Towards Large-Scale Training of Pathology Foundation Models

kaiko.ai, Nanne Aben, Edwin D. de Jong, Ioannis Gatopoulos, Nicolas Känzig, Mikhail Karasikov, Axel Lagré, Roman Moser, Joost van Doorn, Fei Tang[†]
kaiko.ai

***Abstract*— Driven by the recent advances in deep learning methods and, in particular, by the development of modern self-supervised learning algorithms, increased interest and efforts have been devoted to build foundation models (FMs) for medical images. In this work, we present our scalable training pipeline for large pathology imaging data, and a comprehensive analysis of various hyperparameter choices and training techniques for building pathology FMs. We release and make publicly available the first batch of our pathology FMs[1] trained on open-access TCGA whole slide images, a commonly used collection of pathology images. The experimental evaluation shows that our models reach state-of-the-art performance on various patch-level downstream tasks, ranging from breast cancer subtyping to colorectal nuclear segmentation. Finally, to unify the evaluation approaches used in the field and to simplify future comparisons of different FMs, we present an open-source framework[2] designed for the consistent evaluation of pathology FMs across various downstream tasks.**

I. INTRODUCTION

Pathology images contain a wealth of information about patient health status, and deep learning-based methods have become increasingly capable of automatically extracting vital information about a patient's condition from these images [1]–[8]. While expert human pathologists are able to detect certain subtle patterns from Whole Slide Images (WSIs), such as Micro-Satellite Instability [9]–[11], AI methods are increasingly able to detect ever subtler patterns that even expert pathologists are unable to detect. For example, Machine Learning (ML) models have been trained to predict molecular biomarkers [12, 13] and RNA expression levels from pathology images [14].

One of the clearest findings from the past decade of machine learning research is that increasing training dataset size and variety is a primary driver of increased model performance. This trend can be traced from AlexNet [15] and the subsequent development of convolutional neural network architectures enabled by ImageNet, e.g. [16]–[18]. Self-Supervised Learning (SSL) then facilitated exploiting seemingly unlimited further increases in dataset magnitude, up to the internet-scale datasets that are used to train current Large Language Models (LLMs) such as GPT-4 [19].

Hospitals worldwide are collecting data at an unprecedented scale. With this, new challenges and opportunities arise: How can AI methods best be employed to extract and make use of the wealth of medically relevant information contained in large-scale medical imaging datasets?

The work of Campanella et al. [1] was pivotal in pathology machine learning for demonstrating that WSI-level supervision (weak labels) is sufficient to effectively train high-quality pathology models given sufficiently large datasets (approx. 44k WSIs), thus obviating the need for painstaking and time-consuming annotation efforts of pathologists.

That early work and the simultaneous advancement of Self-Supervised Learning (SSL) [20]–[25] paved the way for the recent series of pathology SSL models trained on increasingly large datasets. To the best of our knowledge, the earliest work that applied SSL to pathology images is "Self-supervision closes the gap between weak and strong supervision in histology" [26]. HIPT [27] then used a hierarchical setup where Vision Transformers (ViTs) are trained with the self-supervised DINO [24] algorithm on 10,678 FFPE (formalin-fixed, paraffin-embedded) WSIs from TCGA [28]. Lunit [29] trained on 19M patches from the full set of 21k TCGA WSIs. Phikon (Owkin) [30] is an iBOT ViT-Base model (80M parameters) trained on over 40M patches from 6k WSIs.

Chen et al. [31] trained the UNI model, and it is the first work to use an order of magnitude more WSIs than TCGA. The UNI model was trained on over 100M patches from over 100k diagnostic H&E WSIs across 20 tissue types and evaluated on 33 representative clinical pathology tasks of varying difficulty. Campanella et al. [32] claimed to have collected the largest pathology dataset at the time (Oct 2023) and trained on 3B patches from over 423k WSIs, comparing pre-training of vision transformer models using a masked autoencoder (MAE) vs. DINO. However, these numbers are surpassed remarkably by the 632M parameter Virchow model trained on 1.5M WSIs [33].

In collaboration with the Netherlands Cancer Institute (NKI-AvL), we are building one of the first large-scale pathology FM on a European cohort. NKI-AvL has already accumulated around 1M WSIs, and our dataset can be as large as 10M WSIs when we expand to multiple centers. Working with data at this scale brings both infrastructure and architectural challenges, which we address in this work.

Here, we present our scalable pipeline for training and evaluating foundation models (FMs) on large pathology imaging data. For seamless training, we developed Online Patching, a technique for high-throughput loading of

[†]Correspondence E-mail: fei@kaiko.ai. Other authors are ordered alphabetically.

[1]https://github.com/kaiko-ai/towards_large_pathology_fms

[2]https://github.com/kaiko-ai/eva

TABLE I: Linear probing evaluation of FMs on patch-level downstream datasets. We compare the performance of a randomly initialized model (ViT-S16 *(rand.)*), FMs trained on ImageNet (above the dashed line), and the pathology FMs (below the dashed line). The evaluation was performed with *eva*. For BACH, CRC, MHIST, PCam, and TP53, the numbers represent the balanced accuracy averaged over 5 linear probing runs (the respective standard deviations can be found in the extended version of Table VIII). For CoNSeP, we report the DICE score on the foreground pixels. The best results are highlighted in bold. (*) For Virchow, the model weights were not publicly available. The values were taken from [33], where they were computed in a different setup and may not be directly comparable to the other values. (**) For TP53 and CoNSeP, the evaluation was done separately but will soon be supported in *eva* as well.

Model	Training data	BACH	CRC	MHIST	PCam	TP53**	CoNSeP**
ViT-S16 *(rand.)*	None	0.410	0.617	0.501	0.728	0.500	0.583
DINO ViT-S16 [24]	ImageNet	0.695	0.935	0.831	0.849	0.519	0.611
DINO ViT-B8 [24]	ImageNet	0.710	0.939	0.814	0.856	0.548	0.710
Lunit [29]	TCGA (21k WSIs)	0.801	0.934	0.768	0.895	0.571	0.654
Phikon [30]	TCGA (6k WSIs)	0.725	0.935	0.777	0.915	0.630	0.666
DINO ViT-S16 *(ours)*	TCGA (29k WSIs)	0.797	0.943	0.828	0.893	0.633	0.649
DINO ViT-S8 *(ours)*	TCGA (29k WSIs)	0.834	0.946	**0.832**	0.887	0.621	0.724
DINO ViT-B16 *(ours)*	TCGA (29k WSIs)	0.810	**0.960**	0.826	0.898	0.651	0.658
DINO ViT-B8 *(ours)*	TCGA (29k WSIs)	0.865	0.956	0.809	**0.921**	**0.659**	**0.741**
DINOv2 ViT-L14 *(ours)*	TCGA (29k WSIs)	**0.870**	0.930	0.809	0.898	0.656	0.679
Virchow [33]	Private (1.5M WSIs)	n/a	0.962*	0.830*	0.933*	n/a	n/a

arbitrary image patches cropped from large WSIs residing in blob storage (described in Section IV-B). With this, we trained vision transformers of various sizes using the DINO and DINOv2 SSL algorithms on TCGA WSIs, an open-access pathology image dataset, which has been widely used in the community for training pathology FMs. We present our best-to-date FMs trained on TCGA and compare them to state of the art. The evaluation shows that our FMs perform on par or better than the existing state-of-the-art FMs on most downstream tasks (Section II-A).

We also present an experimental study of various hyperparameter and design choices for training pathology FMs, such as the model initialization strategy (Section II-B), effects of mixing different magnifications (Section II-C), and effects of data sizes (Section II-D.1), which may be of interest to other practitioners in the field of medical machine learning and computational pathology to guide the future development of pathology FMs.

To aid the evaluation of FMs, we introduce a new unsupervised metric that can be used to compare models of different sizes and show that it correlates well with downstream performance and, hence, can be a valuable addition to supervised metrics (Section IV-D.1 and C).

Lastly, we developed *eva* (described in Section IV-D.2), an open-source framework for evaluating FMs on clinically relevant downstream tasks in a straightforward and unified way. We will be extending *eva* with more downstream tasks in the future and invite other practitioners in the field to contribute more downstream tasks to *eva* in order to build a standardized evaluation workflow and to ensure the evaluation results are comparable across different studies.

II. Results and Discussion

A. Training state-of-the-art FMs with online patching

In most existing work, a patch dataset is typically pre-constructed from WSIs offline before training, which results in a fixed set of patches. Moreover, for larger datasets, this approach becomes inefficient for the following reasons: 1) For every new patch extraction strategy, the dataset has to be re-created from scratch, which is costly and time-consuming. 2) In addition, every time a patch dataset is created, it requires a large storage space overhead, which makes dynamic patch sampling and experiments with sampling strategies practically impossible.

To address these issues, we developed *Online Patching* (see Methods, Section IV-B), a method that allows for the online high-throughput extraction of patches of arbitrary size and resolution from WSIs residing in blob storage. Not only does online patching improve data processing efficiency, but it also introduces a key difference to the offline patching approach: the patches are created dynamically. As a result, dynamic patch sampling strategies can be seamlessly incorporated into the training procedure.

Even for a single WSI with $10^5 \times 10^5$ pixels, there can be up to 10^{10} distinct sampled patches. With Online Patching, almost every sampled patch is new because it is sampled at a random position. Thus, the number of distinct patches our models have seen during training is typically much larger than for other models trained with offline patching. Moreover, this number grows with the number of training epochs. On the other hand, many of the patches, despite being unique, do overlap with many other neighboring patches. It is currently unclear how the number of unique patches and their similarity impact the performance of the trained FM. In our experiments, we implicitly show that sampling all patches at random coordinates does not have a negative impact on the performance of the trained FM, and we leave a more comprehensive analysis for future work.

Using online patching, we trained several vision transformer models of different sizes using both the DINO and DINOv2 algorithms on the whole set of 29k open-access Flash-Frozen (FF) and Formalin-Fixed Paraffin-Embedded (FFPE) diagnostic tissue slides from TCGA. We evaluated the resulting models and compared them to the existing

TABLE II: Magnification ablation study: evaluating downstream benchmark performance through different patch magnifications in pre-training phase. All results were generated using *eva*. All runs have, on average, a standard deviation of (±0.002). (*) The images from BACH were downsampled from an mpp of 0.42 µm/px (20×) to 2.88 µm/px (3.47×).

Downstream task	40×	20×	10×	5×	{20, 40}×	{5, 10, 20}×	{5, 10, 20, 40}×
BACH (3.47×)*	0.639	0.685	0.659	0.679	0.689	0.683	**0.753**
CRC (20×)	0.935	0.945	0.939	0.927	0.942	0.944	**0.947**
MHIST (5×)	0.744	0.746	0.648	0.710	0.746	0.744	**0.771**
PCam/val (10×)	0.879	**0.898**	0.873	0.859	0.887	0.870	0.887
PCam/test (10×)	0.824	0.874	0.834	0.820	0.874	0.858	**0.876**

state-of-the-art models.

For training our FMs, we mainly followed the original recipes from DINO [24] and DINOv2 [25]. More specifically, we deviate from the original recipes in the following: 1) We start from the models pre-trained on ImageNet published in [24] and [25], respectively; 2) Our FMs are trained on patches extracted from TCGA WSIs at different magnification levels; 3) We use fewer GPUs and, consequently, a smaller global batch size. (For example, for ViT-B8, the original config from the DINO repository specified 176 GPUs, while we only use 8 GPUs.) We used the linear and square root scaling law for the learning rate in DINO and DINOv2, respectively. For more details, see Section IV-C.

The overview and the performance of our best models is presented in Table I. The ViT-S16 model we trained is comparable to the state-of-the-art models of similar size on all considered downstream tasks. Notably, on BACH, CRC, and MHIST, it achieves higher accuracy than the larger Phikon model, which is a ViT-B16 model trained with iBOT.

Similar to what is reported in DINO [24], reducing the patch size used in the vision transformer considerably improves the model performance. This is especially prominent in the segmentation task, where the top two performing models are ViT-B8 and ViT-S8 surpassing the next in line, ViT-L14, by a large margin, despite the fact that the ViT-L14 has more parameters and was trained with DINOv2 with the additional patch-level objectives.

On the other hand, unlike what was observed in DINO [24], scaling up the model size has shown on TCGA a limited impact on the performance (e.g., the performance on PCam changed from 0.893 for ViT-S16 to 0.921 for ViT-B8, and to 0.898 for ViT-L14.). We hypothesize that the impact could be limited for two possible reasons. First, the performance on some downstream tasks might have reached its maximum, and hence, we can no longer observe any differences between the FMs. Secondly, the larger models might have not reached their full capacities, either because the effective data size of the TCGA images is too small due to the limited diversity or because the larger models were not trained long enough. Virchow [33], for example, achieved superior performance on PCam and CRC with a ViT-H/14 trained on almost two orders of magnitude more WSIs.

B. Starting from FMs pre-trained on ImageNet yields faster convergence

We seek to leverage the advantages of publicly available pre-trained models in conjunction with domain-aligned pre-training. To investigate this, we assess the impact of initializing from models pre-trained on ImageNet compared to starting from scratch. We use the DINO ViT-S16 architecture with default parameters and train it for 120 epochs. The results are shown in Fig. 1.

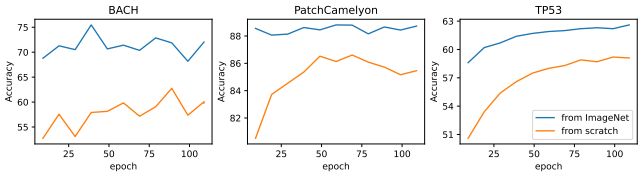

Fig. 1: Validation performance over the course of training a ViT-S16 initialized with random weights (blue) and from a model pre-trained on ImageNet (orange) with DINO. Left: Linear probing performance on BACH. Center: Linear probing performance on PCam, Right: Linear probing performance on TP53.

Our findings indicate that initializing the FM from pre-trained weights accelerates its convergence and improves the computational efficiency. While training from scratch shows a gradual improvement in performance over the training, it does not converge as fast as fine-tuning a pre-trained model. We hypothesize that initializing from pre-trained weights allows the model to prioritize intricate details within image patches, which is particularly crucial for medical images. As a result, it may be able to converge to a higher level than the models initialized from random weights. Similar effects have also been observed in other works, e.g., in [34]. We expect to re-evaluate this effect on larger datasets in the future.

C. Training FMs at multiple magnifications improves robustness

Analyzing pathology images often requires adjusting magnification levels to suit specific contextual demands across different tasks. Lower magnification aids in capturing the overall tissue context, which is particularly beneficial for tasks such as grading prostate cancer. Conversely, tasks focused on individual cell classification benefit from higher magnification to achieve finer resolution. Thus, an ideal pathology FM should be applicable on a range of magnification levels for diverse tasks. We, therefore,

introduce patches of various magnification levels during training in the hope that this will enhance the model's versatility and performance across a wider range of downstream tasks, as in [29]. We evaluate the impact of training an FM using various magnifications, both individually and mixed. For this purpose, we employ a randomly initialized ViT-S16 and train it on TCGA with DINO for 100 epochs. In addition to the resolutions commonly used in the literature, namely, 40× and 20×, we include two additional resolutions, 10× and 5×. The results are presented in Table II.

Our analysis reveals that the model trained exclusively on the 20× magnification level surpasses all other models trained on individual magnifications. However, it does not perform as well as models that simultaneously incorporate multiple magnifications. Our benchmark datasets include various magnification levels, highlighting the lack of consistency in performance across different magnifications for single models, except for the one integrating all four. This integrated model's capability to understand patterns across multiple magnifications provides it with a significant advantage, leading to superior results compared to models trained and evaluated solely on a single magnification level. These findings affirm that we can develop a magnification-agnostic FM without the need for more complex model architectures.

It is also worth noting that mixing patches at multiple magnifications effectively increases the data size and its diversity. The improvement of the model performance, as a result, agrees with our general observations of improved model performance with increasing the data size as discussed in Section II-D.1.

D. The effect of training data size

In this experiment, we investigate how the performance of the FM changes with scaling up the size of the training data in two different ways: 1) increasing the number of training WSIs and 2) increasing the number of patches sampled from a fixed set of WSIs. Note that in these experiments, we only worked with the TCGA FFPE slides (∼10k WSIs).

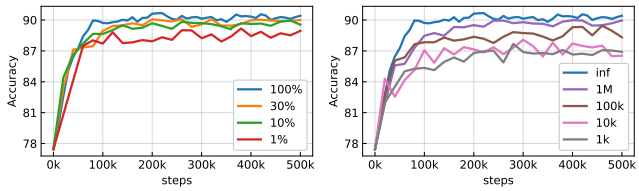

Fig. 2: Validation performance of ViT-S16 throughout the DINO training for 100 epochs on the full TCGA dataset and its random 1%, 10%, 30%, and 100% subsets of WSIs (left) and for different numbers of distinct training patches sampled at random coordinates from random WSIs of 100% TCGA (right). 'inf' represents the training where all training patches are distinct and are sampled from random coordinates. The performance is measured with linear probing on the PCam/val downstream task.

1) The number of training WSIs: To evaluate how the number of WSIs in the training dataset affects the final performance of the FM, we trained the ViT-S16 model with DINO on random subsets of TCGA of increasing size (1%, 10%, 30%, and 100% of WSIs randomly sampled from TCGA). The training was done from scratch on a single GPU for 100 epochs (5000 steps/batches per epoch, with a batch size of 256 patches) and otherwise standard DINO training parameters.

The improvement of the validation accuracy (on the PCam/val downstream task) throughout the training is shown in Fig. 2 (left), and the final test accuracy of the trained FMs on the selected downstream tasks is shown in Table III. The results show that training an FM on more data generally leads to its better performance. However, the gain diminishes with more training data. Surprisingly, even with as few as 108 training WSIs (1% of TCGA), the trained FM provides reasonably high accuracy on all downstream tasks, and the model trained on the 30% subset is nearly indistinguishable from the FM trained on the full TCGA on all downstream tasks except TP53.

TABLE III: The performance of the ViT-S16 model trained with DINO for 100 epochs on 1%, 10%, 30%, and 100% subsets of TCGA, as well as the performance of the model initialized with random weights. The performance is measured as the balanced accuracy of Linear probing on the downstream tasks.

Training set	BACH	CRC	MHIST	PCam	TP53
no training	0.410	0.689	0.500	0.728	0.500
TCGA 1%	0.668	0.908	0.731	0.871	0.560
TCGA 10%	0.662	0.928	0.712	0.887	0.592
TCGA 30%	0.733	0.924	0.752	0.897	0.610
TCGA 100%	0.723	0.927	0.752	0.899	0.621

Unlike BACH, CRC, MHIST, and PCam, the TP53 task is TCGA-based and contains WSIs from the same distribution as those used when training the FM. Note that the DINO SSL training does not see the TP53 labels; hence, this can still be considered a performance on test. We also tested the accuracy on a hold-out subset of TCGA that was not used by DINO and obtained similar results and conclusions, which we skip here for simplicity.

Based on these results, we believe that 1) FM benefits from more unique samples primarily on in-distribution (ID) data, as demonstrated by the improved performance on on the TP53 task with more training slides; 2) It is necessary to collect more diverse datasets (data from different hospitals, more tissue types, more cancer types, etc.) for the FM to generalize better on out-of-distribution (OOD) data.

2) The number of distinct patches: In this experiment, we limit the number of distinct patches cropped from the WSIs during training. More precisely, we execute the usual training pipeline with Online Patching (see Section IV-B) and cache the sampled patches on the local hard drive. After a certain number of patches has been sampled, we randomly sample all the next patches in training from those cached on the local hard drive.

The validation performance (on PCam/val) throughout the training is shown in Fig. 2 (right), and the final test accuracy of the trained FMs on the selected downstream tasks is

TABLE IV: The performance of the ViT-S16 model trained with DINO for 100 epochs on different numbers of distinct training patches. The patches are cropped from random coordinates at random WSIs of 100% TCGA. The first row corresponds to the initial untrained model with random weights. The last row ('inf' patches) represents the training where all training patches are distinct and are sampled from random coordinates. The performance is measured as balanced accuracy of Linear probing on the downstream tasks.

# patches	BACH	CRC	MHIST	PCam	TP53
0	0.410	0.689	0.500	0.728	0.500
10^3	0.661	0.927	0.775	0.846	0.529
10^4	0.699	0.932	0.780	0.860	0.542
10^5	0.644	0.938	0.790	0.864	0.573
10^6	0.683	0.926	0.743	0.898	0.611
inf	0.723	0.927	0.752	0.899	0.621

shown in Table IV. The first row in Table IV corresponds to the initial ViT-S16 model with random weights (before training). The last row ('inf' patches) represents the training without restricting the number of distinct patches. Namely, every training patch is sampled from a random WSI at its random position without caching it on the local hard drive, which results in approximately $100 \cdot 5000 \cdot 256 \approx 10^8$ distinct training patches.

Similar to the previous experiment with training the model on subsets of WSIs, we see that even training the model on as few as 1,000 random patches (which is about one patch per ten WSIs) already achieves a reasonable performance on the OOD downstream tasks, and further increasing this number does not lead to drastic improvements of the performance on the OOD tasks. However, for the in-distribution (ID) TP53 task, the performance grows steadily with exponentially increasing the number of training patches, which suggests that training the FM on more distinct training patches generally leads to better performance, especially on ID data.

The results of these two experiments provide strong evidence that 1) FM training benefits from more unique samples at both slide and patch level; 2) the performance of the FM on OOD tasks can only be significantly improved by substantially enriching and diversifying the training dataset. Even such a seemingly diverse dataset as TCGA is quickly exhausted in its capacity to facilitate the FM in its ability to generalize on OOD data, highlighting the necessity to go beyond TCGA.

III. Conclusion

In this work, we introduced our scalable pipeline for training and evaluating FMs on large pathology imaging data. Towards building a large-scale pathology model, we developed an online patching technique designed to eliminate the space overhead required to store the patches generated offline. Through our experiments on TCGA, we demonstrated that online patching does not compromise model performance and may even offer an advantage by providing more diverse data. Furthermore, this technique enables efficient and flexible experimental setup, which could lead to the discovery of novel strategies for training better pathology FMs. These encouraging results allow us to easily scale up our FM training to datasets orders of magnitude larger than TCGA.

Our experiments on TCGA suggest the following. First, fine-tuning an FM pre-trained on ImageNet on pathology data is more efficient than training a pathology FM from scratch. The initial knowledge contained in the pre-trained FM appears to be relevant for the pathology FM. Second, pathology FMs trained on data of mixed magnifications show a better performance than FMs trained on data of a single magnification. This was, to some extent, anticipated but not fully verified. This also suggests that, more generally, an FM trained on data of mixed distributions (e.g., data of different magnification or data with different staining), could perform as well as an FM trained on individual distributions separately and provides evidence that an FM could truly be foundational and work well on data from multiple distributions.

We observed clear benefits in scaling up the data size in training ViT-S16. However, only limited benefits were observed when the model size was scaled up. We hypothesize that TCGA in its whole is still not large enough. For example, it was shown in DINOv2 [25, Fig. 4] that the benefits of scaling up the model are more prominent on the larger dataset LVD-142M than the smaller ImageNet-22K dataset.

Similarly, we only observed limited benefits of using DINOv2 compared to DINO on TCGA (Appendix B). This could be because TCGA is too small to benefit from using the more advanced DINOv2 algorithm or that the downstream tasks we use to evaluate the FMs are not challenging enough to reveal the difference. We leave the analysis for data sizes beyond the TCGA scale for future work.

Through these extensive experiments and analysis, we recognize the importance of a reliable and fair evaluation. We introduced an unsupervised metric off-diagonal correlation that does not require labels on the downstream data and could provide complementary information about the models in addition to the supervised metrics. Finally, we presented our evaluation framework *eva* to ensure a consistent evaluation protocol when comparing different FMs. It is our hope and expectation that other practitioners in the field of medical machine learning and computational pathology contribute new clinically relevant downstream tasks to *eva* and adopt it for evaluating their own pathology FMs to ensure the results are comparable across different studies.

We are still at the very beginning of developing a truly foundational pathology FM. It will be worth revisiting the analysis in this work when we scale up the model and data sizes.

IV. Methods

A. Data

In our experiments, we used collections of WSIs from diverse human tissues across various medical conditions. We describe these datasets below and summarize them in Table V. For the exact partitioning of the datasets into the

TABLE V: Summary of benchmark datasets used for linear probing evaluation of FMs. (*) For the TP53 task, we randomly sampled 102,400 patches from TCGA and assigned the respective TP53 labels derived from their originating WSIs.

Dataset	# patches	Patch size	Magnification (mpp)	Task	Tissue type
BACH	400	1536×2048	20× (0.42 µm/px)	Cls (4 classes)	Breast
CRC	107,180	224×224	20× (0.50 µm/px)	Cls (9 classes)	Colorectal
MHIST	3,152	224×224	5× (2.00 µm/px)	Cls (2 classes)	Colorectal Polyp
PCam	327,680	96×96	10× (0.97 µm/px)	Cls (2 classes)	Breast lymph node
TP53	102,400*	224×224	20× (0.50 µm/px)	Cls (2 classes)	All TCGA tissues

training, test, and, where applicable, validation subsets, refer to Supplementary Data V.

TCGA This dataset contains approximately 29k hematoxylin and eosin (H&E) stained tissue slides from 32 cancer types at different microscopic magnifications, collected at different hospitals for The Cancer Genome Atlas (TCGA) project [28] by the TCGA Research Network: https://www.cancer.gov/tcga. TCGA has been widely used for training foundation models on pathology images [27, 29, 30]. Following these efforts, we used this dataset to train our foundation models.

TCGA TP53 From TCGA metadata, we constructed a downstream task for predicting TP53 status from WSIs in TCGA. To this end, we assess TP53 to be aberrated when it either harbors a mutation or when both copies are deleted. Following [35], we consider all mutations that are either labeled as moderate (e.g., non-synonymous missense) or high (e.g., nonsense, frameshift) without filtering on variant allele frequency (VAF). The rationale for including non-synonymous missense mutations is that these mutations nearly always showed a high VAF, suggesting positive selection pressure and, hence, functional impact [35]. The rationale for not filtering on VAF (for the remaining mutations, labeled as high impact) is that for nearly all cases where only one allele is mutated, a second mutation could be determined, suggesting the other allele has also been disabled [35]. Following this approach, we identified roughly 6k tumors with functional TP53 and roughly 3.5k tumors with non-functional TP53 in TCGA. We have made these data available for download (see V).

The TP53 status is a patient-level signal; however, in this work, we treat it as a patch-level signal. That is, the task is to predict TP53 status from a patch of a WSI instead of the whole slide. This introduces label noise, as the WSI-level signal will most likely not be detectable from every patch within the WSI. Also, there could be heterogeneity in the expression of the involved genes. Nonetheless, we find that this task can be used to compare between FMs and could be a valid metric. We leave the construction of the slide-level metric for future work.

For evaluation, we randomly sample 102,400 patches from all the TCGA diagnostic slides with equal probability for each slide to be sampled, and we report linear probing balanced accuracy on 5-fold cross-validation of patient-based splits of the patches.

BACH This dataset contains 400 images originally generated for the Grand Challenge on BreAst Cancer Histology images challenge [36]. Each image is of size 1536×2048 pixels, at a scale of 0.42 µm/pixel, and belongs to one of four classes: 1) normal, 2) benign, 3) *in situ* carcinoma, and 4) invasive carcinoma, with each of the four classes having exactly 100 images assigned to it. We downloaded all images with their corresponding metadata from https://iciar2018-challenge.grand-challenge.org and used this metadata to split the entire dataset into a training and a test part, such that different images from the same patient never appear in both training and test parts but only in one of them. As a result, the test set contains 132 images, with 23, 48, 30, and 31 images from each of the four classes, respectively, which is roughly one-third of the entire dataset. Note that this differs from the existing literature, where typically a random split is performed without grouping the patches by patient, e.g., in [29]. For the exact composition of the training and test sets, refer to Supplementary Data V.

CRC The CRC dataset [37] comprises 100,000 training and 7,180 test images (224×224 pixels) at 20× magnification, sourced from H&E stained WSIs representing human colorectal cancer and normal tissue. The training set is derived from 86 WSIs, while the test set is sourced from 25 WSIs. These WSIs are obtained from the NCT Tissue Bank and the University Medical Center Mannheim. The objective is to classify nine tissue classes: adipose tissue, background, debris, lymphocytes, mucus, smooth muscle, normal colon mucosa, cancer-associated stroma, and CRC epithelium. All images undergo color normalization using the Macenko method (NCT-CRC-HE-100K). We do not make any use of the unnormalized (NCT-CRC-HE100K-NONORM) variants.

PatchCamelyon (PCam) This dataset consists of 327,680 patches of 96×96 pixels at a FoV of 0.97 µm/px [38]. These patches are cropped from WSIs of breast lymph node sections and are marked with binary labels indicating the presence of metastatic tissue in the image.

MHIST The MHIST dataset [39] consists of 3,152 H& E-stained FFPE fixed-size images (224×224 pixels) of colorectal polyps, where each image is assigned to one of two classes: 1) Hyperplastic Polyp (HP) or 2) Sessile Serrated Adenoma (SSA).

CoNSeP The Colorectal Nuclear Segmentation and Phenotypes (CoNSeP) dataset [40] consists of 41 H&E 1000×1000 pixel images, and it is split into 27 and 14 images for training and test sets. The data comes from the University Hospitals Coventry and Warwickshire, UK. The annotation contains segmentation masks of each nucleus

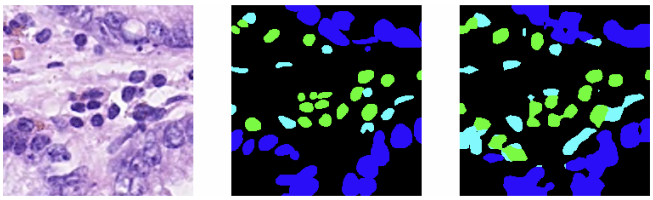

Fig. 3: Quantitative results for the semantic segmentation task. Left: the patch. Center: the semantic segmentation labels. Right: the predictions with our ViT-B8 model.

along with the grouped classes as described in [40]: 1) other, 2) inflammatory, 3) healthy & dysplastic/malignant epithelial, 4) spindle-shaped. To evaluate FMs, we used the CoNSeP dataset as a semantic segmentation task, where each pixel is assigned to one of the five categories: the four cell-type categories and background. We report the DICE score without background.

B. Online high-throughput loading of patches from WSIs

Self-supervised pre-training on WSIs is, as of date, usually not performed at the slide level due to the gigapixel-level size of the WSIs, which exceeds the GPU memory of standard modern hardware. Although sophisticated techniques such as activation checkpointing could be used [41, 42], they usually have a significant performance impact for large models. Therefore, state-of-the-art pathology FMs are currently trained at the patch level, where smaller patches must be extracted. Typically, patches are extracted and stored offline before training to enable the efficient loading of patches during training; see e.g. [32]. In addition to the significant space overhead required to store the pre-processed image patches, this limits the flexibility in the choice of patch size and the magnification level, as any change in these parameters requires another preparation of the patches.

To allow for more flexibility, we developed *Online Patching*, a library that enables high-throughput extraction and loading of patches from WSIs during training. The library allows extracting from any WSI patches at completely arbitrary coordinates and at arbitrary magnification levels. This allows training models on virtually all the patches contained in the WSIs without having to store the patches on disk.

To sample only foreground patches, a U-Net-based foreground segmentation model is used to compute the foreground masks of the WSIs at a lower resolution, usually at thumbnail scale. From this mask, a polygon is computed that can be efficiently stored in memory. During training, a slide is first sampled from all available slides, where different slide-level sampling strategies could be specified (e.g., uniform random sampling or prior-based sampling). From the sampled slide, patch coordinates are randomly sampled with a minimum area overlap of a candidate patch with the polygon. The patch is then extracted from the image level closest to the target magnification level and resized to the target patch size. This method of patch selection increases the diversity of the patches used in training and allows training on more patches than what would be possible with a fixed set of patches. At inference time, patches with sufficiently many foreground pixels can be generated by iterating a grid of a specified size.

The online patching library utilizes a virtual in-memory filesystem to make the whole of TCGA accessible as a single *Zarr* data source [43]. The virtual filesystem allows the original SVS files to be accessible as if they were stored in the Zarr data format, requiring only minimal extraction of the tile byte ranges before running online patching. The library provides optimized functionality for the asynchronous loading of tiles from network or blob storage. Related open-source initiatives are being developed to use Zarr as a unified format for biomedical images [44, 45].

We are aware of other possible solutions where patch extraction is performed on intermediate servers, providing an API to abstract away that complexity. An example of that approach is the WSI DICOMWeb python library from Google [46], which provides a way to extract patches from images stored in the Google DICOM store. The downside of solutions that rely on intermediate servers performing the patch extraction is that it increases the complexity of the infrastructure required (for larger datasets, the patch extraction servers would need to scale up accordingly). Our online patching library, on the other hand, works with WSIs directly and in a cloud-agnostic way.

C. Pretraining setup

We adhere to the suggested training methodology for natural images as outlined in DINO [24] and DINOv2 [25] with slight adjustments: (i) model initialization is performed using ImageNet SSL weights, (ii) the learning rate is reduced by a factor of 10, (iii) random sampling of 256×256 patches is carried out with a minimum of 40% foreground presence, (iv) training encompasses multiple magnification levels, specifically 5×, 10×, 20×, and 40×, and (v) image normalization is conducted using a non-informative mean and standard deviation of 0.5 to scale values within the [-1, 1] range.

In all our experiments, we adopt the ImageNet Epoch concept [47] and define one epoch as 1,280,000 patches.

The pre-training occurred in two phases: In the first phase, we trained exclusively on diagnostic slides for 100 ImageNet epochs. In the second phase, we extended the training by another 100 epochs, incorporating the FF slides while reducing the peak learning rate by half compared to the initial stage.

The ViT-B8 was trained on 8 H100 GPUs with a batch size of 32 per GPU. The DINOv2 ViT-L14 was trained on 16 H100 GPUs with batch size per GPU 32, and all other our models from Table I were trained on 4 H100 GPUs and batch size per GPU 256 for ViT-S16, 64 for ViT-S8 and 128 for ViT-B16.

D. Evaluation setup

To evaluate different training strategies, we apply the trained FMs on a selection of downstream tasks using

public datasets to generate embeddings given input images. The performance of the FMs is evaluated with two groups of metrics: 1) metrics that evaluate the quality of the representations directly without labels; 2) metrics measuring the performance of the representations on the downstream prediction tasks with a lightweight head network, where labels are necessary.

1) Unsupervised metrics: To evaluate FMs without the need for labeled data, we use RankMe [48] as one of the metrics in this study. RankMe estimates the rank of embeddings of test data, and it has been shown to correlate well with downstream performance.

In addition, we introduce another simple unsupervised criterion to evaluate the quality of the representations directly: off-diagonal correlation (ODCorr), which simply measures the average correlation coefficient between the embeddings of different samples in the evaluation dataset, i.e., the off-diagonal elements of a correlation matrix. This metric is motivated by the observation that when the embeddings of different samples are different enough and, thus, not correlated, the samples can be distinguished from each other based on their embedding vectors. This is necessary for learning downstream tasks.

ODCorr is calculated based on the 2D matrix of embedding dimensions and samples, and it takes the square root of the mean square of the correlations between embedding vectors of pairs of *samples*. In related earlier work, the correlations over this 2D matrix are computed in the orthogonal direction, i.e., the correlations between sample-value vectors of pairs of *embedding dimensions* are considered. [49] showed that the amount of correlation in hidden activations corresponds with the amount of overfitting, and [50] visualized the Pearson correlation coefficient of [CLS] embeddings such that highly correlated dimensions are located near each other in blocks.

The off-diagonal correlation metric ranges between 0 and 1, with 0 indicating no correlations between samples and 1 indicating all samples are correlated with each other. Since this metric simply measures the correlations between different samples, it can be used to compare models of different dimensions. Denoting the embedding matrix of an evaluation dataset as $\boldsymbol{Z}$ of shape $N \times K$, the off-diagonal correlation is formally defined as:

$$\text{ODCorr}(\boldsymbol{Z}) = \sqrt{\frac{\sum_{i \neq j} \rho(Z_i, Z_j)^2}{N(N-1)}}, \tag{1}$$

where $\rho(Z_i, Z_j)$ is the Pearson correlation coefficient for the embeddings of samples i and j.

In Appendix, Section C, we show that ODCorr highly correlates with the downstream performance and can be compared between models of different sizes.

2) Evaluation framework: eva: To evaluate foundation models on out-of-distribution (OOD) downstream tasks, we use our open-source evaluation framework *eva*, which has been designed to provide an explainable, fair, and easily reproducible FM-evaluation standard across backbone sizes and architectures.

eva aims to support a large selection of public datasets and tasks. In the first release, PCam [38], BACH [36] for breast cancer classification and colorectal (CRC) cancer classification [37] are included. To evaluate an FM on a downstream task, *eva* prepares a task dataset, performs inference to compute the embeddings, trains a head model for that task, and evaluates the performance.

If a dataset has a designated validation or test split, we use it for evaluation and report results accordingly. However, if such splits are not available, we create a stratified split to ensure proper separation of slides or patients, thus preventing any potential data leakage.

eva prioritizes meaningful evaluation of the FMs over maximum individual downstream performance. Therefore, the head architecture is deliberately chosen to be lightweight, robust in downstream performance, and with minimal bias toward any particular FM architecture.

To achieve this, *eva* follows a standard protocol introduced in [33] that trains a single-layer MLP with a fixed number of training steps and hyperparameters. For small datasets, we reduced the batch size and linearly scaled down the learning rate. To prevent overfitting, *eva* applies early stopping after 5% of the maximum number of epochs. For a detailed configuration, see Appendix, Table VI. We found that with this setup, we achieve stable results across multiple runs for each of the evaluated tasks and FMs.

For the semantic segmentation evaluation on the CoNSeP dataset, we trained on randomly cropped patches of size 224 from the training set and evaluated on grid patches of the same size with stride 194 on the test set. We used Mask2Former [51] as a decoder on top of the FMs. We kept the FM frozen and the decoder light. We deliberately reduced the capacity of the decoder (number of queries: 32, number of encoder layers and attention heads: 4, feature size and hidden dimension: 32). In the same spirit of keeping a lightweight decoder, we did not use ViT-Adapter [52], which is sometimes used when evaluating FMs, e.g., in [31]. We observed some instabilities in training the segmentation decoder. Not all training runs converged in the maximal number of epochs. Thus, for each FM, the results are averaged over three runs that did converge.

V. Supplementary Data

The model checkpoints, and the information for reproducing the evaluation results presented in this work are available for download from https://github.com/kaiko-ai/towards_large_pathology_fms.

VI. Acknowledgments

The authors thank Jonas Teuwen and the AI for Oncology group at the Netherlands Cancer Institute (NKI) for providing a segmentation annotation dataset, which was used to train the foreground/background segmentation model that identifies the foreground regions used by the Online Patching method described in this article.

REFERENCES

[1] G. Campanella, M. G. Hanna, L. Geneslaw, A. Miraflor, V. Werneck Krauss Silva, K. J. Busam, E. Brogi, V. E. Reuter, D. S. Klimstra, and T. J. Fuchs, "Clinical-grade computational pathology using weakly supervised deep learning on whole slide images," Nature medicine, vol. 25, no. 8, pp. 1301–1309, 2019.

[2] M. Y. Lu, D. F. Williamson, T. Y. Chen, R. J. Chen, M. Barbieri, and F. Mahmood, "Data-efficient and weakly supervised computational pathology on whole-slide images," Nature biomedical engineering, vol. 5, no. 6, pp. 555–570, 2021.

[3] A. Echle, N. T. Rindtorff, T. J. Brinker, T. Luedde, A. T. Pearson, and J. N. Kather, "Deep learning in cancer pathology: a new generation of clinical biomarkers," British journal of cancer, vol. 124, no. 4, pp. 686–696, 2021.

[4] D. Tellez, M. Balkenhol, I. Otte-Höller, R. van de Loo, R. Vogels, P. Bult, C. Wauters, W. Vreuls, S. Mol, N. Karssemeijer, et al., "Whole-slide mitosis detection in h&e breast histology using phh3 as a reference to train distilled stain-invariant convolutional networks," IEEE transactions on medical imaging, vol. 37, no. 9, pp. 2126–2136, 2018.

[5] W. Bulten, H. Pinckaers, H. van Boven, R. Vink, T. de Bel, B. van Ginneken, J. van der Laak, C. Hulsbergen-van de Kaa, and G. Litjens, "Automated deep-learning system for gleason grading of prostate cancer using biopsies: a diagnostic study," The Lancet Oncology, vol. 21, no. 2, pp. 233–241, 2020.

[6] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. Van Der Laak, B. Van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," Medical image analysis, vol. 42, pp. 60–88, 2017.

[7] J. Van der Laak, G. Litjens, and F. Ciompi, "Deep learning in histopathology: the path to the clinic," Nature medicine, vol. 27, no. 5, pp. 775–784, 2021.

[8] N. Dimitriou, O. Arandjelović, and P. D. Caie, "Deep learning for whole slide image analysis: an overview," Frontiers in medicine, vol. 6, p. 264, 2019.

[9] L. A. Hildebrand, C. J. Pierce, M. Dennis, M. Paracha, and A. Maoz, "Artificial intelligence for histology-based detection of microsatellite instability and prediction of response to immunotherapy in colorectal cancer," Cancers, vol. 13, no. 3, p. 391, 2021.

[10] J. Zhu, W. Wu, Y. Zhang, S. Lin, Y. Jiang, R. Liu, H. Zhang, and X. Wang, "Computational analysis of pathological image enables interpretable prediction for microsatellite instability," Frontiers in Oncology, vol. 12, p. 825353, 2022.

[11] C. Saillard, O. Dehaene, T. Marchand, O. Moindrot, A. Kamoun, B. Schmauch, and S. Jegou, "Self supervised learning improves dmmr/msi detection from histology slides across multiple cancers," arXiv preprint arXiv:2109.05819, 2021.

[12] Z. R. McCaw, A. Shcherbina, Y. Shah, D. Huang, S. Elliott, P. M. Szabo, B. Dulken, S. Holland, P. Tagari, D. Light, et al., "Machine learning enabled prediction of digital biomarkers from whole slide histopathology images," medRxiv, pp. 2024–01, 2024.

[13] O. S. El Nahhas, C. M. Loeffler, Z. I. Carrero, M. van Treeck, F. R. Kolbinger, K. J. Hewitt, H. S. Muti, M. Graziani, Q. Zeng, J. Calderaro, et al., "Regression-based deep-learning predicts molecular biomarkers from pathology slides," Nature Communications, vol. 15, no. 1, p. 1253, 2024.

[14] B. Schmauch, A. Romagnoni, E. Pronier, C. Saillard, P. Maillé, J. Calderaro, A. Kamoun, M. Sefta, S. Toldo, M. Zaslavskiy, et al., "A deep learning model to predict rna-seq expression of tumours from whole slide images," Nature communications, vol. 11, no. 1, p. 3877, 2020.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Advances in neural information processing systems, vol. 25, 2012.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778, 2016.

[18] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in Proceedings of the IEEE international conference on computer vision, pp. 2961–2969, 2017.

[19] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al., "Gpt-4 technical report," arXiv preprint arXiv:2303.08774, 2023.

[20] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," CoRR, vol. abs/2002.05709, 2020.

[21] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," CoRR, vol. abs/2103.00020, 2021.

[22] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, "Barlow twins: Self-supervised learning via redundancy reduction," CoRR, vol. abs/2103.03230, 2021.

[23] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," CoRR, vol. abs/2006.09882, 2020.

[24] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in Proceedings of the IEEE/CVF international conference on computer vision, pp. 9650–9660, 2021.

[25] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, R. Howes, P.-Y. Huang, H. Xu, V. Sharma, S.-W. Li, W. Galuba, M. Rabbat, M. Assran, N. Ballas, G. Synnaeve, I. Misra, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, "Dinov2: Learning robust visual features without supervision," 2023.

[26] O. Dehaene, A. Camara, O. Moindrot, A. de Lavergne, and P. Courtiol, "Self-supervision closes the gap between weak and strong supervision in histology," arXiv preprint arXiv:2012.03583, 2020.

[27] R. J. Chen, C. Chen, Y. Li, T. Y. Chen, A. D. Trister, R. G. Krishnan, and F. Mahmood, "Scaling vision transformers to gigapixel images via hierarchical self-supervised learning," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16144–16155, 2022.

[28] K. Chang, C. J. Creighton, C. Davis, L. Donehower, J. Drummond, D. Wheeler, A. Ally, M. Balasundaram, I. Birol, Y. S. N. Butterfield, A. Chu, E. Chuah, H.-J. E. Chun, N. Dhalla, R. Guin, M. Hirst, C. Hirst, R. A. Holt, S. J. M. Jones, D. Lee, H. I. Li, M. A. Marra, M. Mayo, R. A. Moore, A. J. Mungall, A. G. Robertson, J. E. Schein, P. Sipahimalani, A. Tam, N. Thiessen, R. J. Varhol, R. Beroukhim, A. S. Bhatt, A. N. Brooks, A. D. Cherniack, S. S. Freeman, S. B. Gabriel, E. Helman, J. Jung, M. Meyerson, A. I. Ojesina, C. S. Pedamallu, G. Saksena, S. E. Schumacher, B. Tabak, T. Zack, E. S. Lander, C. A. Bristow, A. Hadjipanayis, P. Haseley, R. Kucherlapati, S. Lee, E. Lee, L. J. Luquette, H. S. Mahadeshwar, A. Pantazi, M. Parfenov, P. J. Park, A. Protopopov, X. Ren, N. Santoso, J. Seidman, S. Seth, X. Song, J. Tang, R. Xi, A. W. Xu, L. Yang, D. Zeng, J. T. Auman, S. Balu, E. Buda, C. Fan, K. A. Hoadley, C. D. Jones, S. Meng, P. A. Mieczkowski, J. S. Parker, C. M. Perou, J. Roach, Y. Shi, G. O. Silva, D. Tan, U. Veluvolu, S. Waring, M. D. Wilkerson, J. Wu, W. Zhao, T. Bodenheimer, D. N. Hayes, A. P. Hoyle, S. R. Jeffreys, L. E. Mose, J. V. Simons, M. G. Soloway, S. B. Baylin, B. P. Berman, M. S. Bootwalla, L. Danilova, J. G. Herman, T. Hinoue, P. W. Laird, S. K. Rhie, H. Shen, T. Triche, D. J. Weisenberger, S. L. Carter, K. Cibulskis, L. Chin, J. Zhang, G. Getz, C. Sougnez, M. Wang, H. Dinh, H. V. Doddapaneni, R. Gibbs, P. Gunaratne, Y. Han, D. Kalra, C. Kovar, L. Lewis, M. Morgan, D. Morton, D. Muzny, J. Reid, L. Xi, J. Cho, D. DiCara, S. Frazer, N. Gehlenborg, D. I. Heiman, J. Kim, M. S. Lawrence, P. Lin, Y. Liu, M. S. Noble, P. Stojanov, D. Voet, H. Zhang, L. Zou, C. Stewart, B. Bernard, R. Bressler, A. Eakin, L. Iype, T. Knijnenburg, R. Kramer, R. Kreisberg, K. Leinonen, J. Lin, Y. Liu, M. Miller, S. M. Reynolds, H. Rovira, I. Shmulevich, V. Thorsson, D. Yang, W. Zhang, S. Amin, C.-J. Wu, C.-C. Wu, R. Akbani, K. Aldape, K. A. Baggerly, B. Broom, T. D. Casasent, J. Cleland, C. Creighton, D. Dodda, M. Edgerton, L. Han, S. M. Herbrich, Z. Ju, H. Kim, S. Lerner, J. Li, H. Liang, W. Liu, P. L. Lorenzi, Y. Lu, J. Melott, G. B. Mills, L. Nguyen, X. Su, R. Verhaak, W. Wang, J. N. Weinstein, A. Wong, Y. Yang, J. Yao, R. Yao, K. Yoshihara, Y. Yuan, A. K. Yung, N. Zhang, S. Zheng, M. Ryan, D. W. Kane, B. A. Aksoy, G. Ciriello, G. Dresdner, J. Gao, B. Gross, A. Jacobsen, A. Kahles, M. Ladanyi, W. Lee, K.-V. Lehmann, M. L. Miller, R. Ramirez, G. Rätsch, B. Reva, C. Sander, N. Schultz, Y. Senbabaoglu, R. Shen, R. Sinha, S. O. Sumer, Y. Sun, B. S. Taylor, N. Weinhold, S. Fei, P. Spellman, C. Benz, D. Carlin, M. Cline, B. Craft, K. Ellrott, M. Goldman, D. Haussler, S. Ma,

S. Ng, E. Paull, A. Radenbaugh, S. Salama, A. Sokolov, J. M. Stuart, T. Swatloski, V. Uzunangelov, P. Waltman, C. Yau, J. Zhu, S. R. Hamilton, S. Abbott, R. Abbott, N. D. Dees, K. Delehaunty, L. Ding, D. J. Dooling, J. M. Eldred, C. C. Fronick, R. Fulton, L. L. Fulton, J. Kalicki-Veizer, K.-L. Kanchi, C. Kandoth, D. C. Koboldt, D. E. Larson, T. J. Ley, L. Lin, C. Lu, V. J. Magrini, E. R. Mardis, M. D. McLellan, J. F. McMichael, C. A. Miller, M. O'Laughlin, C. Pohl, H. Schmidt, S. M. Smith, J. Walker, J. W. Wallis, M. C. Wendl, R. K. Wilson, T. Wylie, Q. Zhang, R. Burton, M. A. Jensen, A. Kahn, T. Pihl, D. Pot, Y. Wan, D. A. Levine, A. D. Black, J. Bowen, T. C. G. A. R. Network, G. C. Center, G. D. A. Center, S. Center, D. C. Center, T. S. Site, and B. C. R. Center, "The cancer genome atlas pan-cancer analysis project," Nature Genetics, vol. 45, no. 10, pp. 1113–1120, 2013.

[29] M. Kang, H. Song, S. Park, D. Yoo, and S. Pereira, "Benchmarking self-supervised learning on diverse pathology datasets," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3344–3354, June 2023.

[30] A. Filiot, R. Ghermi, A. Olivier, P. Jacob, L. Fidon, A. M. Kain, C. Saillard, and J.-B. Schiratti, "Scaling self-supervised learning for histopathology with masked image modeling," medRxiv, 2023.

[31] R. J. Chen, T. Ding, M. Y. Lu, D. F. K. Williamson, G. Jaume, A. H. Song, B. Chen, A. Zhang, D. Shao, M. Shaban, M. Williams, L. Oldenburg, L. L. Weishaupt, J. J. Wang, A. Vaidya, L. P. Le, G. Gerber, S. Sahai, W. Williams, and F. Mahmood, "Towards a general-purpose foundation model for computational pathology," Nature Medicine, 2024.

[32] G. Campanella, C. Vanderbilt, and T. Fuchs, "Computational pathology at health system scale – self-supervised foundation models from billions of images," in AAAI 2024 Spring Symposium on Clinical Foundation Models, 2024.

[33] E. Vorontsov, A. Bozkurt, A. Casson, G. Shaikovski, M. Zelechowski, S. Liu, K. Severson, E. Zimmermann, J. Hall, N. Tenenholtz, N. Fusi, P. Mathieu, A. van Eck, D. Lee, J. Viret, E. Robert, Y. K. Wang, J. D. Kunz, M. C. H. Lee, J. Bernhard, R. A. Godrich, G. Oakley, E. Millar, M. Hanna, J. Retamero, W. A. Moye, R. Yousfi, C. Kanan, D. Klimstra, B. Rothrock, and T. J. Fuchs, "Virchow: A million-slide digital pathology foundation model," arXiv:2309.07778v5, 2024.

[34] A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly, and N. Houlsby, "Large scale learning of general visual representations for transfer," CoRR, vol. abs/1912.11370, 2019.

[35] L. A. Donehower, T. Soussi, A. Korkut, Y. Liu, A. Schultz, M. Cardenas, X. Li, O. Babur, T.-K. Hsu, O. Lichtarge, J. N. Weinstein, R. Akbani, and D. A. Wheeler, "Integrated analysis of tp53 gene and pathway alterations in the cancer genome atlas.," Cell Rep, vol. 28, pp. 1370–1384, Jul 2019.

[36] G. Aresta, T. Araújo, S. Kwok, S. S. Chennamsetty, M. Safwan, V. Alex, B. Marami, M. Prastawa, M. Chan, M. Donovan, G. Fernandez, J. Zeineh, M. Kohl, C. Walz, F. Ludwig, S. Braunewell, M. Baust, Q. D. Vu, M. N. N. To, E. Kim, J. T. Kwak, S. Galal, V. Sanchez-Freire, N. Brancati, M. Frucci, D. Riccio, Y. Wang, L. Sun, K. Ma, J. Fang, I. Kone, L. Boulmane, A. Campilho, C. Eloy, A. Polónia, and P. Aguiar, "Bach: Grand challenge on breast cancer histology images.," Med Image Anal, vol. 56, pp. 122–139, Aug 2019.

[37] J. N. Kather, N. Halama, and A. Marx, "100,000 histological images of human colorectal cancer and healthy tissue," May 2018.

[38] B. S. Veeling, J. Linmans, J. Winkens, T. Cohen, and M. Welling, "Rotation equivariant CNNs for digital pathology," June 2018.

[39] J. Wei, A. Suriawinata, B. Ren, X. Liu, M. Lisovsky, L. Vaickus, C. Brown, M. Baker, N. Tomita, L. Torresani, J. Wei, and S. Hassanpour, "A petri dish for histopathology image analysis," International Conference on Artificial Intelligence in Medicine (AIME), vol. 12721, pp. 11–24, 2021.

[40] S. Graham, Q. D. Vu, S. E. A. Raza, A. Azam, Y. W. Tsang, J. T. Kwak, and N. Rajpoot, "Hover-net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images," Medical image analysis, vol. 58, p. 101563, 2019.

[41] H. Pinckaers, B. Van Ginneken, and G. Litjens, "Streaming convolutional neural networks for end-to-end learning with multi-megapixel images," IEEE transactions on pattern analysis and machine intelligence, vol. 44, no. 3, pp. 1581–1590, 2020.

[42] S. Dooper, H. Pinckaers, W. Aswolinskiy, K. Hebeda, S. Jarkman, J. van der Laak, G. Litjens, B. Consortium, et al., "Gigapixel end-to-end training using streaming and attention," Medical Image Analysis, vol. 88, p. 102881, 2023.

[43] A. Miles, jakirkham, M. Bussonnier, J. Moore, D. P. Orfanos, J. Bourbeau, A. Fulton, D. Bennett, G. Lee, S. Verma, Z. Patel, R. Abernathey, D. Stansby, M. R. B. Kristensen, M. Rocklin, A. B. AWA, J. Hamman, S. Chopra, E. S. de Andrade, M. Durant, V. Schut, raphael dussin, J. Nunez-Iglesias, C. Barnes, S. Chaudhary, shikharsg, hailiangzhang, and W. Gikunda, "zarr-developers/zarr-python: v2.17.1," Mar. 2024.

[44] J. Moore, C. Allan, S. Besson, J.-M. Burel, E. Diel, D. Gault, K. Kozlowski, D. Lindner, M. Linkert, T. Manz, et al., "Ome-ngff: a next-generation file format for expanding bioimaging data-access strategies," Nature methods, vol. 18, no. 12, pp. 1496–1498, 2021.

[45] J. Moore, D. Basurto-Lozada, S. Besson, J. Bogovic, J. Bragantini, E. M. Brown, J.-M. Burel, X. Casas Moreno, G. de Medeiros, E. E. Diel, et al., "Ome-zarr: a cloud-optimized bioimaging file format with international community support," Histochemistry and Cell Biology, vol. 160, no. 3, pp. 223–251, 2023.

[46] G. HealthAI and G. C. H. teams, "Accelerate ai development for digital pathology using ez wsi dicomweb python library."

[47] M. Kang, H. Song, S. Park, D. Yoo, and S. Pereira, "Benchmarking self-supervised learning on diverse pathology datasets," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3344–3354, 2023.

[48] Q. Garrido, R. Balestriero, L. Najman, and Y. Lecun, "RankMe: Assessing the downstream performance of pretrained self-supervised representations by their rank," in Proceedings of the 40th International Conference on Machine Learning, vol. 202 of Proceedings of Machine Learning Research, pp. 10929–10974, PMLR, 23–29 Jul 2023.

[49] M. Cogswell, F. Ahmed, R. B. Girshick, L. Zitnick, and D. Batra, "Reducing overfitting in deep networks by decorrelating representations," in 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings, 2016.

[50] W. Zhou, B. Y. Lin, and X. Ren, "Isobn: Fine-tuning bert with isotropic batch normalization," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 14621–14629, 2021.

[51] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1280–1289, 2022.

[52] Z. Chen, Y. Duan, W. Wang, J. He, T. Lu, J. Dai, and Y. Qiao, "Vision transformer adapter for dense predictions," 2022.

[53] X. Chen, H. Fan, R. B. Girshick, and K. He, "Improved baselines with momentum contrastive learning," CoRR, vol. abs/2003.04297, 2020.

[54] J. Zhou, C. Wei, H. Wang, W. Shen, C. Xie, A. Yuille, and T. Kong, "ibot: Image bert pre-training with online tokenizer," International Conference on Learning Representations (ICLR), 2022.

[55] A. Ghosh, A. K. Mondal, K. K. Agrawal, and B. Richards, "Investigating power laws in deep representation learning," 2022.

[56] R. Wightman, "Pytorch image models." https://github.com/rwightman/pytorch-image-models, 2019.

[57] A. Krizhevsky, "Learning multiple layers of features from tiny images," tech. rep., University of Toronto, 2009.

[58] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101 – mining discriminative components with random forests," in European Conference on Computer Vision, 2014.

APPENDIX

A. Evaluation setup

For every experiment conducted, we adhered to the setup described in the evaluation of Virchow [33]. Specifically, we employed a linear projection classifier with a batch size of 4,096, utilizing the stochastic gradient descent (SGD) optimizer with a cosine learning rate schedule ranging from 0.01 to 0 over 12,500 iterations. This was done on top of the embeddings produced by the frozen encoder. Moreover, we implemented early stopping, halting training after 5% of the total training epochs. For further details, refer to Table VI.

B. No significant benefits of DINOv2 over DINO on TCGA

Kang et al [29] have concluded that "there is no clear winner" in different SSL methods they have examined, including MoCo v2 [53], SwAV [23], Barlow Twins [22] and DINO [24]. However, the ViT models trained with DINO did seem to be superior in label efficiency (in [29, Table 6]).

Filiot et al [30] have compared iBOT [54] with MoCo v2 [53] and shown better performance with iBOT. But the comparison with DINO was not conclusive as the DINO model was only trained on one specific cohort instead of on the whole pan-cancer dataset.

DINOv2 [25] has been introduced to incorporate the best of DINO and iBOT algorithms. In Table 1 of the DINOv2 paper [25] the benefits of different components in the DINOv2 algorithm in comparison to the iBOT algorithm [54] were shown. However, no such comparison has been made between DINO and DINOv2. In comparison to DINO [24], DINOv2 introduced the following three main components among other improvements:

- Patch-level objective, and untied head weights between image- and patch-level objectives
- Sinkhorn-Knopp centering
- KoLeo regularizer

The patch-level objective, in particular, increases the complexity of loss computation and requirements on GPU memory by the number of patches per image. It is not clear a priori whether this increased complexity is justified by, e.g., greater data efficiency or other benefits, especially on smaller data and model sizes.

TABLE VI: Hyperparameters for the head used in downstream evaluation in *eva*.

Backbone	frozen
Hidden layers	None
Dropout	0.0
Activation function	None
Number of steps	12,500
Base batch size	4,096
Batch size	dataset specific
Base learning rate	0.01
Learning rate	dataset specific
Early stopping	[Max epochs] / 20
Optimizer	SGD
Momentum	0.9
Weight Decay	0.0
Nesterov momentum	True
LR Schedule	Cosine without warmup

TABLE VII: Summary of hyperparameters for training in the experiment comparing DINO and DINOv2 (Section B).

	DINO	**DINOv2**
Batch size	256	256
Learning rate	0.0005	0.0005, 0.001, 0.002
Steps per epoch	5,000	5,000
Max epochs	100	100
iBOT separate head	n/a	False
Layer wise decay	n/a	0.9, 1

In our experiments, we did not observe significant benefits of models trained by DINOv2 over DINO on TCGA. As shown in Table I, a larger model (ViT-L14) trained with DINOv2 only achieved comparable performance in downstream tasks as the smaller ViT-S16 model trained with DINO.

Furthermore, we also examined the individual learning curves with the two algorithms to check whether one learns faster than the other (detailed setup in Section V). Fig. 4 shows the learning curves of training a ViT-S16 model using DINOv2 and DINO algorithms on a single A100-80GB machine, as measured by the linear probing performance on OOD datasets: BACH and PCam, as well as the off-diagonal correlation on in-distribution TCGA data. We have observed that DINOv2 tends to take more time for one training step than DINO. However, we do not compare by walltime directly here as this could be affected by many other factors, such as latency in connecting to cloud storage, amount of validation during training, etc. The number of training steps, on the other hand, is comparable, as we use the same batch size of 256 across different experiments.

No significant difference can be observed in the BACH and PCam learning curves between the two learning algorithms. The models trained with DINOv2 reached comparable performance in comparable number of training steps as the models trained with DINOv1.

However, we do observe better off-diagonal correlation with DINOv2 trained models. We hypothesize that the DINOv2 trained models do tend to generate more distinguishable embeddings, as demonstrated by the left panel in Fig. 4 and in Table I and its extended version Table VIII. The downstream prediction task may however not necessarily need this level of distinction between samples, thus the benefits do not necessarily show up in downstream tasks.

In summary we believe that DINOv2 may be superior to DINOv1 at the expense of slightly more compute and resource requirements. The benefits, however, may not directly translate into downstream task performance, especially if the tasks are relatively simple.

1) Pretraining setup in comparing DINO vs DINOv2: To compare DINO with DINOv2 algorithm, a ViT-S16 model was trained with both training algorithm on one A100-80GB machine with the following hyperparameters in Table VII:

Note that this is not the same setting as we used for training the models in Table I, as here in order to evaluate the effect of DINOv2 we have kept everything else the same

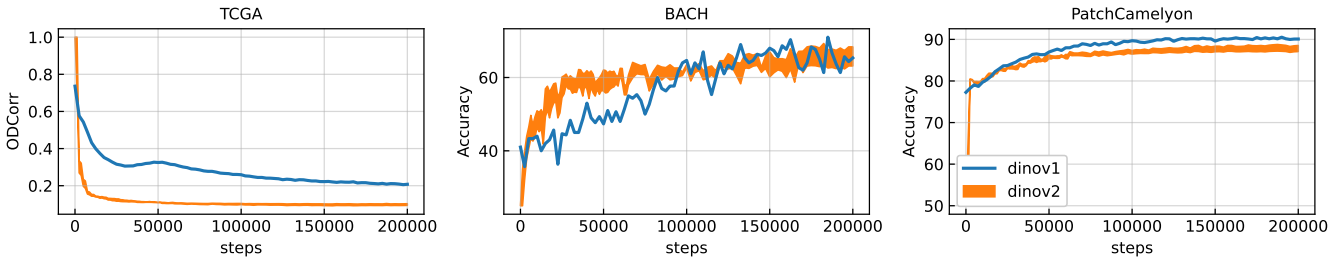

Fig. 4: Validation performance over the course of training a ViT-S16 model using DINOv2 (orange) and DINO (blue). Left: Off-diagonal correlation on randomly selected TCGA patches. Center: Linear probing performance on test split of BACH dataset. Right: Linear probing performance on validation split of PCam dataset. The orange curves for DINOv2 show a range from 4 different runs with different learning rates, while the blue curves show one single run with DINO using the standard setting, details can be found in Section A.

except for the loss definition and the data transformation.

C. ODCorr correlates with downstream performance

Losses in various SSL algorithms are usually not very informative, and, in particular, they are usually not indicative of the performance of the FMs on the downstream tasks. One way to evaluate the quality of the FMs is to apply it on some labeled datasets and evaluate the performance on the downstream tasks. However, the evaluation is constrained, and/or biased by the labels available. A few metrics have been proposed to address this, such as RankMe [48] which estimates the embeddings' rank or α-ReQ [55] which estimates the eigenspectrum decay of the representations.

In [48] it is claimed that RankMe can consistently predict downstream performance for linear and non-linear probing, however, as RankMe depends on the dimension of the representations, it is not comparable between models of different embedding dimensions and should "only be used to compare different runs of a given method". In this study we use RankMe as one of the metrics to evaluate the different training strategies.

1) On natural images: To evaluate the effectiveness of the ODCorr metric in general, we evaluate public available trained models from timm [56] on a few public datasets (CIFAR10 and CIFAR100 [57] and Food101 [58]. In particular we have chosen all pre-trained vit-small models that are available in the timm library which are 17 pre-trained models using a wide range of algorithms from supervised to self-supervised pre-training. Together with the 3 public datasets on natural images, they provide a diverse testing ground to evaluate quality of the ODCorr metric.

Following the same protocol as in [48] we train a linear head on the frozen backbone on the train split of the dataset and compute the top-1 accuracy and the ODCorr on the test split.

As can be seen in Fig. 5 for a given dataset, a lower ODCorr usually corresponds to a higher top-1 accuracy.

2) On whole slide images: We also evaluate the ODCorr metric on different pathology FMs collected from all the above experiments (e.g., for data sizes, for initialization strategies) regardless of how the FM was trained. In particular, we evaluate the relation between linear probing performance and ODCorr, between linear probing and RankMe, as well between ODCorr and RankMe. The linear probe was trained on the respective train split and the results reported on the test split. The ODCorr and RankMe was calculated on the test split only.

In Fig. 6, we first observe in the bottom panel that there is an inverse relation between RankMe and ODCorr where lower ODCorr correlates with higher RankMe, as expected. In addition we also observe that there could be different relations between different model sizes as RankMe depends on the model sizes.

Secondly we observe in the top panel also an inverse relation between the linear probing performance and ODCorr across datasets, where lower ODCorr correlates with higher linear probing performance. Interestingly the linear probing performance plateaus when the ODCorr goes below certain value, e.g., for CRC the linear probing performance does not improve anymore once the ODCorr drops below 0.5, similarly for MHIST although the threshold is higher at around 0.8. For BACH and TP53 the trend is not stopped in all our experiments. For PCam there seems to be a peak at ODCorr at 0.5, and the performance drops with further decreasing ODCorr. However, this could be due to that the fact that the ODCorr is calculated on the test split only, and the linear probing performance is also influenced by the quality of the embeddings of the train split. The existence of the plateau highlights a situation where the ODCorr can provide complementary information to the linear probing situation: for the points on the plateaued part of the curves, the linear probing is no longer able to differentiate between the models as they all show similar performance, in the mean time the ODCorr can still be used to identify better models

In the middle panel we observe the same plateau with linear probing vs. RankMe, i.e., after the RankMe reaches certain number, the linear probing performance stops growing further, in agreement with what we observe with ODCorr. On the other hand, as the RankMe is usually higher with higher embedding dimension, we almost always observe the ViT-L14 models on the right end of the curve, suggesting that they are superior. But this is not necessarily the case, as shown in the upper panel, the ViT-L14 models are not always the best at distinguishing samples as demonstrated by the sometimes higher ODCorr values.

In summary, we believe that ODCorr could be a

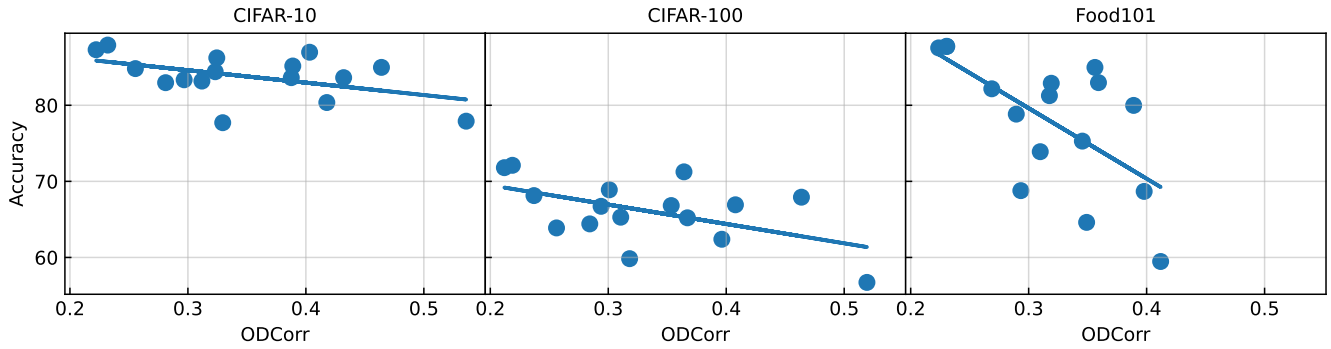

Fig. 5: Correlation between ODCorr and top-1 accuracy of the representation on CIFAR-10 (left), CIFAR-100 (center) and Food101 (right). An inverse correlation between the ODCorr and top-1 accuracy can be observed.

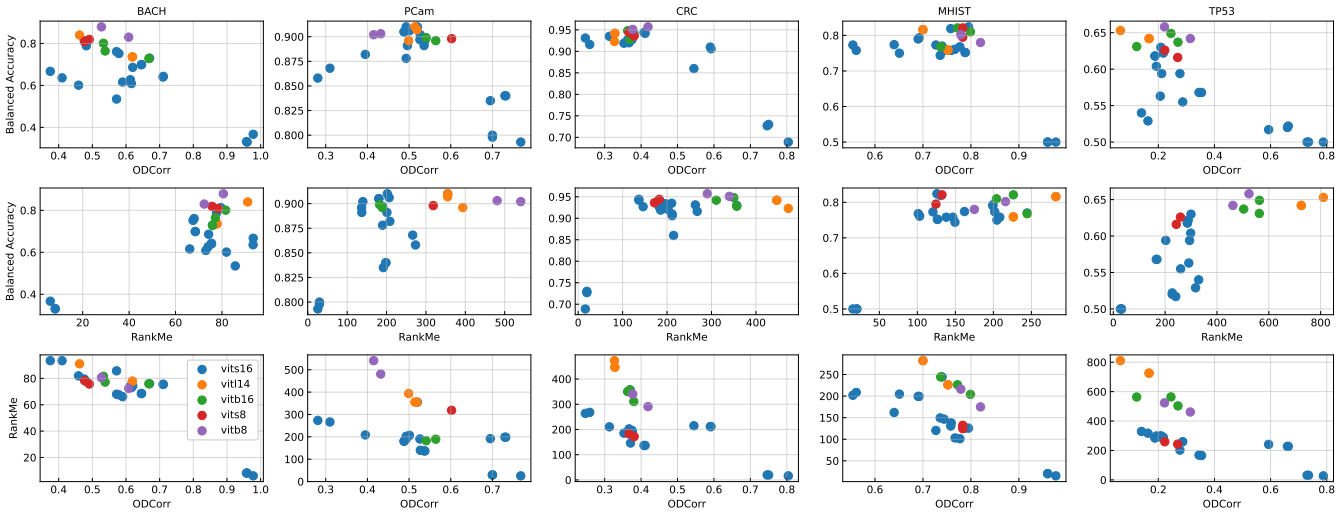

Fig. 6: Relation between linear probing performance, ODCorr and RankMe of the representations on pathology datasets: (upper) Balanced Accuracy vs. ODCorr, (middle) Balanced Accuracy vs. RankMe, (bottom) RankMe vs. ODCorr. Models of different sizes are colored differently

useful unsupervised metric to provide additional information about the FMs, especially when the supervised metrics of downstream tasks do not differ that much anymore. In addition, it could also be useful in comparing models of different sizes.

TABLE VIII: Linear probing evaluation of FMs on patch-level downstream datasets. We report averaged balanced accuracy over 5 linear probing runs and the DICE score on the foreground pixels for the CoNSeP task. Values from Virchow are taken from [33]. All other results were generated using *eva* (except CoNSeP that will be soon supported). We compare the performance from a randomly initialized ViT-S16 model (first line), the generic FMs pre-trained on ImageNet (above the dashed line), and the pathology specific FMs (below the dashed line)

Model	**BACH**	**CRC**	**MHIST**	**PCam/val**	**PCam/test**	**CoNSeP**
ViT-S16 *(rand. weights)*	0.410 (±0.009)	0.617 (±0.008)	0.501 (±0.004)	0.753 (±0.002)	0.728 (±0.003)	0.583 (±0.012)
DINO ViT-S16 [24]	0.695 (±0.004)	0.935 (±0.003)	0.831 (±0.002)	0.864 (±0.007)	0.849 (±0.007)	0.611 (±0.018)
DINO ViT-B8 [24]	0.710 (±0.007)	0.939 (±0.001)	0.814 (±0.003)	0.870 (±0.003)	0.856 (±0.004)	0.710 (±0.005)
Lunit [29]	0.801 (±0.005)	0.934 (±0.001)	0.768 (±0.004)	0.889 (±0.002)	0.895 (±0.006)	0.654 (±0.003)
Phikon [30]	0.725 (±0.004)	0.935 (±0.001)	0.777 (±0.005)	0.912 (±0.002)	0.915 (±0.003)	0.666 (±0.004)
DINO ViT-S16 *(ours)*	0.797 (±0.003)	0.943 (±0.001)	0.828 (±0.003)	0.903 (±0.001)	0.893 (±0.005)	0.649 (±0.013)
DINO ViT-S8 *(ours)*	0.834 (±0.012)	0.946 (±0.002)	0.832 (±0.006)	0.897 (±0.001)	0.887 (±0.002)	0.724 (±0.007)
DINO ViT-B16 *(ours)*	0.810 (±0.008)	0.960 (±0.001)	0.826 (±0.003)	0.900 (±0.002)	0.898 (±0.003)	0.658 (±0.011)
DINO ViT-B8 *(ours)*	0.865 (±0.019)	0.956 (±0.001)	0.809 (±0.021)	0.913 (±0.001)	0.921 (±0.002)	0.741 (±0.002)
DINOv2 ViT-L14 *(ours)*	0.870 (±0.005)	0.930 (±0.001)	0.809 (±0.001)	0.908 (±0.001)	0.898 (±0.002)	0.679 (±0.007)
Virchow [33]	-	0.962	0.830	-	0.933	