

Article

Pretraining Convolutional Neural Networks for Mudstone Petrographic Thin-Section Image Classification

Rafael Pires de Lima ^{1,*}  and David Duarte ²¹ Geological Survey of Brazil, São Paulo 01304-010, Brazil² School of Geosciences, University of Oklahoma, Norman, OK 73019, USA; dduarte@ou.edu

* Correspondence: rafael.lima@cprm.gov.br

Abstract: Convolutional neural networks (CNN) are currently the most widely used tool for the classification of images, especially if such images have large within- and small between- group variance. Thus, one of the main factors driving the development of CNN models is the creation of large, labelled computer vision datasets, some containing millions of images. Thanks to transfer learning, a technique that modifies a model trained on a primary task to execute a secondary task, the adaptation of CNN models trained on such large datasets has rapidly gained popularity in many fields of science, geosciences included. However, the trade-off between two main components of the transfer learning methodology for geoscience images is still unclear: the difference between the datasets used in the primary and secondary tasks; and the amount of available data for the primary task itself. We evaluate the performance of CNN models pretrained with different types of image datasets—specifically, dermatology, histology, and raw food—that are fine-tuned to the task of petrographic thin-section image classification. Results show that CNN models pretrained on ImageNet achieve higher accuracy due to the larger number of samples, as well as a larger variability in the samples in ImageNet compared to the other datasets evaluated.

Keywords: transfer learning; convolutional neural networks; petrography; thin-section images



Citation: Pires de Lima, R.; Duarte, D. Pretraining Convolutional Neural Networks for Mudstone Petrographic Thin-Section Image Classification. *Geosciences* **2021**, *11*, 336. <https://doi.org/10.3390/geosciences11080336>

Academic Editors: Tadeusz Marek Peryt, Eun Young Lee, Annarita D'Addabbo, Dimitrios Piretzidis and Jesus Martinez-Frias

Received: 5 July 2021

Accepted: 5 August 2021

Published: 11 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 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

Although the roots of convolutional neural networks (CNN) emerged in the 1980s [1,2], they were only widely adopted in the 2010s, after a model used by Krizhevsky et al. [3] won the 2012 ImageNet competition challenge [4] by a large margin [5] when competing against traditional machine-learning algorithms. ImageNet [6] is one of several computer vision datasets (e.g., [7–9]) that contributed to the development of CNN models, as well as the standardization of models' analysis. CNN models used for the classification of images are trained on datasets containing pairs of input data (images) and labels (classes). During training, CNN models need to learn mapping from the input data to the desired labels. Compared to fully connected (fc) layers, convolutional layers are neurons that better exploit the locality, stationarity, and compositionality of signals that are well suited to image data. It is generally useful to have a large dataset for training CNNs for two main reasons: image data have a very high dimensionality, e.g., a red-green-blue (RGB) image with 224 × 224 pixels is one sample in 150,528 dimensions space; and CNN models are usually built with very large number of parameters (in the order of millions). The widespread adoption of training and evaluating models in standardized datasets by the computer vision community facilitated the distribution of the trained models at large. The popularization of pretrained models supported the adoption of transfer learning (e.g., [10–13]) by other fields of science in which the amount of labelled data is not as large. The field of geosciences is one of many in which the use of transfer learning has recently become popular.

The main idea behind transfer learning comes from the realization that the representation of the input generated by the layers in a neural network are generic for the layers

closer to the input, and more complex and abstract for layers closer to the output (or label), especially when trained with datasets of natural images such as ImageNet. The interpretability of CNN filters and their outputs is an active area of research, with several examples showing how images activate filters differently, for example [14–17]. Although the representations learned by CNN models contain the information needed to map the input to the output, these transformations can be useful for solving other outputs (tasks), and this procedure tends to be more successful if the tasks are related in some aspect [12,13,18,19]. In transfer learning, a model trained on a primary task (e.g., to classify the ImageNet dataset) is repurposed for a secondary task (e.g., to classify thin-section images). The repurposing step generally requires an adaptation of the model, as well as further training. Zamir et al. [19] investigated how different visual tasks were related to each other as a means of proposing better transfer learning strategies. Their results show, for example, that 2D segmentation is a task more similar to colorization and in-painting than to image denoising. They also observed that the results they found are model- and data-specific, meaning there are still knowledge gaps that should be studied. The objectives of this paper are well aligned with such observations. Here, we investigate the transferability of models trained on different datasets for the task of petrographic classification at a thin-section scale.

In many transfer learning applications, the similarity between the primary task and the secondary task is simplified up to the point of the type of data used. For instance, many transfer learning applications rely on the fact that the ImageNet is a dataset of RGB images; thus, the parameters learned by CNN models to classify an ImageNet dataset can be repurposed to classify other RGB image datasets. For example, Norouzzadeh et al. [20] repurposed CNN models pretrained on ImageNet to automate animal identification in a dataset composed of camera trap images. Tschandl et al. [21] used transfer learning to repurpose a model pretrained on ImageNet to classify pigmented lesions from different populations; Kather et al. [22] used transfer learning to identify microsatellite instability directly from histological images in gastrointestinal cancer. Both Hu et al. [18] and Pires de Lima and Marfurt [23] studied the use of transfer learning for the classification of high-resolution remote-sensing images, both using models pretrained on ImageNet. The list of studies using transfer learning for the classification of geoscience images is also expanding. Examples include Pires de Lima et al. [24], who used transfer learning for the classification of lithofacies using pictures of core data. In contrast to some of the examples cited, Baraboshkin et al. [25] reported inferior performance when using transfer learning than when training a CNN model with randomly initialized weights. Transfer learning showed itself to be highly valuable in the classification of petrographic thin-section images (e.g., [26–29]). All these transfer learning examples for the classification of geoscientific images repurpose models pretrained on ImageNet at some point in their analyses.

What is curious about these examples is that training a CNN model for the classification of ImageNet, an image dataset composed of a wide range of objects and scenes, always seems to improve the model's performance when transferred to a secondary task, even if the images from the secondary task are visually dissimilar to ImageNet (e.g., histology or skin lesion images). The classes in ImageNet have a within-class variance that is much larger than what we usually expect for petrographic thin-section classifications. To put this in perspective, ImageNet-trained models should be able to identify dogs or cats, independent of their scale, lightning conditions, or background. In fact, ImageNet is specifically constructed so that objects in images have different appearances, background clutter and occlusions, as well as varying positions, viewpoints, and poses [6]. In contrast, thin-section images are obtained in well-defined configurations, rendering images that have a much smaller variance, especially when the photographs are taken at the same zoom level. The hypothesis that leads the experiments presented in this manuscript is that models primarily trained on a dataset more visually similar to the secondary task should outperform models primarily trained on a dataset dissimilar to the secondary task. In other words, adapting the weights should be easier when tasks are related, which aligns with recent findings (e.g., [19]). The evaluation of similarity here is somewhat qualitative, but enough to discuss

the findings in depth. To perform such analysis, we evaluate the results of using transfer learning to classify thin-sections images using models primarily trained on ImageNet, the HAM10000 dataset [21], the RawFooT dataset [30], and part of Kather et al. [22]’s dataset (hereinafter, MSI vs. MSS standing for microsatellite instable vs. microsatellite stable). These datasets were selected based on their resolution, the number of samples available, and because they were previously used to train CNN models. Moreover, these datasets have standardized images, reducing unnecessary variance in the samples imaged—a contrast with ImageNet that aims to include variations, as described above. Understanding how the similarity between visual datasets affects the classification of petrographic thin sections can help us create better models, as well as better petrographic datasets.

The main contributions of this manuscript can be summarized as follows:

- An evaluation of how the difference between the dataset used to train a CNN model for the primary task affects the performance of such model when used for transfer learning for the classification of petrographic thin sections for the secondary task;
- The optimization of models that can accurately classify thin-section images from the Sycamore formation with two different magnification levels;
- How the amount of data affects CNN models used for transfer learning;
- How the similarity between primary and secondary tasks affects CNN models used for transfer learning;

Section 2 presents the datasets used for the primary task of image classification, as well as the CNN architectures used and the general methodology of the study. Section 3 shows the results of all experiments conducted, as well as their interpretation. Section 4 shows the discussion regarding the results obtained with the proposed experiments. Section 5 presents the conclusions.

2. Materials and Methods

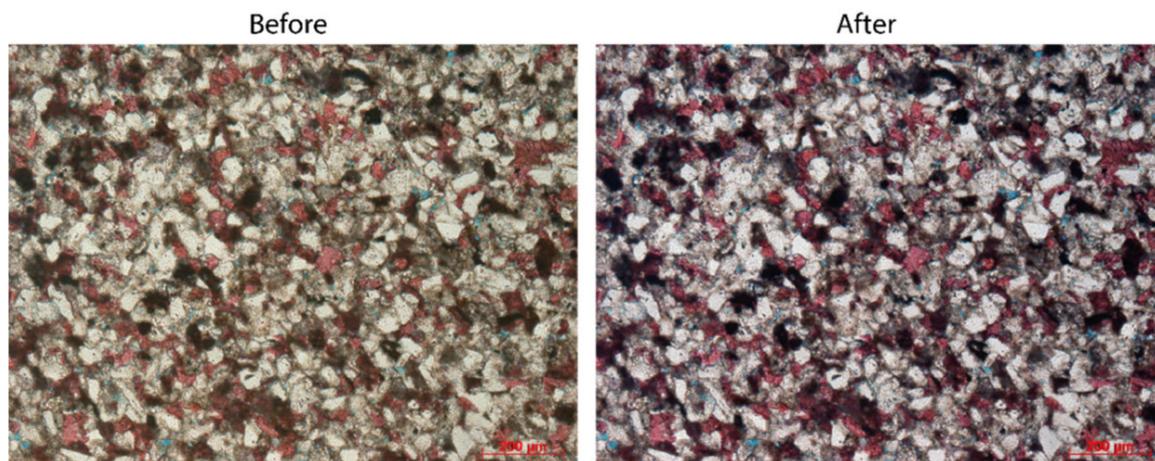
2.1. Petrographic Thin-Section Data

The central data used for transfer learning of this study are 98 thin sections acquired from 5 different cores from the Sycamore formation (early Mississippian strata) in the Ardmore basin, Oklahoma, the same thin sections used in [26]. We took roughly five randomly placed and generally non-overlapping photographic images for every thin section using plane polarized light and a $2.5\times$ or $10\times$ magnification zoom, attaining a total of 513 samples. Then, we classified the thin sections in four microfacies (classes) based on the texture and mineralogical composition. Although this classification took into account a general knowledge about the origin, depth, and geological background of the thin sections, some bias was introduced into the dataset, as the microfacies were defined based on our interpretation. Note, we mixed images with $2.5\times$ and $10\times$ magnification zoom. Even though the relative grain sizes are different at different magnification levels, we argue that the petrographic characteristics that explain the interpreted microfacies are well defined in both scales; thus, this will not negatively affect the models. Table 1 shows the count for each one of the microfacies interpreted, as well as the count of samples in the train and test sets. Except for Argillaceous mudstone (AMdst), the classes are somewhat well balanced. The images were then color balanced following [31], which assumed that the highest values of RGB observed in a photograph corresponded to white and the lowest values corresponded to black (Figure 1a). Moreover, [32] investigated the use of color balancing and found it helpful for image classification using CNN. The same methodology is used in [26]. We then resized the images to 646×484 pixels, maintaining the same aspect ratio of the original photograph and maintaining the same pixel area relation. During training and inference, we used the five-crop technique to augment the data, extracting 224×224 samples from the corners and the center of the images (Figure 1b). The final prediction for a given sample was the resulting mean average prediction across the five crops. The five-crop is a common technique frequently used in computer vision tasks and somewhat simpler than what was discussed in [26]. We then computed the mean of the RGB channels of the training, as well as their standard deviations useful for the normalization of the data, and obtained (0.3579,

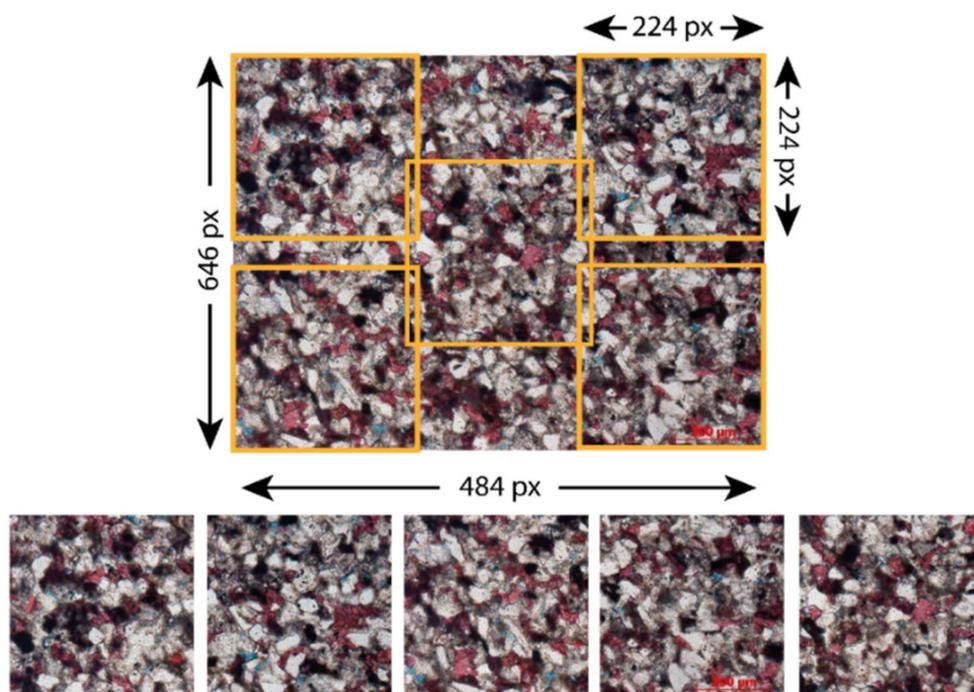
0.2924, 0.3122) for the mean and (0.2028, 0.2011, 0.1877) for the standard deviation. Figure 2 shows examples of images in the test set.

Table 1. Thin-section data details.

Class	Microfacies	Description	Train	Test
AMdst	Argillaceous mudstone	Clay-rich mudstones. Structureless or slightly laminated.	63	20
BMdst	Bioturbated mudstone	Clay-rich mudstones. Evident bioturbation at 10X magnification.	134	20
MCcSt	Massive calcite cemented siltstone	Silt-rich mudstones. Structureless, abundant calcite cemented and calcareous pellets.	104	20
MCSt	Massive calcareous siltstone	Silt-rich mudstones. Structureless, some calcite cement.	132	20



(a)



(b)

Figure 1. Thin-section data: (a) color balancing effect;(b) five-crop technique. The bottom row shows each one of the five crops extracted from the original sample and used for training and inference.

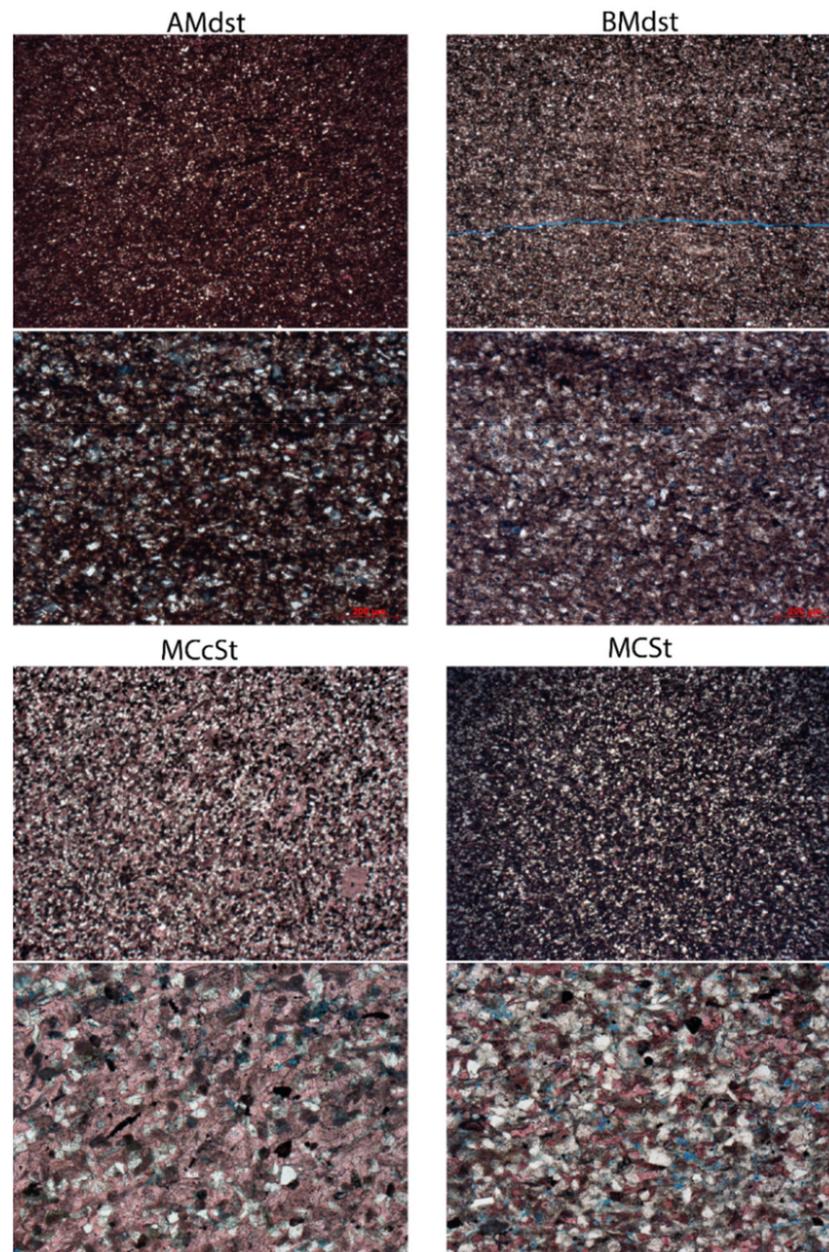


Figure 2. Examples of images in the petrographic thin-section test set and their corresponding labels. The figure shows two images for each one of the classes. The top image of each class is taken with a magnification zoom of $2.5\times$, the bottom image with $10\times$. The red labels in the bottom right-hand corner are randomly present in some $10\times$ images, indicating $200\ \mu\text{m}$. Class names and descriptions are provided in Table 1.

2.2. ImageNet

ImageNet is one of the most popular computer vision datasets. The full dataset is composed of over 15 million labeled high-resolution images belonging to roughly 22,000 classes. The dataset images were of variable resolutions and were collected from the internet and labelled by humans. Many computer vision experiments are conducted on a subset of the ImageNet dataset that originated from an annual competition called the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). The ILSVRC subset of ImageNet has roughly 1000 images in each of the 1000 categories [3,4]. Weights from models trained with the ILSVRC (hereinafter simplified back to ImageNet) are likely the most widely available and are used for transfer learning. We downloaded pretrained weights directly from [33].

2.3. HAM10000

Tschandl et al. [21] released the “Human Against Machine with 10,000 training images”—HAM10000 dataset to facilitate training of neural networks for automated diagnosis of pigmented skin lesions. The dataset contains dermatoscopic images from different populations, acquired and stored by different modalities. The published dataset consists of 10,015 dermatoscopic images which can serve as a training set for academic machine-learning purposes. The images comprise a representative collection of many important pigmented lesion categories. Figure 3 shows some samples of the test set. Over 50% of the imaged lesions were confirmed through histopathology, and the remaining samples were confirmed through either follow-up examination, expert consensus, or confirmation by *in vivo* confocal microscopy. The train/test split is provided by the authors. We computed the mean of the RGB channels for the train set (0.7637, 0.5461, 0.5707), with a standard deviation of (0.0897, 0.1184, 0.1330).

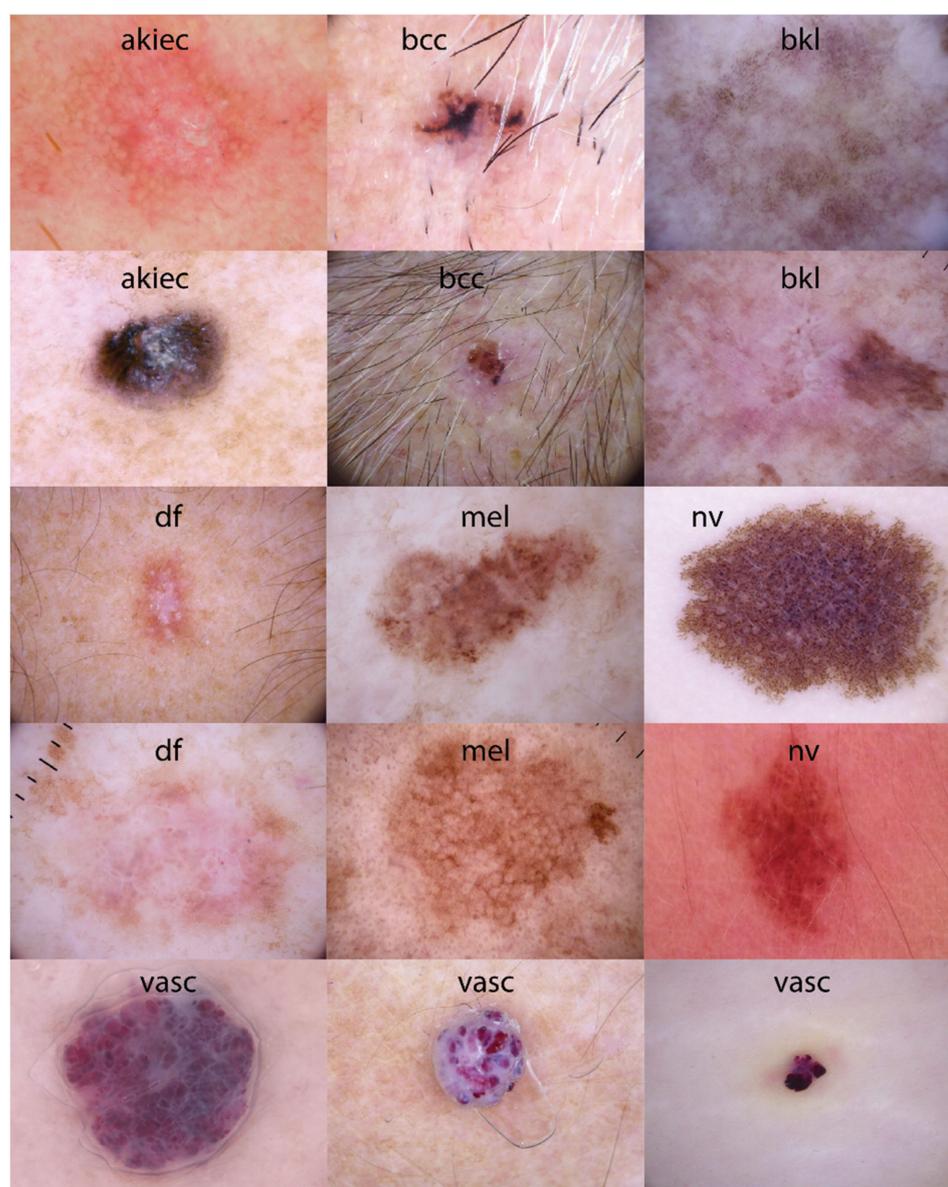


Figure 3. Examples of images in the HAM10000 [21] test set and their corresponding labels. Table 2 shows class names and number of samples in the dataset.

Table 2. The HAM10000 dataset details.

Class	Description	Train	Test
bkl	benign keratosis-like lesions (solar lentigines/seborrheic keratoses and lichen-planus like keratoses)	871	228
nv	Melanocytic nevi	5367	1338
df	Dermatofibroma	87	28
mel	Melanoma	887	226
vasc	Vascular lesions (angiomas, angiokeratomas, pyogenic granulomas and hemorrhage)	121	21
bcc	Basal cell carcinoma	421	93
akiec	Actinic keratoses and intraepithelial carcinoma/Bowen's disease	258	69

2.4. RawFoot

The Raw Food Texture (RawFoot) dataset was designed to study the robustness of classification methods with respect to variations in lighting conditions. The dataset includes images of food with different visual texture, acquired under 46 lighting conditions, with variations in the light direction, in the illuminant color, in its intensity, or in a combination of such factors. The dataset contains 68 classes of raw food and includes different kinds of meat, fish, cereals, bread, and other food [30]. The dataset is available in full image (800 × 800 pixels) and tile (200 × 200 pixels) formats. We used the tiles in this study. The tile images correspond to subdivisions of the original 800 × 800 images into 16 square regions. The authors of the RawFoot dataset proposed a checkerboard pattern and selected eight square regions to be part of the train set, while the remaining eight square regions were allocated to the test set. We used the split defined by the authors. We calculated the mean of the RGB images in the train set (0.3765, 0.2743, 0.1351) and the standard deviation (0.0571, 0.0537, 0.0433). Figure 4 shows examples of images in the test set. Table 3 shows the table with the class names and the count of samples per set.

Table 3. The RawFoot dataset details.

Class	Description	Train	Test	Class	Description	Train	Test
0001	chickpeas	368	368	0035	hazelnut grain	368	368
0002	corn	368	368	0036	flour	368	368
0003	salt	368	368	0037	bread crumbs	368	368
0004	cookie	368	368	0038	pasta (stars)	368	368
0005	lentils	368	368	0039	cut spaghetti	368	368
0006	candies	368	368	0040	pastina	368	368
0007	green peas	368	368	0041	red cabbage	368	368
0008	puffed rice	368	368	0042	grapefruit	368	368
0009	spelt	368	368	0043	hamburger	368	368
0010	white peas	368	368	0044	swordfish	368	368
0011	cous cous	368	368	0045	bread	368	368
0012	sliced bread	368	368	0046	candied fruit	368	368
0013	apple slice	368	368	0047	chili pepper	368	368
0014	pearl barley	368	368	0048	milk chocolate	368	368
0015	oat	368	368	0049	garlic grain	368	368
0016	black rice	368	368	0050	curry	368	368
0017	quinoa	368	368	0051	pink pepper	368	368
0018	buckwheat	368	368	0052	kiwi	368	368
0019	puffed rice	368	368	0053	mango	368	368
0020	basmati rice	368	368	0054	pomegranate	368	368
0021	steak	368	368	0055	currant	368	368
0022	fennel seeds	368	368	0056	pumpkin seeds	368	368
0023	poppy seeds	368	368	0057	tea	368	368
0024	brown sugar	368	368	0058	red lentils	368	368
0025	sultana	368	368	0059	green adzuki	368	368

Table 3. *Cont.*

Class	Description	Train	Test	Class	Description	Train	Test
0026	coffee powder	368	368	0060	linseeds	368	368
0027	polenta flour	368	368	0061	coconut flakes	368	368
0028	salami	368	368	0062	chicory	368	368
0029	air-cured beef	368	368	0063	pork loin	368	368
0030	flatbread	368	368	0064	chicken breast	368	368
0031	corn crackers	368	368	0065	carrots	368	368
0032	oregano	368	368	0066	sugar	368	368
0033	black beans	368	368	0067	salmon	368	368
0034	soluble coffee	368	368	0068	tuna	368	368

**Figure 4.** Examples of images in the RawFoot [30] test set and their corresponding labels. Table 3 shows class names and number of samples in the dataset.

2.5. MSI vs. MSS

Knowing that gastrointestinal cancer MSI patients respond exceptionally well to immunotherapy, and that not every patient is tested for MSI, Kather et al. [22] show that ResNets can predict MSI directly from H&E (hematoxylin and eosin stain) histology images.

Kather et al. also used tiles of larger high-resolution slides for their analysis, and made the tiles available for download. The tiles available for download are separated into train and test set samples with 224×224 pixels, which is equivalent to $0.5 \mu\text{m}$ per pixel. We used the colorectal cancer images only. Figure 5 shows examples of the dataset. Table 4 shows the number of samples in the train and test sets. We computed the mean RGB (0.7263, 0.5129, 0.6925) and standard deviation (0.1444, 0.1833, 0.1310) of the images in the train set. This dataset is named MSI vs. MSS in this study.

Table 4. The MSI vs. MSS dataset details.

Class	Description	Train	Test
MSI	Microsatellite instable	46,704	28,335
MSS	Microsatellite stable	46,704	70,569

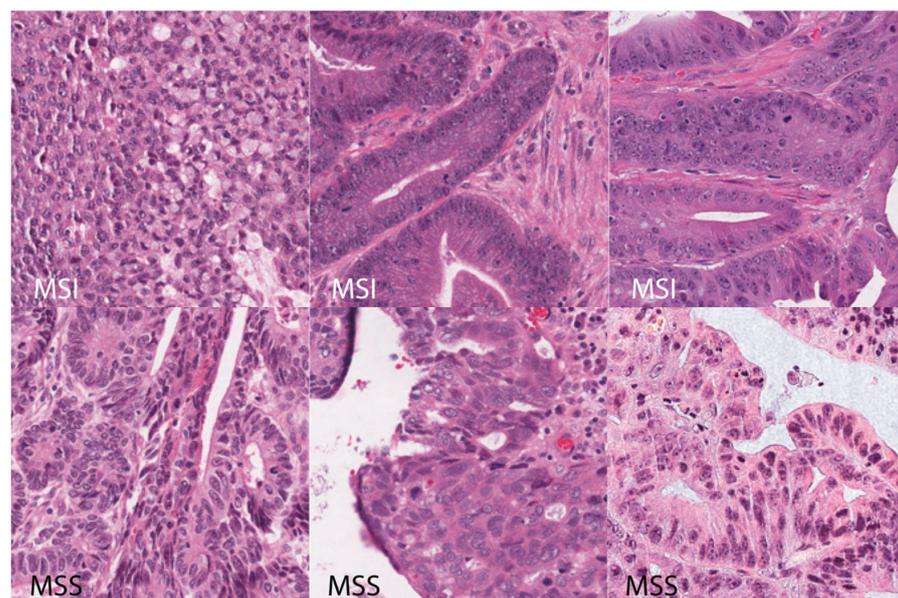


Figure 5. Examples of Kather et al. [22]’s images in the MSI vs. MSS test set and their corresponding labels. Table 4 shows class names and number of samples in the dataset.

2.6. Transfer Learning and Implementation Details

The methodology behind transfer learning is well described in many studies (e.g., [12,13,18,23]). For instance, [26,27,29] described the technique using petrographic thin-section data. Thus, in this paper, we provide only a brief explanation of the transfer learning methodology. Initially, CNN models are created with randomly initialized weights. During training, the weights are updated according to the objective function, reducing the loss of the model regarding the output. For classification problems, the loss is computed according to the true and predicted labels. After training, the weights of the models are in a stage such that they are useful for classifying the input image data. In other words, the models learn to extract features that are useful for classification. What was described is the default training methodology, when the weights of the model were randomly initialized and updated through gradient descent during training. In the transfer learning training methodology, the model previously trained with randomly initialized weights for the primary task is further trained on the secondary task. The trained models can be used as feature extractors, when part of the weights of the models is not updated for the secondary task, or they can be fine-tuned, when all the weights are updated, onto the secondary task. Here, we use the fine-tuning technique.

We used residual nets (ResNets [34]) as the CNN architecture for the experiments, specifically ResNet18 and ResNet50. ResNets introduce shortcut connections to CNN

architectures as a way to address the problem of vanishing gradients. The skip connections allow the gradients to flow from layers close to the output, where the loss is computed, to layers closer to the input. This strategy enabled the adoption of deeper networks, facilitating the optimization of large models. ResNet18 (ResNet50) is composed of 18 (50) parameter layers, either convolution or fc layers. ResNets end with a global average pooling layer, followed by an fc layer. For ResNets trained on ImageNet, the fc layer placed after the global average pooling layer contains 1000 neurons (fc 1000), one for each class of the dataset. In the implemented transfer learning methodology, we randomized the fc 1000 and added another fc layer, with a neuron for each one of the classes of the new dataset. In the fine-tuning technique implemented here, only the fc layers were updated in the first five epochs; then, the remaining layers were unfrozen and all the weights were updated for the remaining epochs.

For all experiments, 20% of the train dataset was randomly selected to be part of the validation set. The model continues training while the validation loss decreases, with a patience of five epochs unless otherwise noted. Patience is a hyperparameter indicating the number of epochs after which the model stops training. The thin-section data was cropped to desired dimensions of 224×224 pixels using the five-crop technique described previously. The images of the remaining datasets were resized to 224×224 pixels. After cropping or resizing, the images were normalized following the computed dataset mean and standard deviation. Data augmentation was used for the train set only, not on the validation or test sets, and used the following pipeline: horizontal flip, vertical flip, and rotation limited to ± 5 degrees, all with 50% probability. The hyperparameter search included mostly batch size by accumulation of gradients, learning rate, and optimizers. Adam [35] was the preferred optimizer, but RMSprop [36] was also evaluated. We used the PyTorch [33] framework for implementation, following PyTorch Lightning [37] structures. Experiments were tracked with Weights and Biases [38].

3. Results and Interpretation

3.1. Results

We start this section showing the results obtained training ResNet18 and ResNet50 for the classification of the HAM10000, RawFoot, and MSI vs. MSS datasets. To recall, these datasets were chosen because they are more visually similar to petrographic thin-section image than the ImageNet; samples from the former, for example, are in the same scale and there are no backgrounds. We evaluated different hyperparameters using the models and datasets described in Section 2. Appendix A (Tables A1 and A2) shows details of the hyperparameters used for training the CNN models on the primary task. Table 5 shows the accuracy of the best performing models on the test set for each one of the datasets, using both ResNet18 and ResNet50. Other performance metrics are important for the evaluation of classification tasks, especially when there are different costs for different classes or when dealing with unbalanced datasets (e.g., [39,40]), but these are omitted here for the sake of simplicity. The results in Table 5 show some overfitting for both MSI vs. MSS and RawFoot datasets, even though the train data were augmented during training. Moreover, the accuracy of ResNet18 on the RawFoot dataset is noticeably higher than the ResNet50. The difference in accuracy between train and test sets for the MSI vs. MSS is considerably larger—0.2 for ResNet18 and 0.19 for ResNet50. Nonetheless, the trained models learned weights that can serve as effective feature extractors for the classification of each one of the datasets, unlike the initial random weights. Thus, the models are then adapted to the secondary task.

The baseline for the classification of thin-section data is a model started with randomly initialized weights. We performed a hyperparameter search training ResNet18 and ResNet50 with the petrographic thin-section data described in Section 2.1. Figure 6 shows the computed accuracy and loss during training for the best performing hyperparameters in ResNet18 and ResNet50 models. Although the train and validation metrics walk closely together for ResNet50, the model takes longer to achieve a higher performance.

Thus, we increased the patience for all ResNet50s to 10 to accommodate possible spurious fluctuations in the validation loss that could make the model stop training before desired. ResNet18 decreases the loss (increases the accuracy) more rapidly; however, it also shows signs of overfitting earlier. ResNet18 models are maintained with a patience of five.

Table 5. Accuracy of the best performing models for the classification of each one of the datasets used for the primary task.

Dataset	Model	Train	Validation	Test
MSI vs. MSS	ResNet18	0.94	0.91	0.71
RawFoot	ResNet18	0.98	0.96	0.88
HAM10000	ResNet18	0.79	0.77	0.76
MSI vs. MSS	ResNet50	0.91	0.90	0.71
RawFoot	ResNet50	0.89	0.87	0.73
HAM10000	ResNet50	0.80	0.76	0.75

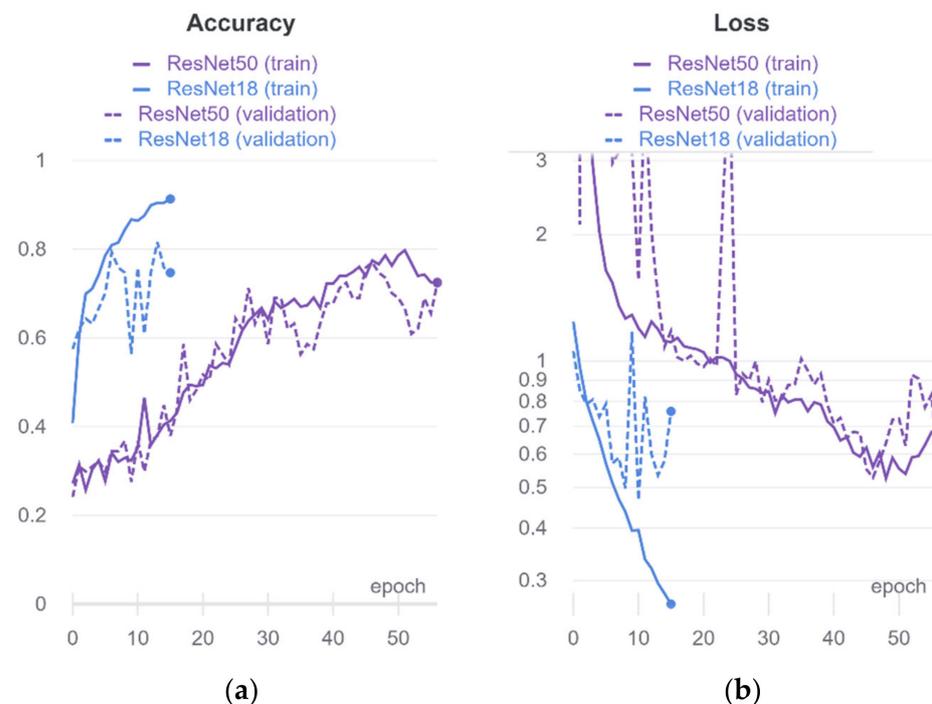


Figure 6. Training accuracy and loss for models trained on the thin-section data when the model starts with randomly initialized weights: (a) accuracy by epoch for the train and validation sets of models ResNet18 and ResNet50; (b) corresponding loss by epoch.

Table 6 shows the hyperparameters searched while training ResNets with randomly initialized weights on the petrographic thin-section data. Due to GPU memory limitations, ResNet18s are trained with a batch size of 16 while ResNet50s use a batch size of 4. The batch size is increased before the optimizer steps by the accumulation of gradients. Results show that minor changes in the choice of hyperparameters can have significant effects in models’ performances. For example, changing the batch size (accumulated) for ResNet18 from 32 to 64 increased the accuracy by 0.03 in the test set, but increasing it again from 64 to 128 reduced the accuracy by 0.03 (Test accuracy, rows 1, 2, and 4 in Table 6). Some hyperparameters bring the performance to noticeable sub optimal results (e.g., rows 5 and 8). The model in row 5 trained only for 8 epochs, which was smaller than most of the experiments reported for ResNet18. This happened because the model achieved a small validation loss in epoch three and then oscillated with slightly larger validation loss values. Due to the patience (five epochs), the model stopped after epoch eight. Changing only the learning rate (row 4), the model is able to improve the accuracy in the test set by 0.2.

Results in rows 8 and 10 show that the optimizer can have a significant impact on the performance of ResNet50, where Adam improves the accuracy in the test set by 0.16 when compared to RMSprop with the same hyperparameters. ResNet18 achieves a mean average of 0.82 for test accuracy, with the worst model achieving 0.66 and the best model reaching 0.89 accuracy. Hereinafter, we summarize the information just provided as: 0.82 (0.66, 0.89). ResNet50 achieves accuracies of 0.75 (0.64, 0.82). Training ResNet18 with the petrographic thin-section data, and using the five-crop strategy, is relatively cheap, with one epoch taking less than one minute with a laptop-based GTX 1050 GPU. ResNet50 training is a little more expensive, taking roughly 1.2 min per epoch with the same hardware.

Table 6. Training hyperparameters and performance for models trained on the thin-section data when the model starts with randomly initialized weights.

Model	Batch Size (Accumulated)	Optimizer	Learning Rate	Last Epoch	Train Accuracy	Validation Accuracy	Test Accuracy
ResNet18	128	Adam	1.00×10^{-4}	18	0.90	0.79	0.86
ResNet18	64	Adam	1.00×10^{-4}	15	0.91	0.75	0.89*
ResNet18	32	Adam	5.00×10^{-5}	15	0.90	0.67	0.81
ResNet18	32	Adam	1.00×10^{-4}	10	0.86	0.72	0.86
ResNet18	32	Adam	1.00×10^{-3}	8	0.78	0.67	0.66
ResNet50	256	Adam	1.00×10^{-3}	33	0.79	0.64	0.76
ResNet50	64	Adam	1.00×10^{-3}	46	0.93	0.75	0.74
ResNet50	128	RMSprop	1.00×10^{-3}	31	0.58	0.53	0.64
ResNet50	128	Adam	1.00×10^{-2}	38	0.71	0.70	0.77
ResNet50	128	Adam	1.00×10^{-3}	36	0.84	0.71	0.80
ResNet50	128	Adam	5.00×10^{-3}	56	0.73	0.72	0.82
ResNet18	128	Adam	1.00×10^{-4}	18	0.90	0.79	0.86*

* Best performing models shown in Figure 6.

With the results obtained by training CNN models with weights that were randomly initialized as the baseline for the classification of petrographic thin-section images, we continued with the transfer learning analysis. The ResNet18 and ResNet50 models were pretrained on the datasets described in Sections 2.2–2.5, and are fine-tuned for the classification of the dataset described in Section 2.1. As before, we perform a hyperparameter search to evaluate how the performance of the models is affected. The full combination of hyperparameters, as well as train, validation, and test set accuracies, is presented in Appendix B (Table A3).

Figure 7 and Table 7 show the most important summary of results of the transfer learning experiments, with details on the test set accuracies. As before, results show that ResNet18 tends to have better performance than ResNet50 in the petrographic thin-section image data; this is likely due to the limited data size. Curiously, training the networks with randomly initialized weights or fine-tuning the proposed datasets have, in general, a comparable performance for mean accuracy. The mean accuracy of ResNet50 trained on the HAM10000 and on the MSI vs. MSS is considerably smaller than the baseline results. However, the best performing models, those showing higher accuracy on the test set, are slightly larger for most of the fine-tuned models. The only exception is ResNet50 pretrained on the MSI vs. MSS dataset, which has the same maximum accuracy as the ResNet50 trained with randomly initialized weights. Although the proposed pretraining datasets provided only a marginal or comparable results than what can be achieved by training networks trained with randomly initialized weights, the technique of pretraining the model on the ImageNet showed the most stable and accurate results. Pretraining the models on ImageNet improved the networks' accuracy, and the results for ResNet18 are consistent, even with different choices of hyperparameters. Although RMSprop was generally detrimental for models pretrained on the other datasets, RMSprop and Adam optimizers with the same batch size and learning rate (64, and 1×10^{-4} , respectively) achieved the highest performance for ResNet18 pretrained on ImageNet. Figure 8 shows a

comparison of the accuracy on the test set for different hyperparameters, identifying the optimizer choice, for ResNet50 trained with randomly initialized weights and pretrained on ImageNet. As previously demonstrated, the results show that pretraining on ImageNet helps the model achieve higher accuracy. The figure makes it clear that RMSprop does not affect the performance negatively when the model is pretrained on ImageNet, whereas RMSprop is detrimental when using randomly initialized weights.

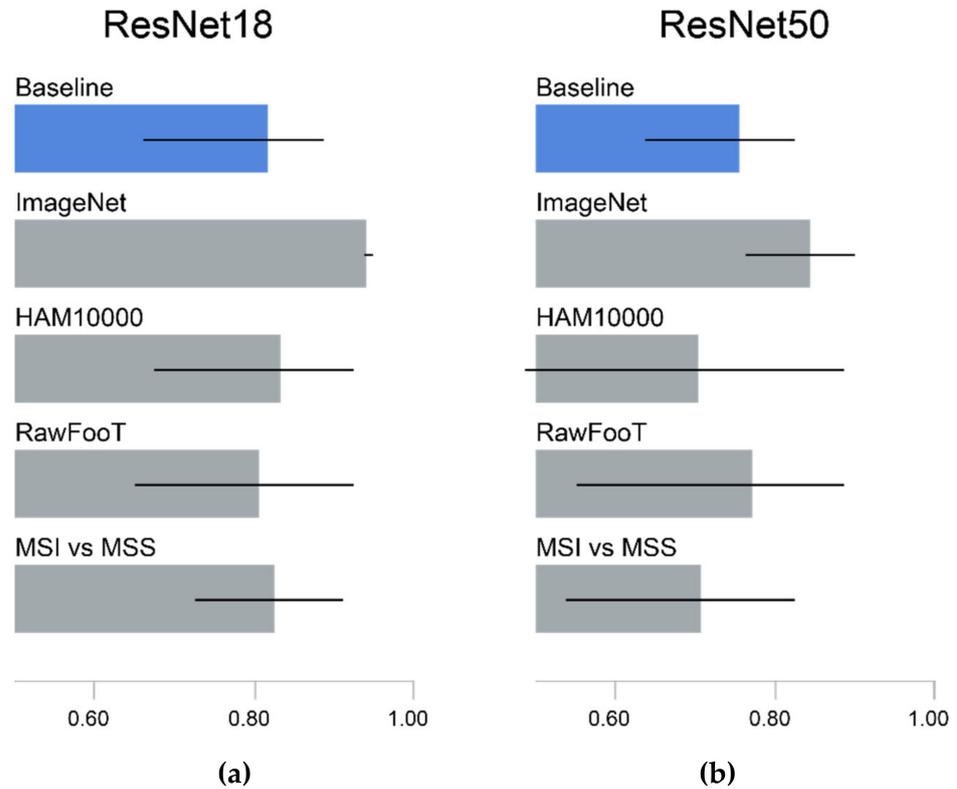


Figure 7. Test set accuracy for the transfer learning experiments: (a) accuracy for fine-tuned ResNet18 and (b) accuracy for fine-tuned ResNet50. Solid bars indicate the mean accuracy values. Black lines indicate minimum and maximum accuracy values. Blue bar on top indicates the baseline models trained with randomly initialized weights, as discussed in the text. The mean, minimum, and maximum values are shown in Table 7.

Table 7. Accuracies displayed in Figure 7.

Dataset	Model	Mean	Minimum	Maximum
Baseline	ResNet18	0.82	0.66	0.89
ImageNet	ResNet18	0.94	0.94	0.95
HAM10000	ResNet18	0.83	0.68	0.93
RawFoot	ResNet18	0.81	0.65	0.93
MSI vs. MSS	ResNet18	0.83	0.73	0.91
Baseline	ResNet50	0.75	0.64	0.82
ImageNet	ResNet50	0.84	0.76	0.90
HAM10000	ResNet50	0.70	0.49	0.89
RawFoot	ResNet50	0.77	0.55	0.89
MSI vs. MSS	ResNet50	0.71	0.54	0.82

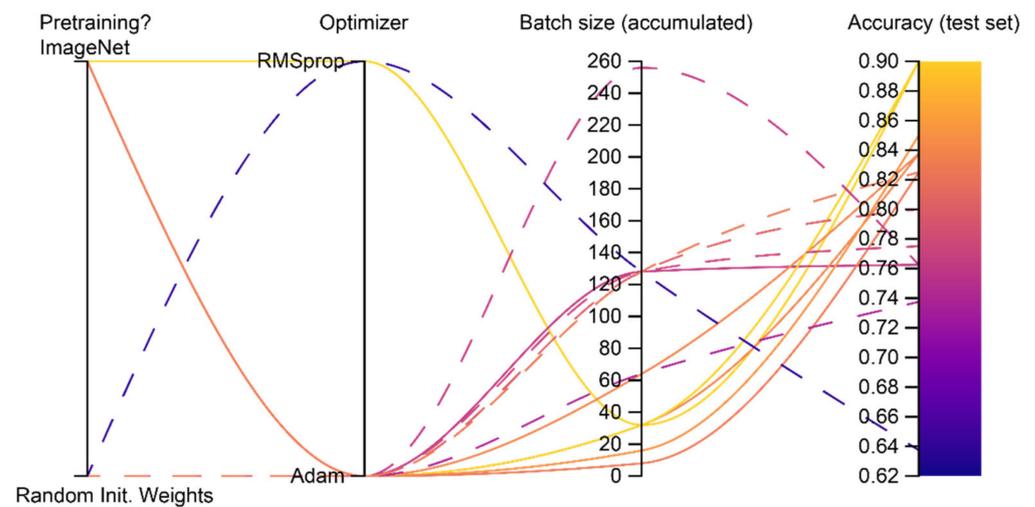


Figure 8. Experiment comparison by hyperparameter for ResNet50. This image shows some of the hyperparameters tested, as well as the accuracy in the test set. The solid lines show results for when the model is pre-trained on the ImageNet and dashed lines show results for ResNet50 trained with randomly initialized weights.

The following figures focus on ResNet18, as its results are better than ResNet50. Figure 9 shows the best performing fine-tuned ResNet18 models when pre-trained on the ImageNet, HAM10000, and MSI vs. MSS datasets. ResNet18 pre-trained on RawFoot shows similar behavior; however, it trained for longer. Results show a break in continuity around epoch five, when the full model is unfrozen and all the weights can be updated. Such a break is more pronounced on the ResNet18 trained on ImageNet (grey curves in Figure 9). Results show that the network tends to overfit after some epochs, with the distance between the train and validation loss (accuracy) keeping somewhat constant or increasing (decreasing). Regularization and decreases in the learning rate can help with overfitting. We choose to present only the best performing models, but the curves obtained during hyperparameter search behave similarly.

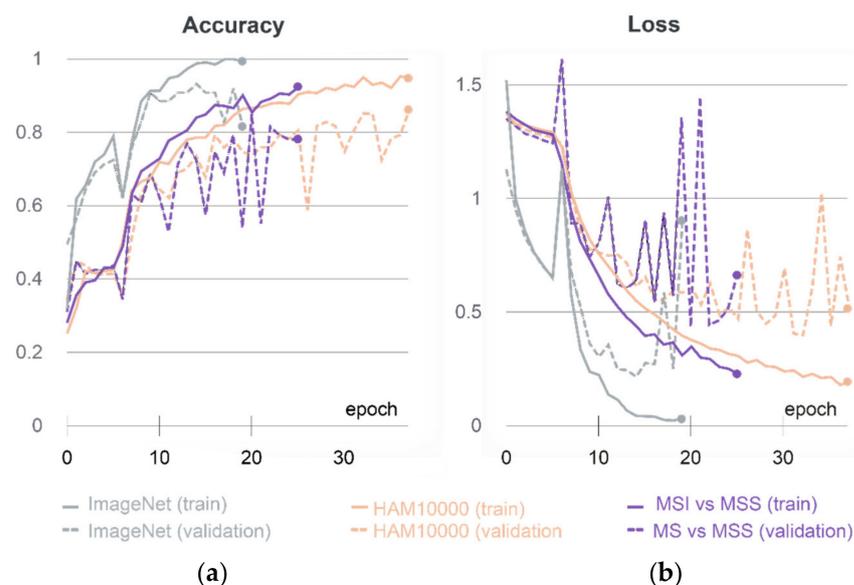


Figure 9. Training accuracy and loss for ResNet18 fine-tuned on the thin-section data: (a) accuracy by epoch for the train and validation sets best performing ImageNet, HAM10000, and MSI vs. MSS pre-trained data; (b) corresponding loss by epoch.

Figure 10 shows the confusion matrix computed on the test set for the best performing models, including the baseline model trained with randomly initialized weights, as well as the fine-tuned models for ResNet18. Results show, in general, that ResNet18 is capable of correctly classifying most of the images. The most significant confusion seems to be for the baseline between the classes BMdst and AMdst. ResNet18 pretrained with the MSI vs. MSS dataset shows a similar performance, but also confuses one BMdst with MCcSt. ResNet18 pretrained with the ImageNet dataset incorrectly classified AMdst as BMdst twice. The remaining confusion is caused by very few, generally one or two, images classified incorrectly. Results in Figure 10 indicate that ResNet18 can achieve high levels of accuracy for the classification of thin-section image data, regardless of pretraining. It also shows that adequate pretraining, when limited data are available, can aid in network differentiation between classes in which the between-class variance is small. Appendix C (Tables A4–A8) shows complementing metrics for the results in Figure 10.

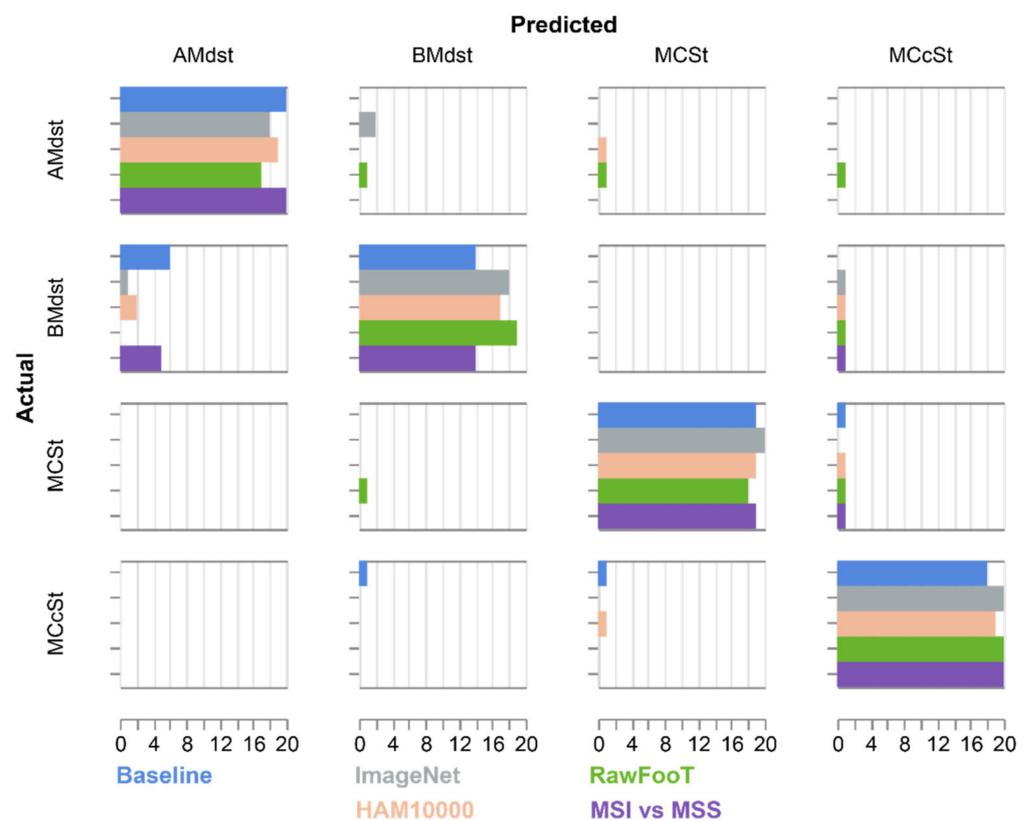


Figure 10. Confusion matrix computed on the test set for ResNet18 models. The color of the bars corresponds to the pretraining dataset. The size of the bar indicates the number of samples classified in each of the actual vs. predicted locations. Description of each one of the classes is given in Table 1.

Moving away from the performance of the models, the next set of figures shows the weights of the models. The first convolutional layer of ResNet18 is composed of 64 kernels with a size of 7 by 7. They represent a fraction of the layers in ResNet18, but they are closer to the model input and are thus easier to exhibit and interpret. Each one of these 64 kernels aligns with a RGB image; therefore, they can also be shown as RGB images. Figure 11 shows the 64 weights of the baseline ResNet18, as well as the fine-tuned ResNet18 pretrained on ImageNet. The weights in Figure 11a are closer to random when compared to the weights in Figure 11b. The weights in Figure 11b are very similar before and after fine-tune, meaning that there are almost not changes during the fine-tuning process.

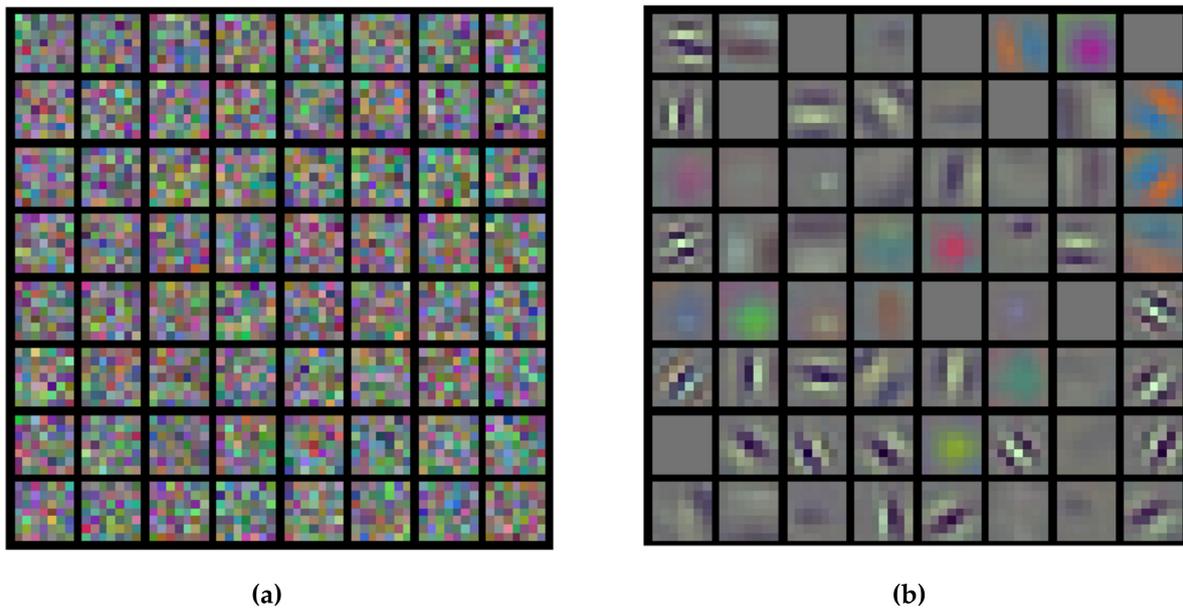


Figure 11. Weights for the first convolutional layer for ResNet18 used for the classification of the thin-section data: (a) baseline weights; (b) ResNet18 pretrained on ImageNet and fine-tuned on the thin-section dataset.

Figure 12 shows a comparison of the weights when RawFoot is the dataset used for the primary task. Figure 12a shows the weights of the first convolutional layer of the model before fine-tuning, and Figure 12b shows the weights after fine-tuning on the thin-section dataset. The weights in Figure 12a act as different colors and edges detectors. In contrast to Figure 11b, the edge filters in Figure 12a show a sensitivity to color and are not as well defined. There are also semi-circular filters, or color blobs. The weights in Figure 12b retain some of such features, but exhibit weaker organization, moving towards values similar to the baseline ResNet18 model (Figure 11a).

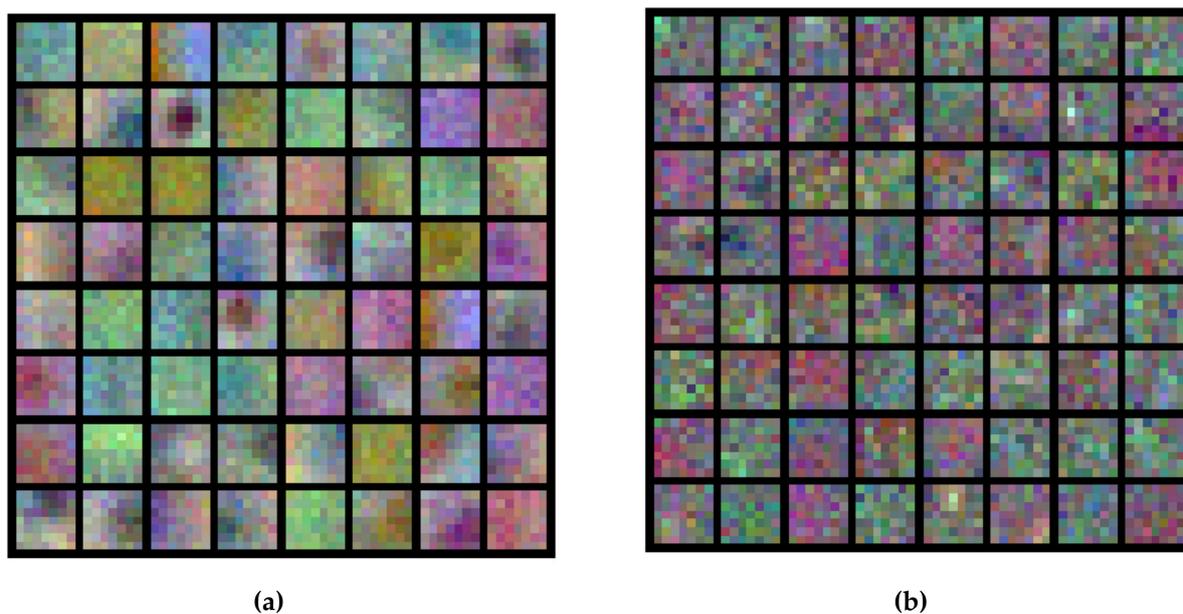


Figure 12. Weights for the first convolutional layer for ResNet18 used for the classification of the thin-section dataset: (a) ResNet18 trained on RawFoot; (b) same weights after the model is fine-tuned on the thin-section dataset.

This transition towards weights similar to what is observed for the baseline is also evident in the weights when MSI vs. MSS is the dataset used for the primary task. Figure 13a shows the weights of the model trained on MSI vs. MSS before fine-tuning, and Figure 13b shows the weights of the model after fine-tuning on the thin-section dataset. The weights in Figure 13a show somewhat oriented edge detectors, as well as some color blobs. Much of such organization, however, is lost after the model is fine-tuned on the thin-section dataset (Figure 13b). ResNet18 pretrained on HAM10000 shows a similar behavior.

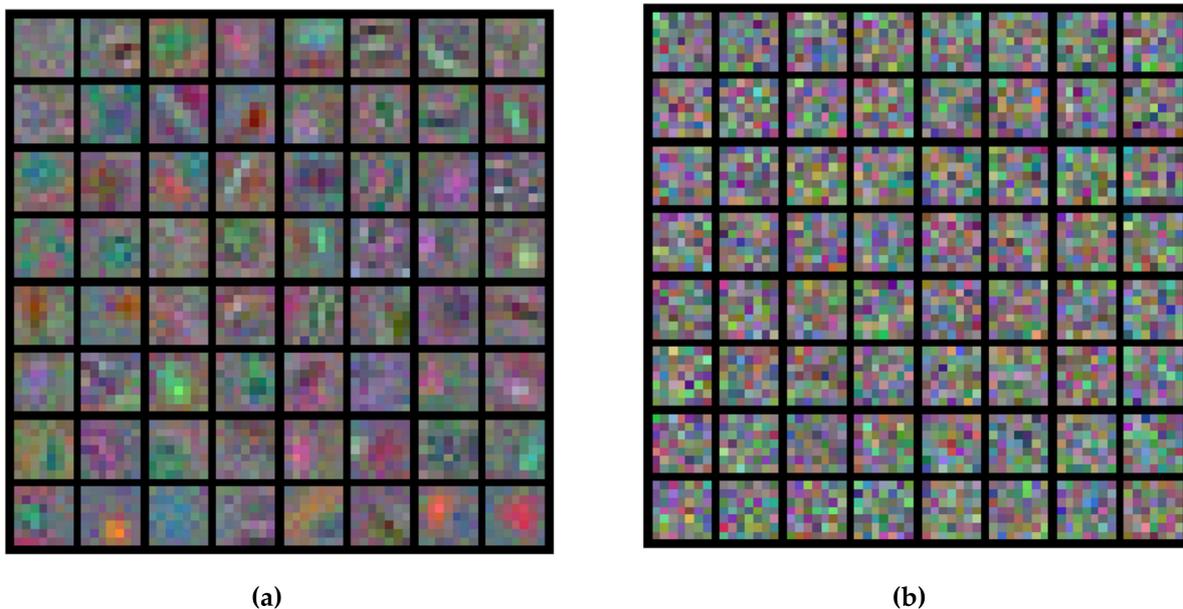


Figure 13. Weights for the first convolutional layer for ResNet18 used for the classification of the thin-section data: (a) ResNet18 trained on MSI vs. MSS; (b) same weights after the model is fine-tuned on the thin-section dataset.

3.2. Interpretation

We interpret that the higher accuracy of ResNet18 when compared to ResNet50 in Table 5 is an indication that the accuracy becomes saturated even with residual connections, or an indication that ResNet50 is harder to train due to its larger number of parameters. Further investigation and different hyperparameters might diminish this issue, as accuracy saturation is one of the issues ResNets were proposed to address. Table 5 also shows that there is a large difference in accuracy between train and test sets for the MSI vs. MSS for both ResNet18 (0.2) and ResNet50 (0.19). We interpret that one of the main reasons for this difference is that the training data are perfectly balanced, whereas MSS data are much larger than MSI on the test set (Table 4).

Table 6 shows that ResNet18 also has a better performance than ResNet50 in the thin-section data. In this case, the interpretation is that the reduced performance is explained by the large number of parameters in the larger model. Without a sufficiently large dataset, ResNet50 has a greater capability to overfit the training data. The table also shows the importance of the choice of hyperparameters. Two examples are easily noted due to their weak performance in comparison to the other experiments, namely rows 5 and 8. As described, the interpretation for the difference in performance is attributed to the choice of hyperparameters.

The most noticeable pattern in the confusion matrix in Figure 10 is the misclassifications of the models between the classes BMdst and AMdst, where the models classify BMdst as AMdst; this is stronger for the baseline model. This confusion is caused because these two classes are very similar clay-rich facies; the only difference between them is the amount of bioturbation, which is higher in the BMdst. Additionally, this bioturbation is described using $2.5\times$ magnification, but it is not particularly evident when using 10X zoom.

However, this demonstrates the accuracy of the models when it comes to differentiating between two classes, even if only showing small differences and being described using different magnifications than the ones used in the model. Figure 10 also shows that the ResNet18 pretrained on RawFoot was the only model that did not incorrectly classify BMdst samples as AMdst. We interpret that the combination of lighting and texture present in RawFoot helped the model to learn filters that are also useful for bioturbation detection, although this is not clearly defined in the weights in Figure 12. In contrast to the challenges between AMdst and BMdst, the models show better results when differentiating between MCSt and MCcSt. The difference between them, MCSt and MCcSt, is the amount of calcite cement that is evident at different scales. Moreover, the samples are stained with red Alizarin, which is used for calcite identification, which also helps the models.

The results in Figure 7 and Table 7 are somewhat unexpected because they indicate that training the networks with randomly initialized weights or fine-tuning the proposed datasets provide similar mean accuracy results, whereas fine-tuning networks pretrained in ImageNet achieves higher performance. Based on Figures 11–13 our interpretation is that the features learned by models trained on ImageNet are more general. For example, Figure 11b shows several edge detectors mostly black and white, indicating color invariance, and some well-defined color blobs. Results in Figures 12a and 13a show some similarity to the ImageNet weights, however sensible to color contrast (the edge detectors are colorful and blobs are not surrounded by a grey background). As the weights obtained when trained on the proposed datasets seem more specific to each one of the datasets used in the primary task, these weights are more disturbed during the fine-tuning and, in general, move towards the values found for the baseline in Figure 11a. Curiously as well, these baseline weights do not show easily recognizable features and visually appear almost random. This is likely due to a combination of the small number of samples in the dataset, the general nature of the thin section images where minerals are generally somewhat randomly distributed, and the small subsection of weights (only one layer) investigated.

4. Discussion

As previously observed in several studies, the results presented here show once again that CNNs can achieve high levels of accuracy for the classification of petrographic thin section image data. Using transfer learning and repurposing models pre-trained for the classification of the ImageNet can help increase the performance of the CNN models. Unlike most of the previous studies, here we evaluate the performance of pretraining CNNs on different datasets that are visually more similar to petrographic thin section images than the natural images of ImageNet. On average, pretraining ResNet18 and ResNet50 on the proposed datasets did not improve the accuracy significantly compared to training such models with randomly initialized weights (Figure 7 and Table 7). However, hyperparameter tuning shows that pretraining ResNet18 (ResNet50) on the proposed datasets can lead to improvements in accuracy of up to 4% (7%) in the performance of classification of petrographic thin section image data as the “Maximum” column in Table 7 shows. In the experiments described here, we limited the search of hyperparameters, selecting among some of the most common hyperparameters as that was sufficient for the proposed analysis. However, several hyperparameters and training strategies could be evaluated when the objective is directly related to improving models’ accuracy. For example, regularization and decaying the learning rate can help with overfitting. Bello et al. [41] studied how training strategies and hyperparameters can affect the performance of ResNets.

Although MSI vs. MSS dataset was the largest of the proposed datasets, with more than 90k samples in the train set (Table 4), pretraining ResNets on MSI vs. MSS did not improve the classification significantly when compared to HAM10000 (roughly 10k samples in the train set, Table 2) or RawFoot (roughly 25k samples in the train set Table 3) datasets. This is unexpected, as the larger dataset was anticipated to help the model generate more robust filters compared to the smaller datasets. In fact, the results obtained by pretraining

the models on the proposed datasets are similar, except for ResNet50 pretrained on the RawFooT dataset that shows a slightly improved mean average and minimum value. This is likely due to ResNet50 learning more generic filters, such as edge and texture detectors, on RawFooT than in the other proposed datasets.

Results of the experiments show that the widely adopted strategy to use models pretrained on ImageNet is usually very favorable for the classification of petrographic thin-section data, even though the datasets have different distributions. Models pretrained on ImageNet tend to exhibit strong performance with a wide selection of hyperparameters, making them easier to train. We argue that such superior performance is due to two main factors: ImageNet number of samples and variability and general GPU hours employed for hyperparameter tuning. As previously described, the number of samples in ImageNet is one or two orders of magnitude larger than the number of samples of the proposed datasets. Computer vision researchers training models with ImageNet datasets generally have access to more powerful GPUs and higher experience in computer vision tasks, and are thus able to perform more experiments in the search for a good hyperparameter setting than the average geoscientist researcher. Moreover, the weights for models trained on ImageNet are easily available in the most popular deep learning frameworks. Thus, using models pretrained on ImageNet remains a valuable approach for the classification of petrographic thin-section image data. Yosinski et al. [13] showed that transferring features even from distant tasks can be better than training models with randomly initialized weights, although feature transferability reduces as the distance between the primary and secondary task grows. The results presented here are then partially aligned with what was observed before in [13]. The initial expectation was that the proposed datasets would facilitate transferability; however, results show that more accurate models can be obtained by fine-tuning models previously trained on ImageNet.

In the study presented here, we use a five-crop technique to accommodate for images larger than what is generally used as input for CNN models and to generate models that are able to classify petrographic thin-section images based on an average of the image. Extracting patches averaging their predictions is a common strategy widely used to increase performance (e.g., [3,42]). Sultana et al. [43] showed that multiple cropping strategies often outperform single cropping strategies. The five-crop technique is somewhat easier than what was presented in [26], with the advantage of also being incorporated in some deep learning frameworks. Generally, in [26], the thin-section photograph was split into six smaller crops, and each crop was treated as one independent sample. The final classification of the full photograph was given based on the classification of each one of such smaller crops. Such technique allows for a finer control on the number of crops extracted from a single photograph, however converting the final probabilities to a single class for each one of the crops, i.e., converting a continuous value to a categorical value might attenuate small differences that are better accommodated based on a photograph mean average.

Results of the experiments presented here also show that the petrographic thin-section image could correctly be classified, even though the images were taken with two different magnification levels, expressing that CNNs can be trained to be scale invariant. Thin-section images at different magnifications were successfully used before, for example, by Koeshidayatullah et al. [28], who assembled thin-section images with different magnifications for carbonate petrography. Graziani et al. [44] observed that CNN trained on natural images must achieve scale invariance due to viewpoint variations, and they studied this property in models trained on ImageNet. It is helpful to know that CNN models did not struggle with such variations in the characteristics of the data, as this can be used to facilitate the assemblage of larger petrographic thin-section datasets. That said, we anticipate that the variable magnification strategy might fail for some objectives. For example, the magnification level necessary for a sandstone modal analysis is very different to the magnification level needed to identify types of clays in the same sample, which indicates the same dataset likely needs labels for different levels of magnification. We believe multiple magnification levels can be used if the general characteristics of the thin

section leading to the defined classification (e.g., mineralogical composition and texture) can be observed at different scales.

In general, the results presented for the best performing ResNet18s pretrained on different datasets show comparable performances. Moreover, [26] showed that the accuracy of such models tends to decrease when applied to classify thin section data processed by different laboratories. Larger datasets would be helpful to understand what characteristics of the images make the models fail the classification. Although petrographers often photograph locations of the thin section on which the grains/crystals—or fossils—exhibit different behaviors, rather than the average behavior used for the thin-section description and microfacies classification, we believe the creation of a larger petrographic thin-section dataset can be helpful for many geoscientists. With larger models, our community would be able to pretrain robust petrographic classifiers and make weights available for fine-tuning on specific formations. However, the results of this paper, specifically the transfer learning analysis performed with the MSI vs. MSS dataset, indicate that such a petrographic dataset should likely have hundreds of thousands of samples.

5. Conclusions

The number of studies using models pretrained on ImageNet, and the fine-tuning of such models for the classification of thin-section data, is increasing. Despite the difference in characteristics of the samples from ImageNet, a dataset with a wide range of classes and image characteristics, with the samples of petrographic thin-section image data, the fine-tuning strategy proved itself reliable and generated models that achieved the best performance in experiments conducted with thin sections from five different wells from the Sycamore formation. Although expected to facilitate fine-tuning, on average, pretraining CNN models on datasets that are visually more similar to petrographic thin-section images did not show significant improvements than training CNN models with randomly initialized weights. Larger petrographic datasets should be helpful for the creation of more robust CNN models.

Author Contributions: Conceptualization, R.P.d.L. and D.D.; methodology, R.P.d.L.; software, R.P.d.L. and D.D.; validation, R.P.d.L. and D.D.; formal analysis, R.P.d.L.; investigation, R.P.d.L. and D.D.; resources, R.P.d.L. and D.D.; data curation, D.D.; writing—original draft preparation, R.P.d.L.; writing—review and editing, R.P.d.L. and D.D.; visualization, R.P.d.L.; supervision, R.P.d.L.; project administration, R.P.d.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The petrographic thin-section data presented in this study can be accessed through <https://zenodo.org/record/5071794#.YONrUuj0IPY> (accessed on 7 August 2021). The HAM10000 can be accessed using the following link: <https://doi.org/10.7910/DVN/DBW86T> (accessed on 7 August 2021). The dataset named in this manuscript as “MSI vs. MSS” can be accessed at https://zenodo.org/record/2530835#.YL_eOPn0IPb (accessed on 7 August 2021). This work uses only the colorectal cancer images. The RawFoot dataset can be accessed at <http://projects.ivl.disco.unimib.it/minisites/rawfoot/> (accessed on 7 August 2021).

Acknowledgments: We thank our colleagues at the Attribute Assisted Seismic Processing and Interpretation consortium at the University of Oklahoma for the discussions that led us to investigate this problem, especially Kurt Marfurt. We thank Katie Welch for fruitful discussions and for help reviewing the manuscript. We thank the reviewers and editors for their valuable comments that greatly helped us improve the quality of the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

This appendix provides extra details about the pretraining step. Table A1 shows the hyperparameter tuning for the different datasets used to pretrain the models. Table A2 shows the accuracies for each one of the sets and for each one of the models presented in Table A1. Table A2 also shows the approximate execution time (training and evaluation),

with identification of the hardware for each one of the experiments. As described in the main text, the model stops training after five epochs without improvements on the validation set. ImageNet training parameters are not shown, as the pretrained models are downloaded directly from [33].

Table A1. Hyperparameters tested for pretraining models.

Model Name	Dataset	# of Layers	Batch Size	Optimizer	Learning Rate
ResNet18-M-128-A3	MSI vs. MSS	18	128	Adam	1×10^{-3}
ResNet18-M-512-A3	MSI vs. MSS	18	512	Adam	1×10^{-3}
ResNet18-M-1024-A3	MSI vs. MSS	18	1024	Adam	1×10^{-3}
ResNet50-M-512-A3	MSI vs. MSS	50	512	Adam	1×10^{-3}
ResNet50-M-1024-A3	MSI vs. MSS	50	1024	Adam	1×10^{-3}
ResNet18-R-A2	RawFooT	18	128	Adam	1×10^{-2}
ResNet18-R-A3	RawFooT	18	128	Adam	1×10^{-3}
ResNet18-R-A4	RawFooT	18	128	Adam	1×10^{-4}
ResNet50-R-A2	RawFooT	50	128	Adam	1×10^{-2}
ResNet50-R-A3	RawFooT	50	128	Adam	1×10^{-3}
ResNet50-R-A4	RawFooT	50	128	Adam	1×10^{-4}
ResNet18-H-64-A3	HAM10000	18	64	Adam	1×10^{-3}
ResNet18-H-128-A3	HAM10000	18	128	Adam	1×10^{-3}
ResNet18-H-128-R3	HAM10000	18	128	RMSprop	1×10^{-3}
ResNet18-H-256-A3	HAM10000	18	256	Adam	1×10^{-3}
ResNet50-H-64-A3	HAM10000	50	64	Adam	1×10^{-3}
ResNet50-H-128-A3	HAM10000	50	128	Adam	1×10^{-3}
ResNet50-H-128-A4	HAM10000	50	128	Adam	5×10^{-4}
ResNet50-H-256-A3	HAM10000	50	256	Adam	1×10^{-3}

Table A2. Accuracy obtained for the primary task and other training information. The bold indicates best performing in group.

Model Name	Training	Validation	Test	GPU	Elapsed Time ¹	Epochs
ResNet18-M-128-A3	0.95	0.92	0.70	GTX 1050	11.5 h	34
ResNet18-M-512-A3	0.94	0.91	0.71	GTX 1050	11.5 h	33
ResNet18-M-1024-A3	0.93	0.91	0.70	GTX 1050	9.5 h	28
ResNet50-M-512-A3	0.92	0.91	0.68	GTX 1050	24 h	32
ResNet50-M-1024-A3	0.91	0.90	0.71	GTX 1050	20 h	27
ResNet18-R-A2	0.97	0.92	0.88	Quadro K1200	12 h	38
ResNet18-R-A3	0.98	0.96	0.88	Quadro K1200	8 h	25
ResNet18-R-A4	0.97	0.92	0.88	Quadro K1200	11.5 h	38
ResNet50-R-A2	0.85	0.79	–	Quadro K1200	6 h	9
ResNet50-R-A3	0.89	0.87	0.73	Quadro K1200	7 h	10
ResNet50-R-A4	0.90	0.83	–	Quadro K1200	9 h	14
ResNet18-H-64-A3	0.78	0.75	0.73	GTX 1050	1 h	16
ResNet18-H-128-A3	0.79	0.77	0.76	GTX 1050	1.5 h	21
ResNet18-H-128-R3	0.80	0.76	0.75	GTX 1050	2 h	30
ResNet18-H-256-A3	0.77	0.73	0.75	GTX 1050	1 h	15
ResNet50-H-64-A3	0.78	0.75	0.73	GTX 1050	2 h	29
ResNet50-H-128-A3	0.78	0.74	0.74	GTX 1050	2 h	28
ResNet50-H-128-A4	0.77	0.76	0.74	GTX 1050	1.5 h	16
ResNet50-H-256-A3	0.80	0.76	0.75	GTX 1050	2.5 h	32

¹ The elapsed time includes all training and testing, but performance might vary according to secondary tasks running concomitantly.

Appendix B

Hyperparameter search for fine-tuned models discussed in Section 3.

Table A3. Hyperparameter search for fine-tuned models.

Model	Pretrained	Batch Size	Batch Size (Accumulated)	Optimizer	Learning Rate	Patience	Last Epoch	Train Accuracy	Validation Accuracy	Test Accuracy
ResNet18	RawFooT	4	8	Adam	1.00×10^{-3}	10	25	0.73	0.64	0.65
ResNet18	RawFooT	4	16	Adam	1.00×10^{-3}	10	49	0.82	0.72	0.85
ResNet18	RawFooT	4	32	Adam	1.00×10^{-4}	10	26	0.84	0.79	0.81
ResNet18	RawFooT	4	32	Adam	1.00×10^{-3}	10	32	0.77	0.78	0.80
ResNet18	RawFooT	4	64	Adam	1.00×10^{-4}	10	35	0.87	0.69	0.80
ResNet18	RawFooT	4	64	RMSprop	1.00×10^{-3}	10	48	0.73	0.61	0.79
ResNet18	RawFooT	4	64	Adam	1.00×10^{-3}	10	72	0.95	0.85	0.93
ResNet18	RawFooT	4	128	Adam	1.00×10^{-3}	10	37	0.85	0.79	0.82
ResNet18	MSI vs. MSS	16	32	Adam	1.00×10^{-4}	5	26	0.91	0.78	0.84
ResNet18	MSI vs. MSS	16	64	Adam	1.00×10^{-3}	5	25	0.91	0.79	0.77
ResNet18	MSI vs. MSS	16	64	Adam	5.00×10^{-5}	5	21	0.87	0.74	0.85
ResNet18	MSI vs. MSS	16	64	RMSprop	1.00×10^{-4}	5	21	0.82	0.56	0.73
ResNet18	MSI vs. MSS	16	64	Adam	1.00×10^{-4}	5	25	0.92	0.78	0.91
ResNet18	MSI vs. MSS	16	128	Adam	1.00×10^{-4}	5	23	0.88	0.75	0.85
ResNet18	ImageNet	16	32	Adam	1.00×10^{-4}	5	14	1.00	0.89	0.94
ResNet18	ImageNet	16	64	RMSprop	5.00×10^{-5}	5	20	1.00	0.92	0.94
ResNet18	ImageNet	16	64	RMSprop	1.00×10^{-4}	5	19	0.99	0.82	0.95
ResNet18	ImageNet	16	64	Adam	5.00×10^{-5}	5	18	0.98	0.93	0.94
ResNet18	ImageNet	16	64	Adam	1.00×10^{-4}	5	15	1.00	0.91	0.95
ResNet18	ImageNet	16	128	Adam	1.00×10^{-4}	5	18	0.99	0.93	0.94
ResNet18	HAM10000	16	32	Adam	5.00×10^{-5}	5	24	0.91	0.77	0.86
ResNet18	HAM10000	16	64	RMSprop	5.00×10^{-5}	5	14	0.82	0.52	0.68
ResNet18	HAM10000	16	64	Adam	1.00×10^{-5}	5	52	0.85	0.79	0.86
ResNet18	HAM10000	16	64	Adam	5.00×10^{-5}	5	37	0.95	0.86	0.93
ResNet18	HAM10000	16	64	Adam	1.00×10^{-4}	5	18	0.89	0.57	0.81
ResNet18	HAM10000	16	128	Adam	5.00×10^{-5}	5	32	0.89	0.78	0.86
ResNet50	RawFooT	4	8	Adam	1.00×10^{-3}	10	58	0.79	0.75	0.84
ResNet50	RawFooT	4	16	Adam	1.00×10^{-3}	10	30	0.75	0.76	0.79
ResNet50	RawFooT	4	32	Adam	1.00×10^{-4}	10	43	0.82	0.71	0.84
ResNet50	RawFooT	4	32	Adam	1.00×10^{-3}	10	50	0.87	0.69	0.73
ResNet50	RawFooT	4	64	RMSprop	1.00×10^{-3}	10	34	0.66	0.48	0.55
ResNet50	RawFooT	4	64	Adam	1.00×10^{-3}	10	39	0.80	0.77	0.89
ResNet50	MSI vs. MSS	4	8	Adam	1.00×10^{-3}	10	53	0.76	0.74	0.65
ResNet50	MSI vs. MSS	4	16	Adam	1.00×10^{-3}	10	35	0.72	0.48	0.82
ResNet50	MSI vs. MSS	4	32	Adam	5.00×10^{-5}	10	25	0.63	0.56	0.69
ResNet50	MSI vs. MSS	4	32	Adam	1.00×10^{-4}	10	21	0.73	0.64	0.66
ResNet50	MSI vs. MSS	4	64	RMSprop	1.00×10^{-3}	10	32	0.60	0.55	0.54
ResNet50	MSI vs. MSS	4	64	Adam	1.00×10^{-3}	10	40	0.83	0.75	0.81
ResNet50	MSI vs. MSS	4	128	Adam	1.00×10^{-3}	10	57	0.92	0.76	0.77
ResNet50	ImageNet	4	8	Adam	1.00×10^{-3}	10	14	0.78	0.70	0.82
ResNet50	ImageNet	4	16	Adam	1.00×10^{-3}	10	14	0.79	0.77	0.85
ResNet50	ImageNet	4	32	RMSprop	1.00×10^{-4}	10	28	1.00	0.91	0.90
ResNet50	ImageNet	4	32	Adam	5.00×10^{-5}	10	31	1.00	0.90	0.84
ResNet50	ImageNet	4	32	Adam	1.00×10^{-4}	10	37	1.00	0.94	0.90
ResNet50	ImageNet	4	64	Adam	1.00×10^{-3}	10	33	0.96	0.85	0.84
ResNet50	ImageNet	4	128	Adam	1.00×10^{-3}	10	27	0.98	0.75	0.76
ResNet50	HAM10000	4	8	Adam	1.00×10^{-3}	10	34	0.69	0.74	0.74
ResNet50	HAM10000	4	16	Adam	1.00×10^{-3}	10	29	0.72	0.61	0.74
ResNet50	HAM10000	4	32	Adam	5.00×10^{-5}	10	24	0.65	0.53	0.68
ResNet50	HAM10000	4	32	Adam	1.00×10^{-4}	10	22	0.74	0.64	0.68
ResNet50	HAM10000	4	64	Adam	1.00×10^{-4}	10	31	0.79	0.68	0.66
ResNet50	HAM10000	4	64	RMSprop	1.00×10^{-3}	10	28	0.65	0.59	0.49
ResNet50	HAM10000	4	64	Adam	1.00×10^{-3}	10	54	0.92	0.79	0.89
ResNet50	HAM10000	4	128	Adam	1.00×10^{-3}	10	39	0.89	0.85	0.77

Appendix C

This appendix provides extra performance metrics for the test set for the best performing fine-tuned ResNet18 models when pretrained on the ImageNet, HAM10000, and MSI vs. MSS datasets, as well as the Baseline ResNet18, as described in Section 3, specifically in relation to the results shown in Figure 10.

Table A4. Classification report for ResNet18—baseline. The overall accuracy is 0.89.

Class	Precision	Recall	F1-Score	Support
AMdst	0.77	1.00	0.87	20
BMdst	0.93	0.70	0.80	20
MCSt	0.95	0.95	0.95	20
MCcSt	0.95	0.90	0.92	20
macro average	0.90	0.89	0.89	80
weighted average	0.90	0.89	0.89	80

Table A5. Classification report for ResNet18 pretrained on ImageNet. The overall accuracy is 0.95.

Class	Precision	Recall	F1-Score	Support
AMdst	0.95	0.90	0.92	20
BMdst	0.90	0.90	0.90	20
MCSt	1.00	1.00	1.00	20
MCcSt	0.95	1.00	0.98	20
macro average	0.95	0.95	0.95	80
weighted average	0.95	0.95	0.95	80

Table A6. Classification report for ResNet18 pretrained on HAM10000. The overall accuracy is 0.93.

Class	Precision	Recall	F1-Score	Support
AMdst	0.90	0.95	0.93	20
BMdst	1.00	0.85	0.92	20
MCSt	0.90	0.95	0.93	20
MCcSt	0.90	0.95	0.93	20
macro average	0.93	0.93	0.92	80
weighted average	0.93	0.93	0.92	80

Table A7. Classification report for ResNet18 pretrained on RawFoot. The overall accuracy is 0.93.

Class	Precision	Recall	F1-Score	Support
AMdst	1.00	0.85	0.92	20
BMdst	0.90	0.95	0.93	20
MCSt	0.95	0.90	0.92	20
MCcSt	0.87	1.00	0.93	20
macro average	0.93	0.92	0.92	80
weighted average	0.93	0.93	0.92	80

Table A8. Classification report for ResNet18 pretrained on MSI vs. MSS. The overall accuracy is 0.91.

Class	Precision	Recall	F1-Score	Support
AMdst	0.80	1.00	0.89	20
BMdst	1.00	0.70	0.82	20
MCSt	1.00	0.95	0.97	20
MCcSt	0.91	1.00	0.95	20
macro average	0.93	0.91	0.91	80
weighted average	0.93	0.91	0.91	80

References

1. Fukushima, K. Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. *Biol. Cybern.* **1980**, *36*, 193–202. [[CrossRef](#)]
2. LeCun, Y.; Boser, B.E.; Denker, J.S.; Henderson, D.; Howard, R.E.; Hubbard, W.E.; Jackel, L.D. Handwritten Digit Recognition with a Back-Propagation Network. In *Advances in Neural Information Processing Systems 2*; Touretzky, D.S., Ed.; Morgan-Kaufmann: Denver, CA, USA, 1990; pp. 396–404.

3. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
4. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
5. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
6. Deng, J.; Dong, W.; Socher, R.; Li, L.; Li, K.; Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
7. LeCun, Y. The MNIST Database of Handwritten Digits. 1998. Available online: <http://yann.lecun.com/exdb/mnist/> (accessed on 7 August 2021).
8. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. *Microsoft COCO: Common Objects in Context BT—Computer Vision—ECCV 2014*; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 740–755.
9. Krizhevsky, A. *Learning Multiple Layers of Features from Tiny Images, Technical Report TR-2009*; University of Toronto: Toronto, ON, Canada, 2009.
10. Pan, S.J.; Yang, Q. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]
11. Bengio, Y. Deep Learning of Representations for Unsupervised and Transfer Learning. In *Proceedings of the ICML Workshop on Unsupervised and Transfer Learning*; Guyon, I., Dror, G., Lemaire, V., Taylor, G., Silver, D., Eds.; PMLR: Bellevue, WA, USA, 2012; Volume 27, pp. 17–36.
12. Razavian, A.S.; Azizpour, H.; Sullivan, J.; Carlsson, S. CNN Features Off-the-Shelf: An Astounding Baseline for Recognition. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops, Columbus, OH, USA, 23–28 June 2014; pp. 512–519.
13. Yosinski, J.; Clune, J.; Bengio, Y.; Lipson, H. How Transferable Are Features in Deep Neural Networks? *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 3320–3328.
14. Simonyan, K.; Vedaldi, A.; Zisserman, A. Deep inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. In Proceedings of the International Workshop at International Conference on Learning Representations, Banff, AB, Canada, 14–16 April 2014.
15. Olah, C.; Mordvintsev, A.; Schubert, L. Feature Visualization. *Distill* **2017**, *2*, e7. [CrossRef]
16. Olah, C.; Satyanarayan, A.; Johnson, I.; Carter, S.; Schubert, L.; Ye, K.; Mordvintsev, A. The Building Blocks of Interpretability. *Distill* **2018**, *3*, e10. [CrossRef]
17. Carter, S.; Armstrong, Z.; Schubert, L.; Johnson, I.; Olah, C. Activation Atlas. *Distill* **2019**, *4*, e15. [CrossRef]
18. Hu, F.; Xia, G.-S.; Hu, J.; Zhang, L. Transferring Deep Convolutional Neural Networks for the Scene Classification of High-Resolution Remote Sensing Imagery. *Remote Sens.* **2015**, *7*, 14680–14707. [CrossRef]
19. Zamir, A.R.; Sax, A.; Shen, W.; Guibas, L.J.; Malik, J.; Savarese, S. Taskonomy: Disentangling Task Transfer Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018.
20. Norouzzadeh, M.S.; Nguyen, A.; Kosmala, M.; Swanson, A.; Palmer, M.S.; Packer, C.; Clune, J. Automatically Identifying, Counting, and Describing Wild Animals in Camera-Trap Images with Deep Learning. *Proc. Natl. Acad. Sci. USA* **2018**, *115*, E5716–E5725. [CrossRef]
21. Tschandl, P.; Rosendahl, C.; Kittler, H. The HAM10000 Dataset, a Large Collection of Multi-Source Dermatoscopic Images of Common Pigmented Skin Lesions. *Sci. Data* **2018**, *5*, 180161. [CrossRef]
22. Kather, J.N.; Pearson, A.T.; Halama, N.; Jäger, D.; Krause, J.; Loosen, S.H.; Marx, A.; Boor, P.; Tacke, F.; Neumann, U.P.; et al. Deep Learning Can Predict Microsatellite Instability Directly from Histology in Gastrointestinal Cancer. *Nat. Med.* **2019**, *25*, 1054–1056. [CrossRef]
23. Pires de Lima, R.; Marfurt, K. Convolutional Neural Network for Remote-Sensing Scene Classification: Transfer Learning Analysis. *Remote Sens.* **2019**, *12*, 86. [CrossRef]
24. Pires de Lima, R.; Suriamin, F.; Marfurt, K.J.; Pranter, M.J. Convolutional Neural Networks as Aid in Core Lithofacies Classification. *Interpretation* **2019**, *7*, SF27–SF40. [CrossRef]
25. Baraboshkin, E.E.; Ismailova, L.S.; Orlov, D.M.; Zhukovskaya, E.A.; Kalmykov, G.A.; Khotylev, O.V.; Baraboshkin, E.Y.; Koroteev, D.A. Deep Convolutions for In-Depth Automated Rock Typing. *Comput. Geosci.* **2020**, *135*, 104330. [CrossRef]
26. Pires de Lima, R.; Duarte, D.; Nicholson, C.; Slatt, R.; Marfurt, K.J. Petrographic Microfacies Classification with Deep Convolutional Neural Networks. *Comput. Geosci.* **2020**, *142*, 104481. [CrossRef]
27. Liu, X.; Song, H. Automatic Identification of Fossils and Abiotic Grains during Carbonate Microfacies Analysis Using Deep Convolutional Neural Networks. *Sediment. Geol.* **2020**, *410*, 105790. [CrossRef]
28. Koeshidayatullah, A.; Morsilli, M.; Lehrmann, D.J.; Al-Ramadan, K.; Payne, J.L. Fully Automated Carbonate Petrography Using Deep Convolutional Neural Networks. *Mar. Pet. Geol.* **2020**, *122*, 104687. [CrossRef]
29. Ma, H.; Han, G.; Peng, L.; Zhu, L.; Shu, J. Rock Thin Sections Identification Based on Improved Squeeze-and-Excitation Networks Model. *Comput. Geosci.* **2021**, *152*, 104780. [CrossRef]
30. Cusano, C.; Napolitano, P.; Schettini, R. Evaluating Color Texture Descriptors under Large Variations of Controlled Lighting Conditions. *J. Opt. Soc. Am. A JOSAA* **2016**, *33*, 17–30. [CrossRef] [PubMed]

31. Limare, N.; Lisani, J.-L.; Morel, J.-M.; Petro, A.B.; Sbert, C. Simplest Color Balance. *Image Process. Line* **2011**, *1*, 297–315. [[CrossRef](#)]
32. Bianco, S.; Cusano, C.; Napoletano, P.; Schettini, R.; Bianco, S.; Cusano, C.; Napoletano, P.; Schettini, R. Improving CNN-Based Texture Classification by Color Balancing. *J. Imaging* **2017**, *3*, 33. [[CrossRef](#)]
33. 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. *Adv. Neural Inf. Proces. Syst.* **2019**, *32*, 8024–8035.
34. 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, 27–30 June 2016; pp. 770–778.
35. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the ICLR, San Diego, CA, USA, 22 December 2014.
36. Tieleman, T.; Hinton, G. Lecture 6.5—RmsProp: Divide the Gradient by a Running Average of Its Recent Magnitude; 2012. Available online: https://www.youtube.com/watch?v=Sj48OZ_qIrc (accessed on 7 August 2021).
37. WA PyTorch Lightning. GitHub. 2019, 3. Available online: <https://github.com/PyTorchLightning/pytorch-lightning> (accessed on 7 August 2021).
38. Biewald, L. Experiment Tracking with Weights and Biases. Available online: <https://www.wandb.com/> (accessed on 7 August 2021).
39. Fawcett, T. An Introduction to ROC Analysis. *Pattern Recognit. Lett.* **2006**, *27*, 861–874. [[CrossRef](#)]
40. Powers, D.M.W. Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness and Correlation. *Int. J. Mach. Learn. Technol.* **2011**, *2*, 37–63.
41. Bello, I.; Fedus, W.; Du, X.; Cubuk, E.D.; Srinivas, A.; Lin, T.-Y.; Shlens, J.; Zoph, B. Revisiting ResNets: Improved Training and Scaling Strategies. *arXiv* **2021**, arXiv:2103.07579.
42. Liu, Y.; Yin, B.; Yu, J.; Wang, Z. Image Classification Based on Convolutional Neural Networks with Cross-Level Strategy. *Multimed Tools Appl.* **2017**, *76*, 11065–11079. [[CrossRef](#)]
43. Sultana, F.; Sufian, A.; Dutta, P. Advancements in Image Classification Using Convolutional Neural Network. In Proceedings of the 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), Kolkata, India, 22–23 November 2018; pp. 122–129.
44. Graziani, M.; Lompech, T.; Müller, H.; Depeursinge, A.; Andrearczyk, V. On the Scale Invariance in State of the Art CNNs Trained on ImageNet. *Mach. Learn. Knowl. Extr.* **2021**, *3*, 19. [[CrossRef](#)]