>
BSS	0.009	0.093	0.899	0.086	0.001	<b>0.912</b>	0.000	0.069	<b>0.931</b>
Bi-Tuning	0.064	0.001	<b>0.934</b>	0.004	0.588	0.408	0.001	0.115	0.884
MultiTune	0.000	0.000	<b>1.000</b>	0.062	0.021	<b>0.917</b>	0.000	0.000	<b>1.000</b>
MARS-PGM	0.000	0.000	<b>1.000</b>	0.001	0.000	<b>0.999</b>	0.000	0.000	<b>1.000</b>

Algorithm	DELTA			SpotTune			BSS		
	$P(\ll)$	$P(=)$	$P(\gg)$	$P(\ll)$	$P(=)$	$P(\gg)$	$P(\ll)$	$P(=)$	$P(\gg)$
L <sup>2</sup> -SP	<b>0.948</b>	0.003	0.049	<b>0.963</b>	0.001	0.036	0.899	0.093	0.009
Fine-Tuning	0.849	0.021	0.130	<b>0.912</b>	0.001	0.086	0.408	0.588	0.004
Co-Tuning	<b>0.911</b>	0.011	0.078	<b>0.987</b>	0.001	0.013	<b>0.931</b>	0.069	0.000
DELTA	—	—	—	0.490	0.000	0.509	0.288	0.027	0.685
SpotTune	0.509	0.000	0.490	—	—	—	0.306	0.001	0.693
BSS	0.685	0.027	0.288	0.693	0.001	0.306	—	—	—
Bi-Tuning	0.544	0.007	0.449	0.460	0.001	0.539	0.237	0.281	0.482
MultiTune	0.103	0.000	0.897	0.000	0.000	<b>0.999</b>	0.000	0.001	<b>0.999</b>
MARS-PGM	0.002	0.000	<b>0.998</b>	0.000	0.000	<b>1.000</b>	0.000	0.000	<b>1.000</b>

Algorithm	Bi-Tuning			MultiTune			MARS-PGM		
	$P(\ll)$	$P(=)$	$P(\gg)$	$P(\ll)$	$P(=)$	$P(\gg)$	$P(\ll)$	$P(=)$	$P(\gg)$
L <sup>2</sup> -SP	<b>0.934</b>	0.001	0.064	<b>1.000</b>	0.000	0.000	<b>1.000</b>	0.000	0.000
Fine-Tuning	<b>0.917</b>	0.021	0.062	<b>0.999</b>	0.000	0.001	<b>1.000</b>	0.000	0.000
Co-Tuning	0.884	0.115	0.001	<b>1.000</b>	0.000	0.000	<b>1.000</b>	0.000	0.000
DELTA	0.449	0.007	0.544	0.897	0.000	0.103	<b>0.998</b>	0.000	0.002
SpotTune	0.539	0.001	0.460	<b>0.999</b>	0.000	0.000	<b>1.000</b>	0.000	0.000
BSS	0.482	0.281	0.237	<b>0.999</b>	0.001	0.000	<b>1.000</b>	0.000	0.000
Bi-Tuning	—	—	—	<b>0.995</b>	0.005	0.000	<b>1.000</b>	0.000	0.000
MultiTune	0.000	0.005	<b>0.995</b>	—	—	—	<b>0.998</b>	0.000	0.002
MARS-PGM	0.000	0.000	<b>1.000</b>	0.002	0.000	<b>0.998</b>	—	—	—

$P(\ll)$ : probability of negative mean difference;  $P(=)$ : probability of equivalent results;  $P(\gg)$ : probability of positive mean difference. Significant values ( $P(\cdot) \geq 0.9$ ) are highlighted in bold.

# Appendix C

**Table A2.** Posterior matrix with probabilities from the pair-wise Bayesian correlated *t*-test and effect size for comparison with fine-tuning, separated by task.

Task	BSS				Bi-Tuning				Co-Tuning			
	$P(\ll)$	$P(=)$	$P(\gg)$	$d/\gamma$	$P(\ll)$	$P(=)$	$P(\gg)$	$d/\gamma$	$P(\ll)$	$P(=)$	$P(\gg)$	$d/\gamma$
1	0.966	0.013	0.021	−1.171	0.910	0.018	0.073	−1.845	0.466	0.072	0.463	−0.004
2	0.705	0.059	0.236	−0.928	0.865	0.042	0.093	−0.755	0.293	0.044	0.664	0.653
3	0.707	0.110	0.183	−0.378	0.996	0.001	0.003	−1.801	0.172	0.181	0.648	0.222
4	0.979	0.005	0.016	−2.452	0.799	0.045	0.156	−0.955	0.871	0.035	0.094	−1.080
5	0.964	0.019	0.017	−0.780	0.931	0.015	0.054	−1.861	0.976	0.009	0.015	−1.241
6	0.226	0.340	0.434	0.065	0.274	0.258	0.468	0.077	0.490	0.170	0.340	−0.089
7	0.313	0.170	0.517	0.119	0.615	0.092	0.294	−0.324	0.493	0.157	0.351	−0.091
8	0.468	0.085	0.446	0.048	0.360	0.075	0.564	−0.379	0.544	0.067	0.388	0.102
9	0.385	0.126	0.489	0.083	0.004	0.003	<b>0.993</b>	1.407	0.037	0.016	<b>0.947</b>	1.379
10	0.643	0.080	0.277	−0.411	0.739	0.070	0.191	−0.598	0.591	0.110	0.299	−0.250
11	0.408	0.098	0.494	0.087	0.133	0.046	0.822	0.919	0.067	0.023	<b>0.910</b>	1.406
12	0.231	0.076	0.693	0.510	0.337	0.082	0.581	0.284	0.048	0.009	<b>0.942</b>	2.623

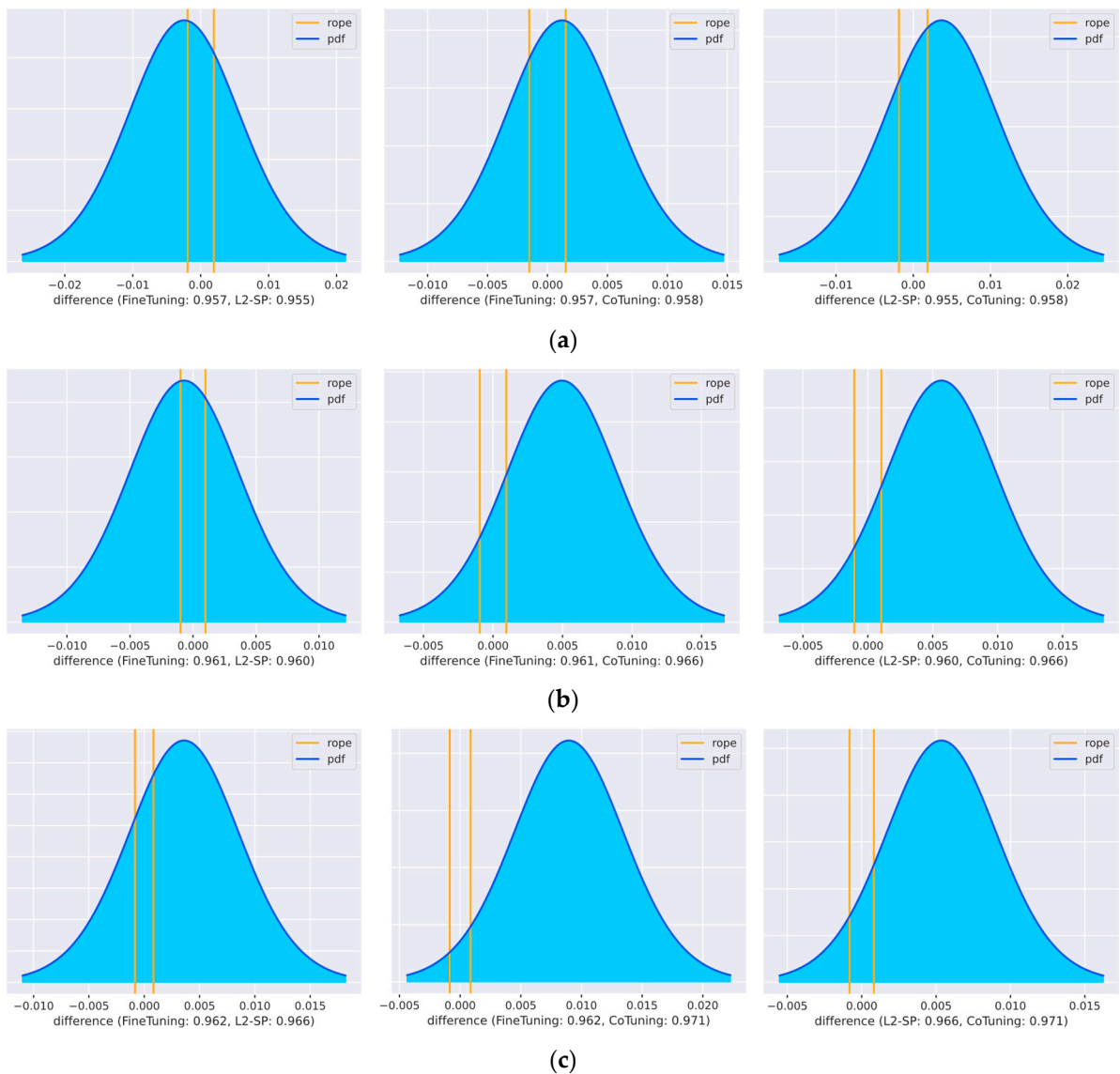
Task	DELTA				L <sup>2</sup> -SP				MARS-PGM			
	$P(\ll)$	$P(=)$	$P(\gg)$	$d/\gamma$	$P(\ll)$	$P(=)$	$P(\gg)$	$d/\gamma$	$P(\ll)$	$P(=)$	$P(\gg)$	$d/\gamma$
1	0.891	0.023	0.087	−1.579	0.515	0.140	0.345	−0.121	0.997	0.000	0.003	−6.447
2	0.984	0.001	0.016	−16.406	0.411	0.043	0.546	0.180	0.951	0.010	0.039	−1.374
3	0.510	0.116	0.374	−0.116	0.051	0.033	<b>0.917</b>	0.879	0.920	0.019	0.061	−1.540
4	0.043	0.020	<b>0.937</b>	1.230	0.089	0.054	0.857	0.707	0.894	0.020	0.085	−1.752
5	0.493	0.209	0.298	−0.094	0.444	0.109	0.447	0.002	0.953	0.010	0.037	−2.102
6	0.599	0.157	0.244	−0.209	0.058	0.052	0.890	0.626	0.840	0.034	0.127	−1.234
7	0.841	0.038	0.121	−1.086	0.668	0.120	0.212	−0.323	0.941	0.013	0.046	−1.908
8	0.081	0.023	0.896	1.692	0.653	0.067	0.279	−0.769	0.821	0.026	0.153	−0.821
9	0.038	0.025	<b>0.937</b>	0.938	0.238	0.138	0.624	0.251	0.943	0.012	0.044	−1.967
10	0.828	0.044	0.128	−0.957	0.783	0.057	0.160	−0.760	0.871	0.029	0.100	−1.332
11	0.038	0.024	<b>0.937</b>	0.986	0.092	0.158	0.750	0.272	0.549	0.086	0.365	−0.209
12	0.669	0.085	0.246	−0.433	0.258	0.100	0.642	0.343	0.177	0.049	0.774	0.878

Task	MultiTune				SpotTune			
	$P(\ll)$	$P(=)$	$P(\gg)$	$d/\gamma$	$P(\ll)$	$P(=)$	$P(\gg)$	$d/\gamma$
1	0.994	0.001	0.005	−4.330	0.991	0.002	0.008	−3.048
2	0.931	0.010	0.059	−1.493	0.849	0.023	0.128	−2.275
3	0.909	0.024	0.067	−1.347	0.962	0.012	0.027	−1.480
4	0.914	0.016	0.069	−1.949	0.739	0.074	0.187	−0.571
5	0.988	0.002	0.010	−2.851	0.820	0.037	0.143	−1.144
6	0.528	0.065	0.406	−0.185	0.106	0.060	0.834	0.682
7	0.961	0.010	0.029	−1.887	0.999	0.000	0.001	−2.932
8	0.705	0.015	0.281	−0.384	0.789	0.048	0.162	−1.139
9	0.924	0.026	0.050	−1.086	0.461	0.074	0.465	0.006
10	0.811	0.037	0.152	−1.144	0.471	0.128	0.401	−0.056
11	0.264	0.062	0.673	0.573	0.045	0.042	<b>0.912</b>	0.670
12	0.163	0.062	0.775	0.690	0.012	0.005	<b>0.983</b>	1.804

$P(\ll)$ : probability of negative mean difference;  $P(=)$ : probability of equivalent results;  $P(\gg)$ : probability of positive mean difference;  $d$ : Cohen's *d*;  $\gamma$ : Akinshin's gamma. Significant positive mean differences ( $P(\cdot) \geq 0.9$ ) are highlighted in bold.

# Appendix D



**Figure A2.** Posterior distributions for the pair-wise comparison of fine-tuning,  $L^2$ -SP, and Co-Tuning using the AUC from the Bayesian correlated  $t$ -test. (a) Dataset size “Base” ( $\times 1$ ). (b) Dataset size “Large” ( $\times 2.5$ ). (c) Dataset size “XLarge” ( $\times 10$ ).

# Appendix E

**Table A3.** Posterior matrix for the dataset size “Base” with probabilities from the pair-wise Bayesian correlated  $t$ -test and effect size.

Algorithm	Fine-Tuning				L <sup>2</sup> -SP				Co-Tuning			
	$P(\ll)$	$P(=)$	$P(\gg)$	$d$	$P(\ll)$	$P(=)$	$P(\gg)$	$d$	$P(\ll)$	$P(=)$	$P(\gg)$	$d$
Fine-Tuning	—	—	—	—	0.525	0.173	0.302	−0.127	0.286	0.241	0.473	0.079
L <sup>2</sup> -SP	0.302	0.173	0.525	0.127	—	—	—	—	0.231	0.173	0.596	0.197
Co-Tuning	0.473	0.241	0.286	−0.079	0.596	0.173	0.231	−0.197	—	—	—	—

$P(\ll)$ : probability of negative mean difference;  $P(=)$ : probability of equivalent results;  $P(\gg)$ : probability of positive mean difference;  $d$ : Cohen’s  $d$ . Significant values ( $P(\cdot) \geq 0.9$ ) are highlighted in bold.

**Table A4.** Posterior matrix for the dataset size “Large” with probabilities from the pair-wise Bayesian correlated *t*-test and effect size.

Algorithm	Fine-Tuning				L <sup>2</sup> -SP				Co-Tuning			
	<i>P</i> (<<)	<i>P</i> (=)	<i>P</i> (>>)	<i>d</i>	<i>P</i> (<<)	<i>P</i> (=)	<i>P</i> (>>)	<i>d</i>	<i>P</i> (<<)	<i>P</i> (=)	<i>P</i> (>>)	<i>d</i>
Fine-Tuning	—	—	—	—	0.476	0.170	0.355	−0.072	0.082	0.088	0.830	0.518
L <sup>2</sup> -SP	0.355	0.170	0.476	0.072	—	—	—	—	0.070	0.081	0.849	0.544
Co-Tuning	0.830	0.088	0.082	−0.518	0.849	0.081	0.070	−0.544	—	—	—	—

*P*(<<): probability of negative mean difference; *P*(=): probability of equivalent results; *P*(>>): probability of positive mean difference; *d*: Cohen’s *d*. Significant values (*P*(·) ≥ 0.9) are highlighted in bold.

**Table A5.** Posterior matrix for the dataset size “XLarge” with probabilities from the pair-wise Bayesian correlated *t*-test and effect size.

Algorithm	Fine-Tuning				L <sup>2</sup> -SP				Co-Tuning			
	<i>P</i> (<<)	<i>P</i> (=)	<i>P</i> (>>)	$\gamma$	<i>P</i> (<<)	<i>P</i> (=)	<i>P</i> (>>)	$\gamma$	<i>P</i> (<<)	<i>P</i> (=)	<i>P</i> (>>)	$\gamma$
Fine-Tuning	—	—	—	—	0.198	0.099	0.703	0.164	0.024	0.025	<b>0.951</b>	0.748
L <sup>2</sup> -SP	0.703	0.099	0.198	−0.164	—	—	—	—	0.061	0.063	0.876	0.619
Co-Tuning	<b>0.951</b>	0.025	0.024	−0.748	0.876	0.063	0.061	−0.619	—	—	—	—

*P*(<<): probability of negative mean difference; *P*(=): probability of equivalent results; *P*(>>): probability of positive mean difference;  $\gamma$ : Akinshin’s gamma. Significant values (*P*(·) ≥ 0.9) are highlighted in bold.

## References

1. Acs, B.; Rantalainen, M.; Hartman, J. Artificial intelligence as the next step towards precision pathology. *J. Intern. Med.* **2020**, *288*, 62–81. [\[CrossRef\]](#) [\[PubMed\]](#)
2. Banerji, S.; Mitra, S. Deep learning in histopathology: A review. *WIREs Data Min. Knowl. Discov.* **2022**, *12*, e1439. [\[CrossRef\]](#)
3. Morales, S.; Engan, K.; Naranjo, V. Artificial intelligence in computational pathology—Challenges and future directions. *Digit. Signal Process.* **2021**, *119*, 103196. [\[CrossRef\]](#)
4. Echle, A.; Rindtorff, N.T.; Brinker, T.J.; Luedde, T.; Pearson, A.T.; Kather, J.N. Deep learning in cancer pathology: A new generation of clinical biomarkers. *Br. J. Cancer* **2021**, *124*, 686–696. [\[CrossRef\]](#) [\[PubMed\]](#)
5. Cui, M.; Zhang, D.Y. Artificial intelligence and computational pathology. *Lab. Investig.* **2021**, *101*, 412–422. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Srinidhi, C.L.; Ciga, O.; Martel, A.L. Deep neural network models for computational histopathology: A survey. *Med. Image Anal.* **2021**, *67*, 101813. [\[CrossRef\]](#) [\[PubMed\]](#)
7. Yosinski, J.; Clune, J.; Bengio, Y.; Lipson, H. How transferable are features in deep neural networks? In Proceedings of the NIPS’14, Montréal, QC, Canada, 8–13 December 2014; pp. 3320–3328.
8. Kornblith, S.; Shlens, J.; Le, Q.V. Do Better ImageNet Models Transfer Better? In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 2656–2666. [\[CrossRef\]](#)
9. He, K.; Girshick, R.; Dollar, P. Rethinking ImageNet Pre-Training. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 4918–4927. [\[CrossRef\]](#)
10. Li, X.; Plataniotis, K.N. How much off-the-shelf knowledge is transferable from natural images to pathology images? *PLoS ONE* **2020**, *15*, e0240530. [\[CrossRef\]](#)
11. Mormont, R.; Geurts, P.; Maree, R. Comparison of Deep Transfer Learning Strategies for Digital Pathology. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, USA, 18–22 June 2018; pp. 2343–2352. [\[CrossRef\]](#)
12. Azizpour, H.; Razavian, A.S.; Sullivan, J.; Maki, A.; Carlsson, S. Factors of Transferability for a Generic ConvNet Representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 1790–1802. [\[CrossRef\]](#)
13. Abnar, S.; Dehghani, M.; Neyshabur, B.; Sedghi, H. Exploring the Limits of Large Scale Pre-training. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022.
14. Mahajan, D.; Girshick, R.; Ramanathan, V.; He, K.; Paluri, M.; Li, Y.; Bharambe, A.; van der Maaten, L. Exploring the Limits of Weakly Supervised Pretraining. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 185–201. [\[CrossRef\]](#)
15. Wan, R.; Xiong, H.; Li, X.; Zhu, Z.; Huan, J. Towards Making Deep Transfer Learning Never Hurt. In Proceedings of the 2019 IEEE International Conference on Data Mining (ICDM), Beijing, China, 8–11 November 2019; pp. 578–587. [\[CrossRef\]](#)
16. Plested, J.; Shen, X.; Gedeon, T. Rethinking Binary Hyperparameters for Deep Transfer Learning. In Proceedings of the Neural Information Processing, Sanur, Bali, Indonesia, 8–12 December 2021; pp. 463–475. [\[CrossRef\]](#)
17. Li, H.; Chaudhari, P.; Yang, H.; Lam, M.; Ravichandran, A.; Bhotika, R.; Soatto, S. Rethinking the Hyperparameters for Fine-tuning. In Proceedings of the International Conference on Learning Representations, Virtual, 26 April–1 May 2020.



18. Raghu, M.; Zhang, C.; Kleinberg, J.; Bengio, S. Transfusion: Understanding Transfer Learning for Medical Imaging. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 3347–3357.
19. Plested, J. Beyond Binary Hyperparameters in Deep Transfer Learning for Image Classification. Ph.D. Thesis, The Australian National University, Canberra, Australia, 2023.
20. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587. [\[CrossRef\]](#)
21. Oquab, M.; Bottou, L.; Laptev, I.; Sivic, J. Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1717–1724. [\[CrossRef\]](#)
22. You, K.; Kou, Z.; Long, M.; Wang, J. Co-Tuning for Transfer Learning. In Proceedings of the 34th International Conference on Neural Information Processing Systems, Virtual, 6–12 December 2020; pp. 17236–17246.
23. Guo, Y.; Shi, H.; Kumar, A.; Grauman, K.; Rosing, T.; Feris, R. SpotTune: Transfer Learning Through Adaptive Fine-Tuning. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, 16–20 June 2019; pp. 4805–4814. [\[CrossRef\]](#)
24. Li, X.; Grandvalet, Y.; Davoine, F. Explicit Inductive Bias for Transfer Learning with Convolutional Networks. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 2825–2834.
25. Liao, W.; Xiong, H.; Wang, Q.; Mo, Y.; Li, X.; Liu, Y.; Chen, Z.; Huang, S.; Dou, D. MUSCLE: Multi-task Self-supervised Continual Learning to Pre-train Deep Models for X-Ray Images of Multiple Body Parts. In Proceedings of the Medical Image Computing and Computer Assisted Intervention—MICCAI 2022, Singapore, 18–22 September 2022; pp. 151–161. [\[CrossRef\]](#)
26. Sagie, N.; Greenspan, H.; Goldberger, J. Transfer Learning via Parameter Regularization for Medical Image Segmentation. In Proceedings of the 2021 29th European Signal Processing Conference (EUSIPCO), Virtual, 23–27 August 2021; pp. 985–989. [\[CrossRef\]](#)
27. Su, K.; Zhou, E.; Sun, X.; Wang, C.; Yu, D.; Luo, X. Pre-trained StyleGAN Based Data Augmentation for Small Sample Brain CT Motion Artifacts Detection. In Proceedings of the Advanced Data Mining and Applications: 16th International Conference, ADMA 2020, Foshan, China, 12–14 November 2020; pp. 339–346. [\[CrossRef\]](#)
28. An, R.; Han, T.; Wang, Y.; Ai, D.; Wang, Y.; Yang, J. Cross-Domain Transfer Learning for Vessel Segmentation in Computed Tomographic Coronary Angiographic Images. In Proceedings of the Image and Graphics: 11th International Conference, ICIG 2021, Haikou, China, 26–28 December 2021; pp. 571–583. [\[CrossRef\]](#)
29. Nguyen, T.; Pernkopf, F. Lung Sound Classification Using Co-Tuning and Stochastic Normalization. *IEEE Trans. Biomed. Eng.* **2022**, *69*, 2872–2882. [\[CrossRef\]](#)
30. Jiang, J.; Shu, Y.; Wang, J.; Long, M. Transferability in Deep Learning: A Survey. *arXiv* **2022**, arXiv:2201.05867.
31. Sauter, D.; Lodde, G.; Nensa, F.; Schadendorf, D.; Livingstone, E.; Kukuk, M. Deep learning in computational dermatopathology of melanoma: A technical systematic literature review. *Comput. Biol. Med.* **2023**, *163*, 107083. [\[CrossRef\]](#) [\[PubMed\]](#)
32. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1997**, *1*, 67–82. [\[CrossRef\]](#)
33. Zhang, W.; Deng, L.; Zhang, L.; Wu, D. A Survey on Negative Transfer. *IEEE/CAA J. Autom. Sin.* **2023**, *10*, 305–329. [\[CrossRef\]](#)
34. Sharma, Y.; Ehsan, L.; Syed, S.; Brown, D.E. HistoTransfer: Understanding Transfer Learning for Histopathology. In Proceedings of the 2021 IEEE EMBS International Conference on Biomedical and Health Informatics (BHI), Athens, Greece, 27–30 July 2021; pp. 1–4. [\[CrossRef\]](#)
35. Kim, Y.-G.; Kim, S.; Cho, C.E.; Song, I.H.; Lee, H.J.; Ahn, S.; Park, S.Y.; Gong, G.; Kim, N. Effectiveness of transfer learning for enhancing tumor classification with a convolutional neural network on frozen sections. *Sci. Rep.* **2020**, *10*, 21899. [\[CrossRef\]](#) [\[PubMed\]](#)
36. Tellez, D.; Litjens, G.; Bándi, P.; Bulten, W.; Bokhorst, J.-M.; Ciompi, F.; van der Laak, J. Quantifying the effects of data augmentation and stain color normalization in convolutional neural networks for computational pathology. *Med. Image Anal.* **2019**, *58*, 101544. [\[CrossRef\]](#) [\[PubMed\]](#)
37. Pan, S.J.; Yang, Q. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [\[CrossRef\]](#)
38. McCloskey, M.; Cohen, N.J. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In *Psychology of Learning and Motivation*; Bower, G.H., Ed.; Academic Press: Cambridge, MA, USA, 1989; pp. 109–165.
39. French, R.M. Catastrophic forgetting in connectionist networks. *Trends Cogn. Sci.* **1999**, *3*, 128–135. [\[CrossRef\]](#) [\[PubMed\]](#)
40. Rosenstein, M.T.; Marx, Z.; Kaelbling, L.P.; Dietterich, T.G. To Transfer or Not to Transfer. In Proceedings of the NIPS 2005 Workshop on Transfer Learning, Vancouver, BC, Canada, 9 December 2005.
41. Housby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; de Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; Gelly, S. Parameter-Efficient Transfer Learning for NLP. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 2790–2799.
42. Sung, Y.-L.; Cho, J.; Bansal, M. VL-Adapter: Parameter-Efficient Transfer Learning for Vision-and-Language Tasks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 19–24 June 2022; pp. 5227–5237. [\[CrossRef\]](#)

43. Zhou, K.; Liu, Z.; Qiao, Y.; Xiang, T.; Loy, C.C. Domain Generalization: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 4396–4415. [\[CrossRef\]](#)
44. Muandet, K.; Balduzzi, D.; Schölkopf, B. Domain Generalization via Invariant Feature Representation. In Proceedings of the 30th International Conference on Machine Learning, Atlanta, GA, USA, 17–19 June 2013; pp. 10–18.
45. Wang, Y.; Yao, Q.; Kwok, J.T.; Ni, L.M. Generalizing from a Few Examples: A Survey on Few-Shot Learning. *ACM Comput. Surv.* **2020**, *53*, 1–34. [\[CrossRef\]](#)
46. Pourpanah, F.; Abdar, M.; Luo, Y.; Zhou, X.; Wang, R.; Lim, C.P.; Wang, X.-Z.; Wu, Q.M.J. A Review of Generalized Zero-Shot Learning Methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 4051–4070. [\[CrossRef\]](#)
47. Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; Neubig, G. Pre-Train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv.* **2023**, *55*, 1–35. [\[CrossRef\]](#)
48. Shu, Y.; Kou, Z.; Cao, Z.; Wang, J.; Long, M. Zoo-Tuning: Adaptive Transfer from A Zoo of Models. In Proceedings of the 38th International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 9626–9637.
49. Nguyen, C.; Hassner, T.; Seeger, M.; Archambeau, C. LEEP: A New Measure to Evaluate Transferability of Learned Representations. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 7294–7305.
50. Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A.A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 3521–3526. [\[CrossRef\]](#) [\[PubMed\]](#)
51. Ding, K.; He, Y.; Dong, X.; Yang, J.; Zhang, L.; Li, A.; Zhang, X.; Mo, L. GFlow-FT: Pick a Child Network via Gradient Flow for Efficient Fine-Tuning in Recommendation Systems. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, 17–21 October 2022; pp. 3918–3922. [\[CrossRef\]](#)
52. Gouk, H.; Hospedales, T.M.; Pontil, M. Distance-Based Regularisation of Deep Networks for Fine-Tuning. In Proceedings of the International Conference on Learning Representations, Virtual, 3–7 May 2021. [\[CrossRef\]](#)
53. Li, X.; Xiong, H.; Wang, H.; Rao, Y.; Liu, L.; Chen, Z.; Huan, J. DELTA: DEep Learning Transfer using Feature Map with Attention for Convolutional Networks. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019. [\[CrossRef\]](#)
54. Zhong, J.; Wang, X.; Kou, Z.; Wang, J.; Long, M. Bi-tuning: Efficient Transfer from Pre-trained Models. In Proceedings of the European Conference on Machine Learning and Principles and PKDD, Torino, Italy, 18–22 September 2023; pp. 357–373. [\[CrossRef\]](#)
55. Chen, X.; Wang, S.; Fu, B.; Long, M.; Wang, J. Catastrophic Forgetting Meets Negative Transfer: Batch Spectral Shrinkage for Safe Transfer Learning. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 1908–1918.
56. Kou, Z.; You, K.; Long, M.; Wang, J. Stochastic Normalization. In Proceedings of the 34th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 6–12 December 2020; pp. 16304–16314.
57. Guo, Y.; Li, Y.; Wang, L.; Rosing, T. AdaFilter: Adaptive Filter Fine-Tuning for Deep Transfer Learning. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 4060–4066.
58. Vrbančič, G.; Podgorelec, V. Transfer Learning with Adaptive Fine-Tuning. *IEEE Access* **2020**, *8*, 196197–196211. [\[CrossRef\]](#)
59. Wanjiku, R.N.; Nderu, L.; Kimwele, M. Dynamic fine-tuning layer selection using Kullback–Leibler divergence. *Eng. Rep.* **2023**, *5*, e12595. [\[CrossRef\]](#)
60. Royer, A.; Lampert, C. A Flexible Selection Scheme for Minimum-Effort Transfer Learning. In Proceedings of the 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), Snowmass, CO, USA, 1–5 March 2020; pp. 2180–2189. [\[CrossRef\]](#)
61. Wang, Y.; Plested, J.; Gedeon, T. MultiTune: Adaptive Integration of Multiple Fine-Tuning Models for Image Classification. In Proceedings of the 27th International Conference, ICONIP 2020, Bangkok, Thailand, 18–22 November 2020; pp. 488–496. [\[CrossRef\]](#)
62. Zhang, Y.; Zhang, Y.; Yang, Q. Parameter Transfer Unit for Deep Neural Networks. In Proceedings of the Advances in Knowledge Discovery and Data Mining: 23rd Pacific-Asia Conference, PAKDD 2019, Macau, China, 14–17 April 2019; pp. 82–95. [\[CrossRef\]](#)
63. Nagae, S.; Kawai, S.; Nobuhara, H. Transfer Learning Layer Selection Using Genetic Algorithm. In Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC), Glasgow, UK, 19–24 July 2020; pp. 1–6. [\[CrossRef\]](#)
64. Imai, S.; Kawai, S.; Nobuhara, H. Stepwise PathNet: A layer-by-layer knowledge-selection-based transfer learning algorithm. *Sci. Rep.* **2020**, *10*, 8132. [\[CrossRef\]](#) [\[PubMed\]](#)
65. Li, Z.; Hoiem, D. Learning without Forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 2935–2947. [\[CrossRef\]](#) [\[PubMed\]](#)
66. Ge, W.; Yu, Y. Borrowing Treasures from the Wealthy: Deep Transfer Learning Through Selective Joint Fine-Tuning. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 10–19. [\[CrossRef\]](#)
67. Mehta, R. Sparse Transfer Learning via Winning Lottery Tickets. In Proceedings of the Workshop on Learning Transferable Skills (NeurIPS 2019), Vancouver, BC, Canada, 13–14 December 2019. [\[CrossRef\]](#)
68. van Soelen, R.; Sheppard, J.W. Using Winning Lottery Tickets in Transfer Learning for Convolutional Neural Networks. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; pp. 1–8. [\[CrossRef\]](#)

69. Veit, A.; Wilber, M.J.; Belongie, S. Residual Networks Behave Like Ensembles of Relatively Shallow Networks. In Proceedings of the 30th International Conference on Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 550–558.
70. Frankle, J.; Carbin, M. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In Proceedings of the ICLR 2019: International Conference for Learning Representations, New Orleans, LA, USA, 6–9 May 2019. [CrossRef]
71. Benavoli, A.; Corani, G.; Demšar, J.; Zaffalon, M. Time for a Change: A Tutorial for Comparing Multiple Classifiers Through Bayesian Analysis. *J. Mach. Learn. Res.* **2017**, *18*, 1–36.
72. Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.
73. Aksac, A.; Demetrick, D.J.; Ozyer, T.; Alhajj, R. BreCaHAD: A dataset for breast cancer histopathological annotation and diagnosis. *BMC Res. Notes* **2019**, *12*, 82. [CrossRef]
74. Veta, M.; Heng, Y.J.; Stathonikos, N.; Bejnordi, B.E.; Beca, F.; Wollmann, T.; Rohr, K.; Shah, M.A.; Wang, D.; Rousson, M.; et al. Predicting breast tumor proliferation from whole-slide images: The TUPAC16 challenge. *Med. Image Anal.* **2019**, *54*, 111–121. [CrossRef]
75. Litjens, G.; Bandi, P.; Ehteshami Bejnordi, B.; Geessink, O.; Balkenhol, M.; Bult, P.; Halilovic, A.; Hermesen, M.; van de Loo, R.; Vogels, R.; et al. 1399 H&E-stained sentinel lymph node sections of breast cancer patients: The CAMELYON dataset. *GigaScience* **2018**, *7*, giy065. [CrossRef] [PubMed]
76. Kaczmarzyk, J.R.; Abousamra, S.; Kurc, T.; Gupta, R.; Saltz, J. Dataset for Tumor Infiltrating Lymphocyte Classification (304,097 Image Patches from TCGA), 2022. Available online: <https://zenodo.org/records/6604094> (accessed on 28 June 2023).
77. Abousamra, S.; Gupta, R.; Hou, L.; Batiste, R.; Zhao, T.; Shankar, A.; Rao, A.; Chen, C.; Samaras, D.; Kurc, T.; et al. Deep Learning-Based Mapping of Tumor Infiltrating Lymphocytes in Whole Slide Images of 23 Types of Cancer. *Front. Oncol.* **2022**, *11*, 806603. [CrossRef] [PubMed]
78. Saltz, J.; Gupta, R.; Hou, L.; Kurc, T.; Singh, P.; Nguyen, V.; Samaras, D.; Shroyer, K.R.; Zhao, T.; Batiste, R.; et al. Spatial Organization and Molecular Correlation of Tumor-Infiltrating Lymphocytes Using Deep Learning on Pathology Images. *Cell Rep.* **2018**, *23*, 181–193. [CrossRef] [PubMed]
79. Kather, J.N.; Weis, C.-A.; Bianconi, F.; Melchers, S.M.; Schad, L.R.; Gaiser, T.; Marx, A.; Zöllner, F.G. Multi-class texture analysis in colorectal cancer histology. *Sci. Rep.* **2016**, *6*, 27988. [CrossRef] [PubMed]
80. Stadler, C.B.; Lindvall, M.; Lundström, C.; Bodén, A.; Lindman, K.; Rose, J.; Treanor, D.; Blomma, J.; Stacke, K.; Pinchaud, N.; et al. Proactive Construction of an Annotated Imaging Database for Artificial Intelligence Training. *J. Digit. Imaging* **2021**, *34*, 105–115. [CrossRef] [PubMed]
81. Thomas, S.M.; Lefevre, J.G.; Baxter, G.; Hamilton, N.A. Non-melanoma skin cancer segmentation for histopathology dataset. *Data Brief* **2021**, *39*, 107587. [CrossRef] [PubMed]
82. Varoquaux, G.; Cheplygina, V. Machine learning for medical imaging: Methodological failures and recommendations for the future. *NPJ Digit. Med.* **2022**, *5*, 48. [CrossRef]
83. Choi, W.-S. Problems and alternatives of testing significance using null hypothesis and P-value in food research. *Food Sci. Biotechnol.* **2023**, *32*, 1479–1487. [CrossRef]
84. Cohen, J. *Statistical Power Analysis for the Behavioral Sciences*, 2nd ed.; Erlbaum: Hillsdale, NJ, USA, 1988; ISBN 0805802835.
85. Herbold, S. Autorank: A Python package for automated ranking of classifiers. *J. Open Source Softw.* **2020**, *5*, 2173. [CrossRef]
86. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems 25 (NIPS 2012), Lake Tahoe, NV, USA, 3–8 December 2012; pp. 1097–1105.
87. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015.
88. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778. [CrossRef]
89. Chollet, F. Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1800–1807. [CrossRef]
90. Tan, M.; Le, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019; pp. 6105–6114.
91. Jaegle, A.; Borgeaud, S.; Alayrac, J.-B.; Doersch, C.; Ionescu, C.; Ding, D.; Koppula, S.; Zoran, D.; Brock, A.; Shelhamer, E.; et al. Perceiver IO: A General Architecture for Structured Inputs & Outputs. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022.
92. Jaegle, A.; Gimeno, F.; Brock, A.; Zisserman, A.; Vinyals, O.; Carreira, J. Perceiver: General Perception with Iterative Attention. In Proceedings of the 38th International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 4651–4664.
93. Kolesnikov, A.; Dosovitskiy, A.; Weissenborn, D.; Heigold, G.; Uszkoreit, J.; Beyer, L.; Minderer, M.; Dehghani, M.; Houlsby, N.; Gelly, S.; et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In Proceedings of the 9th International Conference on Learning Representations (ICLR), Virtual, 3–7 May 2021.
94. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All you Need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.

95. Liu, Z.; Mao, H.; Wu, C.-Y.; Feichtenhofer, C.; Darrell, T.; Xie, S. A ConvNet for the 2020s. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 11976–11986. [\[CrossRef\]](#)
96. Liu, S.; Chen, T.; Chen, X.; Chen, X.; Xiao, Q.; Wu, B.; Kärkkäinen, T.; Pechenizkiy, M.; Mocanu, D.C.; Wang, Z. More ConvNets in the 2020s: Scaling up Kernels Beyond 51x51 using Sparsity. In Proceedings of the The Eleventh International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
97. Ding, X.; Zhang, X.; Han, J.; Ding, G. Scaling Up Your Kernels to  $31 \times 31$ : Revisiting Large Kernel Design in CNNs. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 11953–11965. [\[CrossRef\]](#)
98. Chen, Y.; Liu, J.; Zhang, X.; Qi, X.; Jia, J. LargeKernel3D: Scaling up Kernels in 3D Sparse CNNs. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 13488–13498. [\[CrossRef\]](#)
99. Woo, S.; Debnath, S.; Hu, R.; Chen, X.; Liu, Z.; Kweon, I.S.; Xie, S. ConvNeXt V2: Co-designing and Scaling ConvNets with Masked Autoencoders. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 16133–16142. [\[CrossRef\]](#)
100. Geirhos, R.; Rubisch, P.; Michaelis, C.; Bethge, M.; Wichmann, F.A.; Brendel, W. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
101. PyTorch Foundation. Convnext\_Tiny. Available online: [https://pytorch.org/vision/main/models/generated/torchvision.models.convnext\\_tiny.html](https://pytorch.org/vision/main/models/generated/torchvision.models.convnext_tiny.html) (accessed on 23 May 2023).
102. Gupta, A.; Ramanath, R.; Shi, J.; Keerthi, S.S. Adam vs. SGD: Closing the generalization gap on image classification. In Proceedings of the OPT2021: 13th Annual Workshop on Optimization for Machine Learning, Virtual, 13–14 December 2021.
103. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 11–17 October 2021; pp. 9992–10002.
104. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520. [\[CrossRef\]](#)
105. Hendrycks, D.; Gimpel, K. Gaussian Error Linear Units (GELUs). *arXiv* **2016**, arXiv:1606.08415.
106. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer Normalization. *arXiv* **2016**, arXiv:1607.06450.
107. Jiang, L.; Yuan, B.; Ma, W.; Wang, Y. JujubeNet: A high-precision lightweight jujube surface defect classification network with an attention mechanism. *Front. Plant Sci.* **2023**, *13*, 1108437. [\[CrossRef\]](#) [\[PubMed\]](#)
108. Li, X.; Grandvalet, Y.; Davoine, F.; Cheng, J.; Cui, Y.; Zhang, H.; Belongie, S.; Tsai, Y.-H.; Yang, M.-H. Transfer learning in computer vision tasks: Remember where you come from. *Image Vis Comput* **2020**, *93*, 103853. [\[CrossRef\]](#)
109. Poehlmann, A.; Villalba, S. *TiffSlide—Cloud Native Openslide-Python Replacement*; Zenodo: Leverkusen, Germany, 2022.
110. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*; Curran Associates, Inc.: Red Hoo, NY, USA, 2019; pp. 8024–8035.
111. TensorFlow Developers. *TensorFlow*; Zenodo: Mountain View, CA, USA, 2023.
112. Buslaev, A.; Iglovikov, V.I.; Khvedchenya, E.; Parinov, A.; Druzhinin, M.; Kalinin, A.A. Albumentations: Fast and Flexible Image Augmentations. *Information* **2020**, *11*, 125. [\[CrossRef\]](#)
113. Demšar, J. Baycomp. Available online: <https://pypi.org/project/baycomp/> (accessed on 21 September 2023).
114. Kruschke, J.K.; Liddell, T.M. The Bayesian New Statistics: Hypothesis testing, estimation, meta-analysis, and power analysis from a Bayesian perspective. *Psychon. Bull. Rev.* **2018**, *25*, 178–206. [\[CrossRef\]](#)
115. Tuli, S.; Dasgupta, I.; Grant, E.; Griffiths, T.L. Are Convolutional Neural Networks or Transformers more like human vision? In Proceedings of the 43rd Annual Meeting of the Cognitive Science Society, Vienna, Austria, 26–29 July 2021.
116. Ray, I.; Raipuria, G.; Singhal, N. Rethinking ImageNet Pre-training for Computational Histopathology. In Proceedings of the 2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), Glasgow, UK, 11–15 July 2022; pp. 3059–3062. [\[CrossRef\]](#)
117. Howlader, K.; Liu, L. Transfer Learning Pre-training Dataset and Fine-tuning Effect Analysis on Cancer Histopathology Images. In Proceedings of the 2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Las Vegas, NV, USA, 6–8 December 2022; pp. 3015–3022. [\[CrossRef\]](#)
118. McBee, P.; Moradinasab, N.; Brown, D.E.; Syed, S. Pre-training Segmentation Models for Histopathology. In Proceedings of the Medical Imaging with Deep Learning, Short Paper Track, Nashville, TN, USA, 10–12 June 2023.

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.