



# Exploring the Generalization Potential and Distortion Robustness of Foundation Models in Medical Image Classification by Introducing a New Benchmark for the Multidimensional MedMNIST+ Dataset Collection

**Master Thesis**

Master of Science in Applied Computer Science

Alexander Haas

December 27, 2024

**Supervisor:**

1st: Prof. Dr. Christian Ledig
2nd: Sebastian Dörrich

Chair of Explainable Machine Learning
Faculty of Information Systems and Applied Computer Sciences
Otto-Friedrich-University Bamberg

## Abstract

Over time, benchmarks have been instrumental in driving progress in Deep Learning by offering consistent and reliable frameworks for model evaluation. They promote scientific competition, inspire innovation, and provide researchers with an objective means to assess advancements across various tasks.

This thesis introduces a benchmark specifically designed for image classification in the biomedical domain, focusing on the Multidimensional MedMNIST+ Database. The MedMNIST+ Database is a curated collection of preprocessed, lightweight 2D medical image datasets that support 12 distinct machine learning classification tasks in medical AI research.

The benchmark addresses the unique challenges of medical imaging and aims to drive advancements in model generalization, robustness, and adaptability. It proposes two key challenges. The first is the Generalization Challenge, which evaluates a single model's performance across all 12 classification tasks, emphasizing adaptability to diverse medical imaging datasets. The second is the Robustness Challenge, which assesses model performance on a corrupted version of the dataset (MedMNIST-C) to test resilience against noisy and distorted input data. This challenge addresses issues such as imperfections in data acquisition, processing errors, or real-world variability in medical imaging.

This thesis introduces Dino, Dinov2, UNI, and Prov-GigaPath as baseline models to establish performance standards. These are foundation models — large, pre-trained machine learning models that serve as a versatile base for various downstream tasks across different domains. While Dino and Dinov2 were pre-trained on highly versatile, general-purpose data, UNI and Prov-GigaPath were specifically pre-trained on medical domain data.

To evaluate baseline performance, different training approaches were compared concerning generalization and robustness: Using the pre-trained backbones to evaluate how well the models can perform out of the box without fine-tuning the backbone by utilizing different classifiers. Additionally, two end-to-end training techniques are introduced, utilizing multi-task learning to fine-tune the backbone across all 12 datasets simultaneously.

Pre-trained Dino is found to work best for the MedMNIST+ dataset, showcasing strong generalization across diverse medical imaging tasks. On the other hand, Prov-GigaPath, when fine-tuned with end-to-end training, proves to be the most robust to distortions, demonstrating the most resilience to noisy or corrupted inputs.

The code supporting this thesis is available on GitHub[1].


[1] `https://github.com/ahaas99/ExploringGeneralizationPotential`

## Zusammenfassung

Im Laufe der Zeit haben Benchmarks eine entscheidende Rolle bei der Förderung des Fortschritts im Deep Learning gespielt, indem sie konsistente und zuverlässige Rahmenwerke für die Modellbewertung bieten. Sie fördern den wissenschaftlichen Wettbewerb, inspirieren Innovationen und bieten Forschern ein objektives Mittel, um Fortschritte über verschiedene Aufgaben hinweg zu bewerten.

Diese Thesis führt einen Benchmark speziell für die Bildklassifikation im biomedizinischen Bereich ein, wobei der Fokus auf der Multidimensionalen MedMNIST+ Datenbank liegt. Die MedMNIST+ Datenbank ist eine kuratierte Sammlung vorverarbeiteter 2D-medizinischer Bilddatensätze, die 12 verschiedene Klassifikationsaufgaben im Bereich der medizinischen KI-Forschung unterstützen.

Der Benchmark geht auf die einzigartigen Herausforderungen der medizinischen Bildgebung ein und hat das Ziel, Fortschritte bei der Generalisierung, Robustheit und Anpassungsfähigkeit von Modellen voranzutreiben. Er schlägt zwei zentrale Herausforderungen vor. Die erste ist die Generalisierungs-Challenge, die die Leistung eines einzelnen Modells über alle 12 Klassifikationsaufgaben hinweg bewertet und die Anpassungsfähigkeit an verschiedene medizinische Bilddatensätze betont. Die zweite ist die Robustheits-Challenge, die die Leistung des Modells auf einer korrumpierten Version des Datensatzes (MedMNIST-C) bewertet, um die Resilienz gegenüber verrauschten und verzerrten Eingabedaten zu testen. Diese Herausforderung adressiert Probleme wie Unvollkommenheiten bei der Datenerfassung, Verarbeitungsfehler oder die Variabilität von Bilddaten in der realen Welt.

Diese Thesis stellt Dino, Dinov2, UNI und Prov-GigaPath als Basismodelle vor, um Leistungsstandards festzulegen. Dies sind Foundation-Modelle — große, vortrainierte Maschinenlernmodelle, die als vielseitige Basis für verschiedene nachgelagerte Aufgaben in verschiedenen Bereichen dienen. Während Dino und Dinov2 auf hochgradig vielseitigen, allgemeinen Daten vortrainiert wurden, wurden UNI und Prov-GigaPath speziell auf medizinische Domänendaten vortrainiert.

Um die Baseline-Leistung zu bewerten, wurden verschiedene Trainingsansätze hinsichtlich Generalisierung und Robustheit verglichen: Es wird das vortrainierte Backbone genutzt, um zu bewerten, wie gut die Modelle direkt ohne Feintuning mit verschiedenen Klassifikatoren abschneiden. Zusätzlich werden zwei End-to-End-Trainingstechniken vorgestellt, bei denen Multi-Task-Learning verwendet wird, um das Backbone gleichzeitig über alle 12 Datensätze hinweg zu optimieren.

Es wurde festgestellt, dass das vortrainierte Dino-Modell am besten für den MedMNIST+ Datensatz funktioniert und eine starke Generalisierung über verschiedene medizinische Bildklassifikationsaufgaben hinweg zeigt. Andererseits stellt sich heraus, dass Prov-GigaPath, wenn es mit End-to-End-Training feinabgestimmt wird, am robustesten gegenüber Verzerrungen ist und die größte Resilienz gegenüber verrauschten oder korrumpierten Eingaben zeigt.

Der Code dieser Thesis ist auf GitHub verfügbar[2].

[2] `https://github.com/ahaas99/ExploringGeneralizationPotential`

# Contents

**List of Figures** **v**

**List of Tables** **xii**

**List of Acronyms** **xvi**

**1 Introduction** **1**

1.1 Context and Motivation . . . . . 1

1.2 Related Work . . . . . 3

1.2.1 Benchmarks . . . . . 3

1.2.2 Multi-Task Learning . . . . . 5

1.2.3 Introduction of the Benchmark . . . . . 6

1.3 Contribution . . . . . 6

1.4 Outline . . . . . 7

**2 Theoretical Background** **8**

2.1 Machine Learning . . . . . 8

2.2 Deep Learning . . . . . 10

2.3 Data Augmentations . . . . . 14

2.4 Vision Transformer . . . . . 16

2.5 Foundation Models . . . . . 18

2.5.1 Dino . . . . . 19

2.5.2 Dinov2 . . . . . 21

2.5.3 UNI . . . . . 22

2.5.4 Prov-GigaPath (Prov) . . . . . 24

2.6 Evaluation Metrics . . . . . 24

**3 Methodology** **28**

3.1 Benchmark . . . . . 28

3.2 Training Methods . . . . . 28

3.2.1 Training only the Classification Heads . . . . . 29

3.2.2 Training End-to-End . . . . . 35

**4 Used Dataset** **39**
4.1 MedMNIST+ . . . . . 39
4.2 MedMNIST-C . . . . . 41

**5 Experiments and Results** **45**
5.1 Results for the MedMNIST+ Dataset . . . . . 45
5.1.1 Training without the Backbone . . . . . 49
5.1.2 Training End-to-End . . . . . 54
5.2 Results for the MedMNIST-C Dataset . . . . . 55

**6 Comparison and Discussion** **63**
6.1 Performance on MedMNIST+ . . . . . 63
6.2 Performance on MedMNIST-C . . . . . 66

**7 Limitations and Future Work** **69**
7.1 Training Methodology . . . . . 69
7.1.1 Training only the Classification Heads . . . . . 69
7.1.2 Training End-to-End . . . . . 69
7.1.3 Combination of both Approaches . . . . . 70
7.2 Usage of an additional Metric to evaluate the Robustness to Distortions 70

**8 Conclusion** **72**

**A Appendix** **73**
A.1 Every Plot for the Performance on MedMNIST+ . . . . . 73
A.2 Performance Metrics for every Dataset on MedMNIST+ . . . . . 101
A.3 Plots for Performance on MedMNIST-C . . . . . 125
A.4 Balanced Accuracy Values for mm-PT with Prov-backbone at a Resolution of 224×224 across different severity Levels of MedMNIST-C . 145

**Bibliography** **150**

# List of Figures


1 A foundation model and the tasks it can be generalized to (Bommasani et al., 2021). . . . . 3

2 Categories of ML algorithms (Sarker, 2021b). . . . . 8

3 Typical workflow of machine learning and deep learning (Turing, 2024). 10

4 Architecture of a deep neural network (Parmar, 2018). . . . . 11

5 Schematic representation of the mathematical model of an artifcial neuron (LeCun et al., 2015). . . . . 11

6 Overview over basic data augmentation techniques (Kumar et al., 2023). 15

7 Examples of rotation, flipping, cropping & resize, and noise injection applied (Kumar et al., 2023). . . . . 17

8 Architecture of a vision transformer (Dosovitskiy et al., 2020). . . . . 17

9 Simplified architecture of Dino. The teacher parameters get updated by an exponential moving average (ema), and the gradients are not backpropagated (sg=stop gradient) to keep the teacher framework stable (Caron et al., 2021). . . . . 21

10 Architecture of UNI (Chen et al., 2023). . . . . 23

11 Flow chart of the architecture of Prov-Gigpath. **a**, Prov-GigaPath first transforms each input into 256x256 image tiles and converts each image tile into an embedding. **b**, Image tile-level pre-training using the Dinov2 learning algorithm. **c**, Slide-level pre-training with a LongNet (Xu et al., 2024). . . . . 25

12 Explanation of support vector machines (Javatpoint, 2023). . . . . 29

13 Illustration of the kNN algorithm. In this example, the red squares and green triangles represent two different classes of data points, and the yellow circle with a question mark represents a new point whose class is unknown (Imandoust and Bolandraftar, 2013). . . . . 31

14 Illustration of the random forest algorithm (Liu et al., 2022b). . . . . 32

15 Forming of the decision trees in gradient boosting methods (Zhang et al., 2023b). . . . . 34

16 Illustration of the multi-domain multi-task pre-training algorithm introduced in Woerner et al. (2024). . . . . 36

17 All possible augmentations with flipping and rotation (Sonogashira et al., 2020). . . . . 37

18 An overview over MedMNIST+ (Yang et al., 2023). . . . . 39

19 Four examples of MedMNIST-C. Four distinct corruptions applied to PathMNIST, ChestMNIST, DermaMNIST, and RetinaMNIST, listed from top to bottom. Severity level 1 indicates the lowest level of corruption, while severity level 5 signifies the highest level of distortion (Salvo et al., 2024). . . . . 42

20 Accuracies for SVMs in 128×128 size for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated. . . . . 50

21 Accuracies for LightGBM in 128×128 size for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated. . . . . 51

22 AUCs for linear probing in 64×64 resolution for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated. . . . . 52

23 Balanced accuracies for linear probing in 224×224 resolution for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated. . 53

24 Balanced accuracies for multi-domain multi-task pre-training in 224×224 resolution for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated. . . . . 54

25 Balanced accuracies for multi-domain multi-task pre-training with data augmentation in 224×224 resolution for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated. . . . . 56

26 Balanced accuracies for two of the best training methods at a resolution of 224×224 on the two datasets are illustrated. The top plots represent Dino Linear Probing, while the bottom plots correspond to Prov mm-PT. Performance on MedMNIST+ is shown in the left column, whereas performance on MedMNIST-C is depicted in the right column. . . . . 60

27 Every metric for 28×28 resolution with SVM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 73

28 Every metric for 64×64 resolution with SVM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 74

29 Every metric for 128×128 resolution with SVM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 75

30 Every metric for 224×224 resolution with SVM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 76

31 Every metric for 28×28 resolution with LightGBM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 77

32 Every metric for 64×64 resolution with LightGBM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 78

33 Every metric for 128×128 resolution with LightGBM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 79

34 Every metric for 224×224 resolution with LightGBM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 80

35 Every metric for 28×28 resolution with random forest as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 81

36 Every metric for 64×64 resolution with random forest as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 82

37 Every metric for 128×128 resolution with random forest as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 83

38 Every metric for 224×224 resolution with random forest as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 84

39 Every metric for 28×28 resolution with kNN as classification head on the pre-trained backbones.Test performance indicated in blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 85

40 Every metric for 64×64 resolution with kNN as classification head on the pre-trained backbones. Test performance indicated in blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 86

41 Every metric for 128×128 resolution with kNN as classification head on the pre-trained backbones. Test performance indicated in blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 87

42 Every metric for 224×224 resolution with KNN as classification head on the pre-trained backbones. Test performance indicated in blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 88

43 Every metric for 28×28 resolution with a a linear classifier as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 89

44 Every metric for 64×64 resolution with a linear classifier as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 90

45 Every metric for 128×128 resolution with a linear classifier as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 91

46 Every metric for 224×224 resolution with a linear classifier as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 92

47 Every metric for 28×28 resolution with multi-domain multi-task pre-training for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 93

48 Every metric for 64×64 resolution with multi-domain multi-task pre-training for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 94

49 Every metric for 128×128 resolution with multi-domain multi-task pre-training for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 95

50 Every metric for 224×224 resolution with multi-domain multi-task pre-training for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 96

51 Every metric for 28×28 resolution with multi-domain multi-task pre-training for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 97

52 Every metric for 64×64 resolution with multi-domain multi-task pre-training with data-augmentation for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 98

53 Every metric for 128×128 resolution with multi-domain multi-task pre-training with data-augmentation for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 99

54 Every metric for 224×224 resolution with multi-domain multi-task pre-training with data-augmentation for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated. . . . . 100

55 Every metric for 28×28 resolution with SVM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 125

56 Every metric for 64×64 resolution with SVM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 126

57 Every metric for 128×128 resolution with SVM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 127

58 Every metric for 224×224 resolution with SVM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 128

59 Every metric for 28×28 resolution with LightGBM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 129

60 Every metric for 64×64 resolution with LightGBM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 130

61 Every metric for 128×128 resolution with LightGBM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 131

62 Every metric for 224×224 resolution with LightGBM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 132

63 Every metric for 28×28 resolution with linear probing for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot . . . . . 133

64 Every metric for 64×64 resolution with linear probing for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 134

65 Every metric for 128×128 resolution with linear probing for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 135

66 Every metric for 224×224 resolution with linear probing for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 136

67 Every metric for 28×28 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 137

68 Every metric for 64×64 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 138

69 Every metric for 128×128 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 139

70 Every metric for 224×224 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 140

71 Every metric for 28×28 resolution with multi-domain multi-task pre-training with data augmentation for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 141

72 Every metric for 64×64 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 142

73 Every metric for 128×128 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 143

74 Every metric for 224×224 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot. . . . . 144

# List of Tables


1 Overview of the evaluated model architectures, highlighting their parameter counts, and feature dimensions before the final classification layer. Parameter counts are expressed in millions (M). . . . . 20

2 Details of the MedMNIST+ dataset, including the data modality, classification task type (with the number of classes), and data splits. (ML: Multi-Label, MC: Multi-Class, BC: Binary-Class, OR: Ordinary Regression) (Yang et al., 2023). . . . . 42

3 This table summarizes the chosen types of image corruptions, each implemented at five progressively increasing severity levels. The symbols [+/-] indicate whether the intensity of corruption is increased, decreased, or varied in both directions, resulting in distinct forms of corruption (Salvo et al., 2024). . . . . 43

4 Accuracy values for each dataset of MedMNIST+ on the test split if only the majority class gets predicted. . . . . 45

5 Summary of benchmark results presenting the average mean and standard deviation for accuracy and the area under the receiver operating characteristic curve (AUC) across all datasets, covering all combinations of training schemes, models, and image resolutions. Since the k-NN algorithm lacks a training phase, it is not influenced by the stochasticity inherent in model training and reports only the overall accuracy. The best result for each resolution across all training schemes and models is highlighted with a background color, while the best result for each training scheme and resolution is highlighted in bold. . . . . 46

6 Summary of benchmark results presenting the average mean and standard deviation for balanced accuracy and Cohen´s kappa across all datasets, covering all combinations of training schemes, models, and image resolutions. Furthermore, since k-NN directly uses embeddings and labels for classification, it does not provide a reliable AUC score. The best result for each resolution across all training schemes and models is highlighted with a background color, while the best result for each training scheme and resolution is highlighted in bold. . . . . 48

7 Summary of benchmark results presenting the average mean and standard deviation for accuracy and the area under the receiver operating characteristic curve (AUC) across the 12 corrupted datasets of MedMNIST-C, covering all combinations of training schemes, models, and image resolutions. The best result for each resolution across all training schemes and models is highlighted with a background color, while the best result for each training scheme and resolution is highlighted in bold. kNN and random forest were not evaluated because they performed worse than LightGBM and SVM on MedMNIST+. . . . . 57

8 Summary of benchmark results presenting the average mean and standard deviation for balanced accuracy and Cohen´s kappa across the 12 corrupted datasets of MedMNIST-C, covering all combinations of training schemes, models, and image resolutions. The best result for each resolution across all training schemes and models is highlighted with a background color, while the best result for each training scheme and resolution is highlighted in bold. Since there was no time to evaluate all models, kNN and random forest were not evaluated because they performed worse than LightGBM and SVM on the clean dataset. 59

9 A detailed summary of benchmark results showing balanced accuracy for each dataset in MedMNIST-C and each type of corruption, with the average and standard deviation calculated across all severity levels. The results are based on the best training scheme: Prov mm-PT at resolution 224×224. . . . . 61

10 Performance for accuracy and AUC for **BloodMNIST** across different resolutions, backbones, and training schemes. . . . . 101

11 Performance for balanced accuracy and Cohen´s kappa for **BloodMNIST** for every resolution, every backbone and every training scheme. 102

12 Performance for accuracy and AUC for **BreastMNIST** across different resolutions, backbones, and training schemes. . . . . 103

13 Performance for balanced accuracy and Cohen´s kappa for **BreastMNIST** for every resolution, every backbone and every training scheme. 104

14 Performance for accuracy and AUC for **ChestMNIST** for every resolution, every backbone, and every training scheme. . . . . 105

15 Performance for balanced accuracy and Cohen´s kappa for **ChestMNIST** for every resolution, every backbone and every training scheme. 106

16 Performance for accuracy and AUC for **DermaMNIST** for every resolution, every backbone and every training scheme. . . . . 107

17 Performance for balanced accuracy and Cohen´s kappa for **DermaMNIST** for every resolution, every backbone and every training scheme. 108

18 Performance for accuracy and AUC for **OctMNIST** for every resolution, backbone, and training scheme. . . . . 109

19 Performance for balanced accuracy and Cohen´s kappa for **OctMNIST** for every resolution, every backbone and every training scheme. 110

20 Performance for accuracy and AUC for **OrganaMNIST** for every resolution, every backbone and every training scheme. . . . . 111

21 Performance for balanced accuracy and Cohen´s kappa for **OrganAMNIST** for every resolution, every backbone and every training scheme. 112

22 Performance for accuracy and AUC for **OrgancMNIST** for every resolution, every backbone and every training scheme. . . . . 113

23 Performance for balanced accuracy and Cohen´s kappa for **OrganCMNIST** for every resolution, every backbone and every training scheme. 114

24 Performance for accuracy and AUC for **OrganSMNIST** for every resolution, every backbone and every training scheme. . . . . 115

25 Performance for balanced accuracy and Cohen´s kappa for **OrganSMNIST** for every resolution, every backbone and every training scheme. 116

26 Performance for accuracy and AUC for **PathMNIST** for every resolution, every backbone and every training scheme. . . . . 117

27 Performance for balanced accuracy and Cohen´s kappa for **PathMNIST** for every resolution, every backbone and every training scheme. 118

28 Performance for accuracy and AUC for **PneumoniaMNIST** for every resolution, every backbone, and every training scheme. . . . . 119

29 Performance for balanced accuracy and Cohen´s kappa for **PneumoniaMNIST** for every resolution, every backbone and every training scheme. . . . . 120

30 Performance for accuracy and AUC for **RetinaMNIST** for every resolution, every backbone, and every training scheme. . . . . 121

31 Performance for balanced accuracy and Cohen´s kappa for **retinaMNIST** for every resolution, every backbone and every training scheme. 122

32 Performance for accuracy and AUC for **TissueMNIST** across various resolutions, backbones, and training schemes. . . . . 123

33 Performance for balanced accuracy and Cohen´s kappa for **TissueMNIST** across various resolutions, backbones, and training schemes. . 124

34 A detailed summary of benchmark results showing balanced accuracy for each dataset and each type of corruption for severity 1. The results are based on the best training scheme: Prov mm-PT at a resolution of 224×224. . . . . 145

35 A detailed summary of benchmark results showing balanced accuracy for each dataset and each type of corruption for severity 2. The results are based on the best training scheme: Prov mm-PT at a resolution of 224×224. . . . . 146

36 A detailed summary of benchmark results showing balanced accuracy for each dataset and each type of corruption for severity 3. The results are based on the best training scheme: Prov mm-PT at a resolution of 224×224. . . . . 147

37 A detailed summary of benchmark results showing balanced accuracy for each dataset and each type of corruption for severity 4. The results are based on the best training scheme: Prov mm-PT at a resolution of 224×224. . . . . 148

38 A detailed summary of benchmark results showing balanced accuracy for each dataset and each type of corruption for severity 5. The results are based on the best training scheme: Prov mm-PT at a resolution of 224×224. . . . . 149

# List of Acronyms

| | |
|---|---|
| AI | Artificial Intelligence |
| ML | Machine Learning |
| CNN | Convolutional Neural Networks |
| NLP | Natural Language Processing |
| MTL | Multi-Task Learning |
| GLUE | General Language Understanding Evaluation |
| DNN | Deep Neural Network |
| BCEwithLogits | Binary Cross-Entropy with Logits |
| DL | Deep Learning |
| ViT | Vision Transformer |
| Prov-GigaPath | Prov |
| SVM | Support Vector Machine |
| kNN | k-Nearest Neighbors |
| LightGBM | Light Gradient Boosting Machine |
| mm-PT | multi-domain multi-task pre-training |
| mm-PT aug | multi-domain multi-task pre-training with data augmentation |
| RF | Random Forest |
| TP | True Positives |
| FP | False Positives |
| TN | True Negatives |
| FN | False Negatives |
| TPR | True Positive Rate |
| FPR | False Positive Rate |
| Acc | Accuracy |
| Bal_Acc | Balanced Accuracy |
| Co | Cohen´s Kappa |
| AUC | Area Under the Curve |

# 1 Introduction

## 1.1 Context and Motivation

The growth of Artificial Intelligence (AI) over the past few decades has been remarkable. This growth is driven by advances in algorithms and machine learning models (Krizhevsky et al., 2012), computational power (Hooker, 2020), and access to large datasets (Manyika et al., 2011), (Brynjolfsson and McAfee, 2017). AI has evolved from a theoretical research topic to a transformative technology that is now embedded in countless industries, products, and services (Brynjolfsson and McAfee, 2017), (Jordan and Mitchell, 2015).

But what is AI exactly: Russell and Norvig (2010) defines it as "the study of agents that receive percepts from the environment and perform actions". McCarthy (2004) defines AI as "... the science and engineering of making intelligent machines, especially intelligent computer programs", while Kaplan and Haenlein (2019) defines AI as following: "Artificial Intelligence is a system's ability to interpret external data correctly, to learn from such data and to use those learnings to achieve specific goals and tasks through flexible adaptation". So AI is a system that learns to do a specific task and it achieves that by learning from a given data source.

There are many different subsets of AI, such as Machine Learning (ML), Natural Language Processing (NLP), Robotics, and many more (Russell and Norvig, 2010). Within the scope of this thesis, the primary emphasis is placed on ML. ML focuses on developing algorithms and models that enable computers to perform a specific task. The main idea is that, instead of being programmed manually to perform the task, machines are trained to learn from data to perform their specific task (Mitchell, 1997). In the case of this thesis, the task is image classification. More precisely, the task is to train one model on the MedMNIST+ dataset, a medical dataset that contains 12 different types of biomedical images (Yang et al., 2023). All images are labeled with the different diseases that are or are not shown in them. The goal is to have **one** model that can classify **all 12** classification tasks of the dataset and achieve a good performance by doing so.

What is the motivation behind the idea of using only one model for all tasks instead of a single model for every task specifically? Developing countries are struggling a lot more with good health and medical resources than other countries. Not only do developing countries represent the majority of the world population, but they also host almost 90 % of global diseases (Alemayehu et al., 2018). So these countries would benefit the most from models that can diagnose diseases based on images. However, designing 12 models that can diagnose only their designated task is a lot more expensive than designing one model that can classify all tasks at once. The reason for this is higher computational costs and more storage to run all models individually (Zhang and Yang, 2017). In addition, more development time is needed to implement each model and train each model individually. So, there is a need for

such models that learn using a "Multi-Task Learning"(MTL)-Technique and are able to classify medical images.

But in real-world applications the data is often noisy or incomplete. That is because various factors, such as sensor malfunctions, data transmission errors, or poor image quality, can lead to data corruption. Testing models on corrupted data helps to evaluate the robustness and generalization potential of the model and provides essential information on whether a model is capable of performing well on imperfect and noisy inputs (Ciresan et al., 2011). Ciresan et al. (2011) also shows that the evaluation of corrupted data is especially relevant in computer vision tasks such as image classification. Hendrycks and Dietterich (2019) highlights that testing models on corrupted data improves the model stability. The paper shows how models respond to unstable or missing input and help to handle unexpected data scenarios. So both models on a clean as well as on a corrupted dataset are evaluated to improve overall model quality.

The performance of the models should serve as a baseline of a newly created benchmark that is part of this thesis. Benchmarks play an enormous role in the realm of machine learning. Since ML is a rapidly evolving field, benchmarks serve as a way to measure, compare, and evaluate the performances of models across various different tasks. Thiyagalingam et al. (2022) highlights how benchmarks are essential in identifying the best ML algorithms for complex scientific datasets. This quote from Zaharia et al. (2024) shows some important aspects of why benchmarks are needed in ML: "Benchmarks allow us to quantitatively know the capabilities of different models, software, and hardware. They allow ML developers to measure the inference time, memory usage, power consumption, and other metrics that characterize a system. In addition, benchmarks create standardized measurement processes, allowing fair comparisons between different solutions." Thiyagalingam et al. (2022) and Zaharia et al. (2024) show that benchmarks help drive progress in both industrial and scientific applications by promoting improvements in model accuracy, efficiency and, robustness.

The models this thesis suggests as a baseline are the so-called foundation models. A foundation model refers to a large-scale, pre-trained model that serves as the basis for various downstream tasks in ML. As Figure 1 shows, foundation models can be pre-trained by different data sources like speech, texts, images, and so on. This results in a pre-trained model on a large amount of diverse data. This model is versatile and can be fine-tuned for a wide range of tasks, such as question-answering, information extraction, object recognition, and more. So, these models are used as a "Foundation" for creating new models for many different tasks (Bommasani et al., 2021), (Radford et al., 2021b). In the case of this thesis, they are used as a foundation for an image classification task. Since these models are already pre-trained on a wide range of different data, they are able to perform the underlying task with minimal modification using a smaller, domain-specific dataset, while having a good performance (Bommasani et al., 2021). This is a great opportunity for domains where task-specific data is limited, for example in healthcare (Bommasani et al., 2021). That makes foundation models the perfect choice as a model for the

underlying medical dataset and image classification tasks and further for the baseline of the benchmark that this thesis introduces.

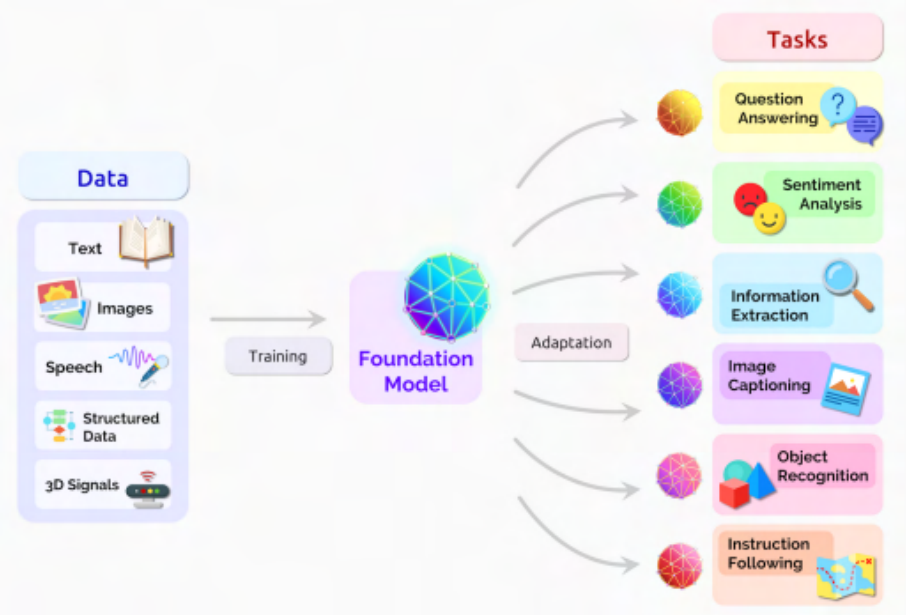


Figure 1: A foundation model and the tasks it can be generalized to (Bommasani et al., 2021).

## 1.2 Related Work

This section reviews related work in the areas of benchmarks and Multi-Task Learning (MTL) within the field of machine learning. It begins by presenting an overview of existing datasets and their associated benchmarks, followed by a discussion of state-of-the-art MTL methods pertinent to this thesis.

### 1.2.1 Benchmarks

Wang et al. (2018) introduced General Language Understanding Evaluation (GLUE). It is a benchmark that evaluates models in a suite of NLP tasks, such as sentiment analysis, sentence similarity and natural language inference. Wang et al. (2019) describes GLUE´s successor: SuperGLUE. SuperGLUE includes more difficult tasks, including reading comprehension and coreference resolution and is designed to be more challenging for models that have excelled on GLUE. This also shows the need for new and more challenging benchmarks to support and increase the rapid development of new and innovative machine-learning models and algorithms. DecaNLP is a multi-task benchmark that includes 10 different NLP tasks, such as machine translation, summarization, and question answering. The goal is to evaluate models on their ability to handle diverse NLP tasks within a single architecture. This

paper demonstrates the importance to "explore models that generalize to many different kinds of NLP tasks. decaNLP encourages a single model to simultaneously optimize for ten tasks" (McCann et al., 2018). Although this quote highlights the significance of NLP architectures, its relevance extends to other areas of machine learning, such as image classification, as demonstrated in this thesis. These three examples represent benchmarks in the NLP domain that utilize multiple tasks to assess the performance of a single model.

But there are also examples of such datasets and benchmarks in the field of Computer Vision: Visual Taks Adaptation Benchmark (VTAB) evaluates models' generalization ability across a wide variety of vision tasks. It includes datasets from areas like natural images, specialized tasks, and real-world scenarios, making it a comprehensive benchmark for multi-task vision learning (Zhai et al., 2019). The Medical Segmentation Decathlon emphasizes the importance of standardized assessments for medical image segmentation tasks involving various anatomical structures. The results of this challenge demonstrate that the achievement of consistent performance across multiple tasks is a strong indicator of overall robust generalizability (Antonelli et al., 2022).

CheXpert is a large-scale dataset of chest radiographs labeled for the presence of 14 different conditions (Irvin et al., 2019). PADChest, for example, includes a wide variety of labels for conditions detected on chest radiographs (Bustos et al., 2020). Numerous individual datasets focus on specific diseases and conditions, such as the two mentioned, which can be used to evaluate model performance. However, far fewer datasets are designed for benchmarking a model across multiple datasets and tasks, enabling the comparison of a model's ability to generalize across diverse tasks. MedFMC (Wang et al., 2023) is one of them. It presents a dataset, which was developed to address the challenge of adapting foundation models for the classification of medical images. This dataset contains 22,349 images across five different medical imaging tasks, collected from multiple healthcare institutions. These tasks include:

1. ChestDR: Thoracic disease screening using chest X-rays
2. ColonPath: Lesion detection in pathology images for cancer screening
3. Endo: Lesion detection in endoscopy images
4. NeoJaundice: Neonatal jaundice evaluation based on skin photos
5. Retino: Diabetic retinopathy classification using retinal images

The MedFMC dataset and its benchmarks aim to test the effectiveness of large-scale foundation models when applied to real-world clinical tasks. It supports the evaluation from both the accuracy and the cost-effectiveness perspective, providing a robust framework for improving medical AI systems. The same can be said for Yang et al. (2023). They introduce the MedMNIST v2 dataset, which is explained in detail later since it is the foundational dataset for this thesis. A dataset consisting of 12 2D classification tasks and 6 3D tasks. Furthermore, there are 4 different resolutions:

28×28, 64×64, 128×128, and 224×224, which makes this dataset special. They benchmark different models on each dataset individually.

Another benchmark worth noticing is the WILDS-benchmark: A benchmark of in-the-Wild Distribution Shifts (Koh et al., 2021). It focuses on evaluating models across diverse domains under realistic distribution shifts. The benchmark introduces a wide range of diverse datasets, each of which captures real-world shifts in data distribution. As a widely recognized and respected benchmark in the realm of ML, the design of the website for our benchmark is highly influenced by this benchmark.

### 1.2.2 Multi-Task Learning

The following section introduces related work in regard to multi-task learning methods used to train different machine learning models.

Argyriou et al. (2007) proposed a technique to improve multi-task learning by encouraging tasks to share relevant features. This method identifies and learns a shared set of features that are useful across different tasks, promoting efficient and collaborative learning through shared representations. The approach is particularly useful in scenarios where related tasks can benefit from overlapping input information.

Collobert and Weston (2008) introduced a deep neural network architecture that jointly learns multiple natural language processing tasks, such as tagging part of speech and recognition of named entities. By sharing representations across tasks, their model leverages task similarities, improving performance on individual tasks. This approach was remarkable for NLP as it demonstrated how multi-task learning could streamline various NLP tasks into a single, efficient framework, reducing the need for separate models.

Liu et al. (2019) proposed a multi-task learning model specifically designed for various natural language understanding tasks, such as sentiment analysis and natural language inference. By sharing parameters across tasks, their model efficiently learns both shared and task-specific features, resulting in improved performance on individual tasks. This paper demonstrated the effectiveness of Multi-Task Learning in NLP by enabling related tasks to reinforce each other, enhancing overall model generalization and robustness.

He et al. (2021) and Graham et al. (2023) introduced models that use multi-task learning in image classification tasks, demonstrating the advantages of the technique. He et al. (2021) proposed a method to adaptively combine multiple convolutional neural networks (CNNs) for large-scale image classification tasks. By grouping image categories with semantic correlations into a hierarchical ontology, the model efficiently handles complex tasks while maintaining competitive accuracy on datasets like ImageNet10K. This work highlights the scalability of deep-mixture models for tasks with thousands of categories.

Graham et al. (2023) introduced the Cerberus model for MTL, which simultaneously predicts multiple histological tasks, such as segmentation and classification, using a shared ResNet34 encoder and independent decoders. The model demonstrates

improved feature generalization and data efficiency, leveraging transfer learning to boost performance for downstream tasks like object detection. This framework addresses challenges like task-specific data variations and gradient conflicts during optimization.

As these papers show, shared representations and multi-task learning can help enhance the overall performance of models in different fields of machine learning. Thus, the multi-task learning approach is a suitable method to examine in this thesis and to get the advantages this method brings with it. The multi-task learning approach suggested by this thesis is from Woerner et al. (2024) and will be introduced later on.

### 1.2.3 Introduction of the Benchmark

The benchmark of this thesis ties in on the MedMNIST+ dataset and work, since it uses the 2D collection of datasets introduced by Yang et al. (2023). Therefore, the proposed benchmark primarily emphasizes image classification within the biomedical domain. A comprehensive description of the MedMNIST+ dataset collection will be provided later. Notably, the MedMNIST+ collection encompasses images from various biomedical domains, supporting diverse imaging tasks such as binary classification, multi-class classification, multi-label classification, and ordinal logistic regression.

This presents the challenge of developing a model that performs well across diverse tasks. MedMNIST+ includes 12 datasets that span various imaging modalities such as X-rays, microscopy, and CT, with tasks that cover different regions of the body. Considering these variations in modalities, tasks, and medical domains, a key objective of our benchmark is to assess a model's generalization capability.

Another key aspect of the benchmark is the robustness to distortions. The benchmark also gives the opportunity to evaluate the models on a corrupted version of the dataset: MedMNIST-C introduced by Salvo et al. (2024). This paper introduces different corruptions on MedMNIST+ that can appear in real-world scenarios.

This thesis establishes a baseline using foundation models applied out of the box, without fine-tuning, to evaluate their pre-trained capabilities. Additionally, these models are fine-tuned using a Multi-Task Learning approach to investigate whether fine-tuning enhances their generalization potential and robustness to distortions.

## 1.3 Contribution

Key Contributions of this thesis:

- **Evaluation of foundation models**:
  - Investigate the performance of foundation models on multi-domain medical image classification tasks.

  - Explore whether these models generalize across tasks with limited task-specific data.

- **Multi-task learning techniques**:
  - Evaluate the impact of fine-tuning foundation models on medical image datasets using a Multi-Task Learning technique.
  - Analyze changes in performance and robustness.

- **Robustness to real-world distortions**:
  - Test model performance on distorted images (e.g., noise, blur) to simulate real-world clinical scenarios.
  - Emphasize the critical need for robustness of models when deploying solutions in healthcare settings.

- **Set a new benchmark**:
  - Establish a baseline for the MedMNIST+ dataset, facilitating model comparisons based on generalization performance.
  - Establish a baseline for the MedMNIST-C dataset, allowing model comparisons on noisy or corrupted data.
  - Encourag the development of more robust models for medical applications.

In summary, this thesis provides a comprehensive benchmark that evaluates models for medical image classification, multi-task learning, and robustness to distortions, offering valuable insights for the development of reliable AI in healthcare.

## 1.4 Outline

The remainder of this thesis is structured as follows. Chapter 2 provides the technical foundation necessary for understanding the subsequent chapters. Chapter 3 outlines the methodology employed to train the models, while Chapter 4 describes the datasets used for training and evaluation. Chapter 5 presents the performance of the different models, and Chapter 6 compares and discusses these results. Chapter 7 addresses the limitations and suggests future work, while Chapter 8 offers a conclusion.

# 2 Theoretical Background

This section gives an overview of the theoretical foundation needed to understand this thesis. It starts by introducing machine learning and deep learning (DL) and everything that comes with it. After that, foundation models are generally explained, and the four foundation models used in this thesis are introduced. This section closes with the definition of the evaluation metrics used to measure the performance of the models.

## 2.1 Machine Learning

ML systems automatically learn from data and experience without being specifically programmed and allow the system "to function in an intelligent manner"(Sarker, 2021b). The effectiveness and efficiency of these models are based on the nature and characteristics of the data the model learns from and the performance of the learning algorithm used. Since there is a very wide range of different tasks, data, and algorithms, it can be challenging to find a good algorithm for a specific domain (Sarker, 2021b). The existing algorithms can be categorized into four categories

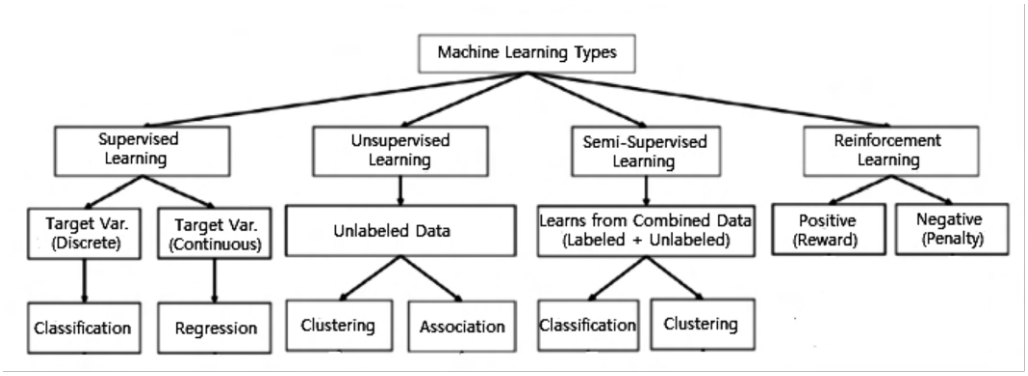


Figure 2: Categories of ML algorithms (Sarker, 2021b).

as shown in figure 2: Supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

Supervised Learning: Supervised learning is the most common form of ML. Usually, it tackles the task, which maps input to output. It learns from labeled input-output pairs. Common supervised learning tasks are classification tasks that separate the data. A typical classification task can be the classification of an image into different classes, e.g., into "pictures of a dog" and "not pictures of a dog"(Sarker, 2021b), (LeCun et al., 2015).

Unsupervised Learning: Unlike supervised learning, unsupervised learning analyzes unlabeled data and does not need human support. This mainly intends to extract generative characteristics, identify meaningful trends and structures, groupings in the results, and exploratory purposes (Sarker, 2021b).

Semi-Supervised Learning: This technique can be defined as a hybrid form of the two techniques just described since it can deal with both labeled and unlabeled data. That is especially useful in real-world scenarios where the labeled data is rare, and a semi-supervised model can be useful. It aims to outperform supervised models restricted to learning only on the labeled data. Machine translation, fraud detection, labeling data, and text classification are the main areas where semi-supervised models find use (Sarker, 2021b).

Reinforcement Learning: According to Sarker (2021b) "Reinforcement learning is a machine learning algorithm that enables software agents and machines to automatically evaluate optimal behavior in a particular context or environment to improve efficiency." This learning method uses a reward and penalty system. It aims to take an action that maximizes the reward or minimizes the risk. It is used in robotics, autonomous driving tasks, manufacturing, and supply chain logistics (Sarker, 2021b).

Self-Supervised Learning: Self-supervised learning is an additional machine learning approach, not included in the figure, where models learn to extract meaningful features from unlabeled data. This can, for instance, be achieved by solving pretext tasks, which generate supervisory signals directly from the data itself, removing the need for manually annotated labels (Jaiswal et al., 2021).

Although there are different learning techniques, the main focus of this thesis is Supervised Learning since the observed task is image classification. Three common classification problems are explained in the following paragraph. All three problems are part of the dataset used in this thesis.

The first classification problem is binary classification. It is a classification task with exactly two class labels, e.g., "yes" and "no", "true" and "false" or "ill" and "healthy". One of the classes can be considered the normal state while the other class can be the abnormal class (Sarker, 2021b).

Multiclass classification describes those classification tasks that have more than two class labels. It does not work after the principle of normal and abnormal. Instead, it operates in a range of specified classes, while an example is categorized as one of these classes (Sarker, 2021b). It can be such a task to classify various types of disease in a sample, e.g., "normal", "COVID-19", "Other pneumonia"and "Tuberculosis" as in Punn and Agarwal (2020).

Multi-label classification describes a problem where an example is relevant to several classes or labels. It is a matter of generalization of the multiclass classification. In this problem, the classes involved are hierarchically structured, and each sample can belong to more than one class (Sarker, 2021b).

However, machine learning algorithms for image classification cannot classify the input immediately. There is a step called feature extraction in between, as Figure 3 demonstrates. In the top half of the figure, the typical workflow of machine learning is depicted. An input is provided for the model to classify; in this case, an image of a car is used. This image must be preprocessed, and its features must be extracted. Feature extraction is a step where raw data is transformed into a more manageable and meaningful form. In machine learning, feature extraction aims to enhance

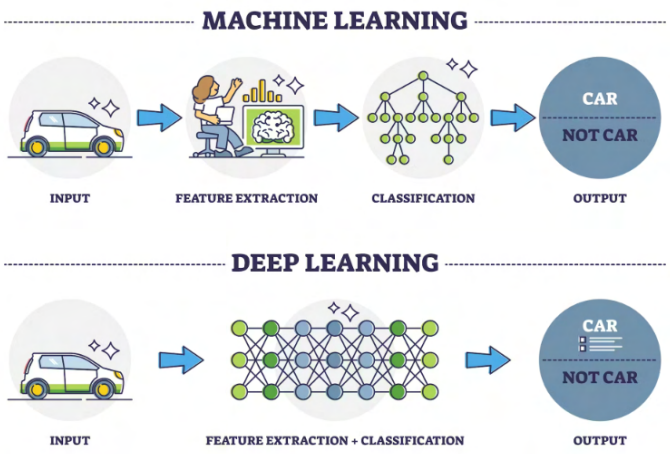


Figure 3: Typical workflow of machine learning and deep learning (Turing, 2024).

the predictive accuracy of models by selecting or deriving new features that better capture the underlying patterns in the data. Methods for feature extraction range from statistical approaches (like principal component analysis) to machine learning-specific techniques such as autoencoders (Li et al., 2016), (Berahmand et al., 2024). Then, these features are passed into the machine learning algorithm that gives an output prediction. In this example, a binary classification decides if this image is an image of a car or not. In the bottom half of figure 3, you can see the workflow of a deep learning model. As you can see, it only needs the image as input since it can extract the features of the images by itself. The machine learning algorithms this thesis uses get their feature extractions from foundation models, which are deep learning architectures. Deep learning and foundation models are introduced in the next chapter.

## 2.2 Deep Learning

Deep learning and deep neural networks (DNN) have emerged as an important branch of machine learning. They find use in various domains like healthcare, visual recognition, text analytics, and many more, particularly in tasks that require complex pattern recognition and data representation (Sarker, 2021a). As the term neural networks suggests, its origin is in neuroscience, and parallels can be drawn between the human brain's decision-making and DNNs (Hassabis et al., 2017). In figure 4, the architecture of a DNN is shown. DNNs have a multi-layer architecture. Each of these layers is designed to process different aspects of the input data, while the collection of all layers determines the final output together (LeCun et al., 2015). A typical DNN has an input layer, multiple hidden layers, and an output layer. Each of the layers is composed of many simple processing elements or neurons. These neurons are connected to neurons in other layers. Each neuron generates a series of activations for the target outcome.

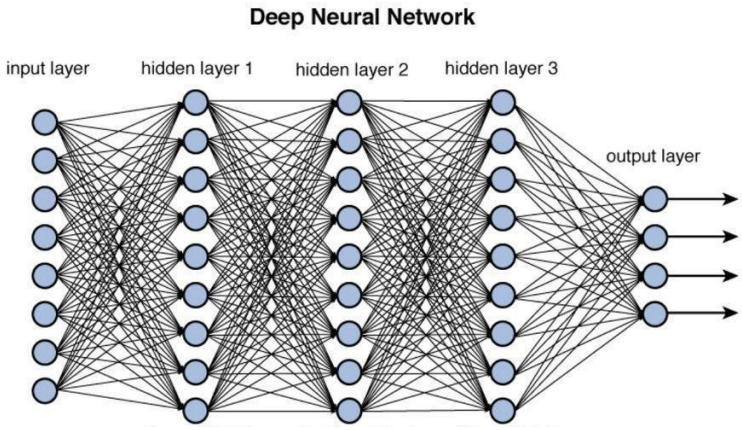


Figure 4: Architecture of a deep neural network (Parmar, 2018).

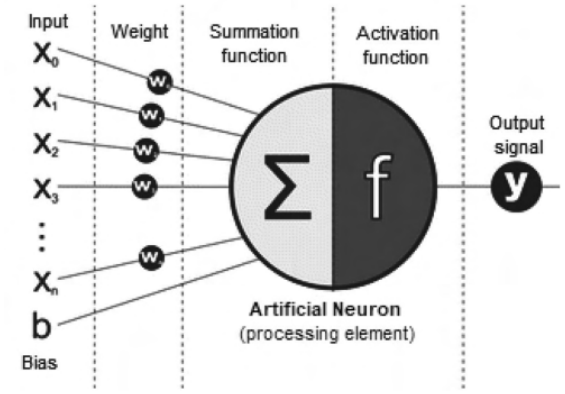


Figure 5: Schematic representation of the mathematical model of an artifcial neuron (LeCun et al., 2015).

Figure 5 shows the schematic representation of the mathematical model of such a neuron. There is the input($\boldsymbol{x}$), the weight($\boldsymbol{w}$), a bias($b$), a summation function($\sum$), an activation function($f$), and an output signal($y$). Each neuron gets the outputs of the neurons of the previous layer as input $\boldsymbol{x}$ if they are connected. This input is weighted with $\boldsymbol{w}$. That means each neuron weights the information (input) from previous neurons differently. After that, a bias $b$ is added to the weighted sum of the input and the weights. That together gives the activation $z$ as shown in equation 1 (Zhang et al., 2023a), (Montavon et al., 2018).

$$z = \boldsymbol{x}\boldsymbol{w} + b \tag{1}$$

This activation $z$ goes into an activation function $f$, and the output of this function is the output of the neuron that is given to the subsequent neurons. This shows that the weights and biases of the neurons are crucial for the network's output and, therefore, for the model's performance. So, these DNNs are iteratively trained, and the weights and biases are updated to minimize errors. The error or loss of a DNN is the difference between the true output and the actual output that the model gives.

There are many loss functions, each suiting different data types and tasks (Zhang et al., 2023a). This thesis mainly uses cross-entropy loss (Doerrich et al., 2024b).

It is applied to the logits and is defined as follows:

$$\text{CE} = -\frac{1}{N}\sum_{n=1}^{N}\log\left(\frac{\exp(z_{n,y_n})}{\sum_{c=1}^{C}\exp(z_{n,c})}\right) \tag{2}$$

Where:

- $N$ is the number of samples in the current batch,
- $C$ is the total number of classes,
- $z_{n,c}$ represents the logit for class $c$ of the $n$-th sample and
- $z_{n,y_n}$ corresponds to the logit of the target class for the $n$-th sample.

For binary classification, where $C = 2$, this equation reduces to the binary cross-entropy loss (Doerrich et al., 2024b).

Another significant loss for this paper is the binary cross-entropy with logits (BCEwithLogits) loss. One of the classification problems of this thesis is a multi-label classification task. This thesis reframes it as a multi-label binary classification problem, where the objective is to predict the presence or absence of each class label $c$ for a given sample $n$, similar to Doerrich et al. (2024b). The Binary Cross-Entropy with Logits (BCEwithLogits) loss function is applied across all class labels $c \in C$. The loss function is defined as:

$$\text{BCEwithLogits} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{c=1}^{C}\left[y_{n,c}\log\sigma(z_{n,c}) + (1-y_{n,c})\log(1-\sigma(z_{n,c}))\right] \tag{3}$$

where $N$ represents the number of samples in the current batch, $z_{n,c}$ is the logit corresponding to sample $n$ and class $c$, $y_{n,c}$ is the binary label for sample $n$ and class $c$, indicating the presence (1) or absence (0) of class $c$, and $\sigma(\cdot)$ is the sigmoid activation function applied to the logit $z_{n,c}$ (Doerrich et al., 2024b).

To minimize the network loss, it is necessary to understand how the loss function is affected by the current weights and biases of the model. This information is contained in the gradient (Zhang et al., 2023a). Here comes backpropagation into play. Backpropagation is a technique used to compute the parameters' gradients in a neural network. It works by moving backward through the network, starting from the output layer and proceeding to the input layer, following the chain rule from calculus. During this process, the algorithm saves intermediate values (such as partial derivatives) needed for gradient calculation concerning the parameters (Zhang et al., 2023a). For example, if we have functions $y = f(x)$ and $z = g(y)$,

we can use the chain rule to determine the derivative of $y$ concerning $x$ with this equation:

$$\frac{\partial y}{\partial x} = \frac{\partial g}{\partial f}\frac{\partial f}{\partial x} \tag{4}$$

Backpropagation computes the gradients of the neurons' parameters and their impact on the loss. These gradients guide optimization algorithms, such as gradient descent, as they train neural networks.

This leads to the next important topic in Neural Networks: Optimization Algorithms. An optimization algorithm is a method used to adjust the parameters of a model. Optimization algorithms work by using the gradients of the loss function concerning the model's parameters to update these in the direction that reduces the loss (Loshchilov and Hutter, 2017). This is often done through iterative updates, meaning the algorithm repeatedly adjusts the parameters step by step, moving toward an optimal solution. This thesis uses the AdamW optimization algorithm. AdamW is an optimization algorithm that improves upon the Adam optimizer by addressing the issue of weight decay regularization. In deep learning, weight decay (L2 regularization) is commonly used to prevent overfitting by penalizing large weights, thereby encouraging smaller and more generalizable model parameters (Loshchilov and Hutter, 2017).

All these techniques can now be combined to train the models. The following four steps are here to summarize the learning of a model:

1. **Forward Pass**: Input data is passed through the network layer by layer, where each neuron applies a weighted sum of its inputs followed by a nonlinear activation function. This produces an output from the network.

2. **Loss Calculation**: The output is compared to the actual target values using a loss function, quantifying how far off the predictions are from the actual values (Zhang et al., 2023a).

3. **Backward Pass**: This is where backpropagation comes into play. The algorithm calculates the gradient of the loss function with respect to each weight in the network by applying the chain rule of calculus. It determines how the error changes with small changes in the weights (Zhang et al., 2023a).

4. **Weight Update**: The weights are updated to minimize the loss. A learning rate dictates here the size of the steps the algorithm takes when moving toward the minimum of the loss function. Selecting an appropriate learning rate is crucial for effective model training, as it balances the speed of convergence with the risk of overshooting the global minimum (Bishop, 2006; Kingma and Ba, 2015).

These four steps are repeated until a termination criterion is met. An epoch is one complete cycle through the training dataset. One termination criterion can be a predefined number of epochs. Usually, the dataset is split into training, validation,

and test sets. The training set is the most significant portion of the dataset and is used to train the model. The validation set helps monitoring the model's performance and ensures it does not overfit the training data. The test set assesses the model's final performance after training is complete (Nielsen, 2015). The test set must be kept entirely separate from both the training and validation sets to provide an unbiased evaluation of how well the model generalizes to new, unseen data. Overfitting in this context is a problem where a model learning learns the data too well instead of capturing the underlying structures. This results in a model that performs very well on the training data but poorly on the validation and test set. To prevent overfitting, models can implement different strategies, e.g., Early Stopping (Nielsen, 2015).

When training large models with enough representational capacity to overfit the task, it is common to see a steady decline in training error over time while the validation set error increases again. This implies that a model with improved validation set error (and potentially better test set error) can be trained by reverting to the parameter configuration when the validation set error is lowest. Each time the validation set error improves, a copy of the model parameters gets saved. Upon completing the training algorithm, these saved parameters are restored instead of using the most recent ones. An Early Stopping criterion can be defined as a fixed number of epochs during which the model fails to improve its validation set error. If a model trains for n epochs without improvement in the validation loss, the training gets stopped, and the parameters of the best-performing model get saved (Nielsen, 2015).

In practice, calculating the gradients for the entire dataset is typically avoided due to memory constraints. So, the dataset gets split into batches. The batch size refers to the number of training samples utilized in a single iteration of model training during mini-batch gradient descent, where the training data is divided into random mini-batches for iterative parameter updates. Larger batches yield a more precise estimate of the gradient but demand more memory during training. Conversely, smaller batches may improve the model's generalization capabilities due to the regularizing effect of the noise present in smaller batches. However, they can also slow down the overall training process (Goodfellow et al., 2016).

## 2.3 Data Augmentations

Since good, diverse, and large training data is essential for training machine learning models, data augmentations are critical for ML. It involves generating additional training data from an existing dataset by applying transformations, effectively expanding the dataset's diversity. These modifications help prevent overfitting and improve a model's ability to generalize to new data by exposing it to varied examples during training.

Figure 6 overviews fundamental transformations. Geometric data augmentation involves altering an image's geometric properties, such as its position, orientation, and

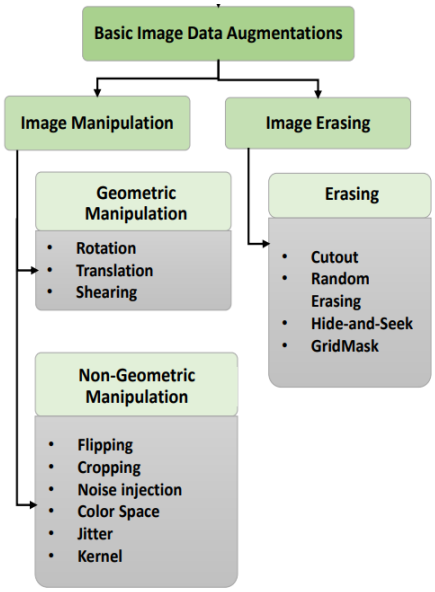


Figure 6: Overview over basic data augmentation techniques (Kumar et al., 2023).

aspect ratio. This can include transformations like rotation, translation, scaling, and flipping, which adjust the spatial layout of objects within the image to increase data variability and improve model robustness to changes in viewpoint and alignment.

**Rotation**. Rotation is a geometric transformation applied to images to turn them around a specific point, usually the image's center. The angle of rotation can vary, often measured in degrees, to simulate different viewpoints or perspectives of the objects within the image. For instance, rotating an image by 90 degrees clockwise changes its orientation without altering the content itself (Kumar et al., 2023).

**Translation**. Translation data augmentation involves shifting an image in any direction — upward, downward, left, or right — by a certain number of pixels. This technique alters the object's position within the frame while preserving its original appearance, helping the model learn to recognize objects regardless of where they appear within the image (Kumar et al., 2023).

**Shearing**. Shearing data augmentation involves shifting one part of an image in a specific direction while the opposite part shifts in reverse, creating a slanted or skewed effect. This transformation provides new and varied perspectives of the image data (Kumar et al., 2023).

Non-geometric data augmentation emphasizes enhancing an image's visual properties — such as color balance, brightness, contrast, saturation, or noise — without changing its spatial structure or layout. This approach diversifies the dataset by introducing variations in visual appearance, helping the model become resilient to changes in lighting, color shifts, and other visual conditions that can occur in real-world scenarios. (Kumar et al., 2023).

**Flipping**. Flipping is an image data augmentation technique that involves mirroring an image horizontally (left to right) or vertically (top to bottom). This transformation creates a reversed version of the original image (Kumar et al., 2023).

**Cropping and Resizing**. This technique involves cropping a portion of the image and then resizing it to its original dimensions while maintaining the original label. Cropping as a data augmentation method introduces variety by zooming into different parts of an image, which can help the model become more robust to partial views of objects. However, improper cropping might result in missing essential features (Kumar et al., 2023).

**Noise Injection**. Noise refers to random variations in brightness or color information that are often unwanted distortions (Kumar et al., 2023).

**Color Space**. This data augmentation adjusts the image's brightness and overall color balance (Kumar et al., 2023).

**Jitter**. Jitter is used for changing image features like brightness, contrast, saturation, and hue (Kumar et al., 2023).

**Kernel Filtering**. Kernel filtering is a technique used in data augmentation that enhances or softens the image. An example of such an augmentation is Gaussian-blur (Kumar et al., 2023).

There are also image erasing data enhancements. As the name suggests, these augmentations remove specific parts of a picture by replacing them with either 0, 255, or the mean of the dataset. Cutout, random erasing, and grid mask data augmentation are examples (Kumar et al., 2023).

Figure 7 shows examples of rotation, flipping, cropping & resize, and noise injection, which are the most important augmentations for this thesis. In addition, there are a lot more data augmentations that are more advanced that involve, for example, image mixing, multi-image mixing, or reinforcement learning data augmentations, to name a few (Kumar et al., 2023). These are not specified further since they are not part of the scope of this thesis.

## 2.4 Vision Transformer

Before foundation models are introduced, the transformer architecture, more precisely vision transformers (ViTs), are shortly introduced since they make big known foundation models possible.

Figure 8 shows the architecture of a vision transformer (Dosovitskiy et al., 2020).

Here, the input image, shown at the bottom left of the figure, is first divided into smaller, equally sized patches. Each of these patches is flattened into a 1D vector of pixel values. Position embeddings are added to each patch embedding to give the model an understanding of the position of each patch in the image. This ensures that the model can distinguish between patches in different locations. A special "class" embedding token is included, which will eventually carry the information needed to classify the entire image (Dosovitskiy et al., 2020).

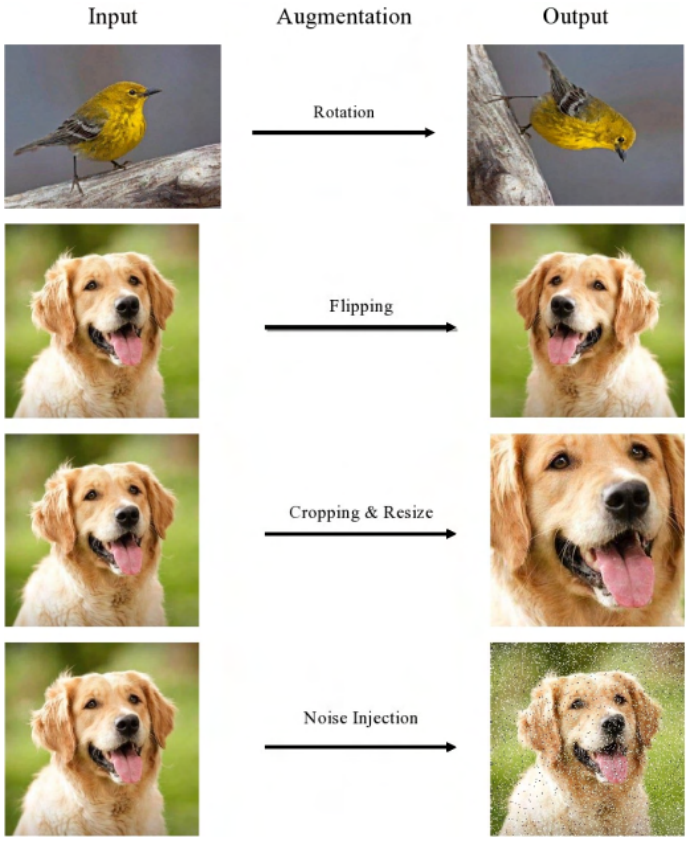


Figure 7: Examples of rotation, flipping, cropping & resize, and noise injection applied (Kumar et al., 2023).

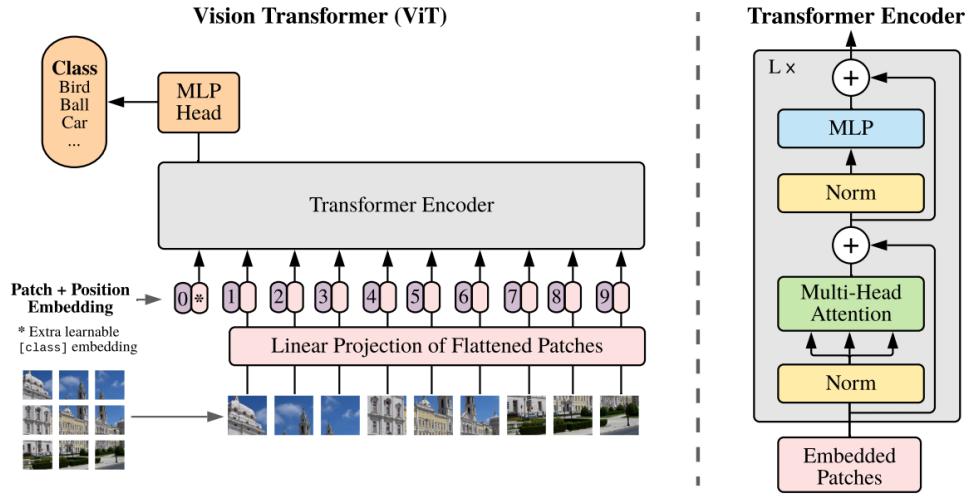


Figure 8: Architecture of a vision transformer (Dosovitskiy et al., 2020).

Each patch (with its position embedding) is then passed through a linear projection layer. This step transforms the flattened patch vectors into a uniform embedding

size suitable for processing in the transformer. These embedded patches are stacked to create a sequence.

The sequence of patch embeddings is fed into a transformer Encoder, which consists of multiple layers of:

- **Multi-Head Attention**: This component allows the model to understand relationships between patches by looking at how each patch "attends" to every other patch. It can learn which patches are significant relative to others.
- **Normalization Layers**: These layers help stabilize the training and ensure the outputs from each layer are standardized.
- **MLP (Multi-Layer Perceptron)**: After the attention mechanism, the output goes through an MLP to capture additional patterns and features.

This process is repeated for a set number of layers (represented by "L ×" on the right side of the figure 8), enabling the model to learn complex representations of the image (Dosovitskiy et al., 2020).

After passing through the transformer encoder, the "class" token contains information representing the entire image (Dosovitskiy et al., 2020). This token is then processed by an MLP Head (a simple neural network) to make the final prediction or classification, such as identifying whether the image shows a bird, ball, car, etc.

In summary, a vision transformer splits an image into patches, embeds them with positional information, and processes them using attention mechanisms in the transformer encoder to understand the relationships between patches. The "class" token accumulates this information and predicts the image's category. This approach leverages the powerful self-attention mechanism of transformers to capture both local and global patterns within an image (Dosovitskiy et al., 2020).

Dosovitskiy et al. (2020) also show that, unlike CNNs, ViTs are better at leveraging large datasets for high accuracy, which leads to foundation models. ViTs help lay the groundwork for visual foundation models by making transformers applicable to images. Still, foundation models, in general, aim to be universal and capable of handling multiple data types and tasks across domains.

## 2.5 Foundation Models

Bommasani et al. (2021) defines the term foundation model as "any model that is trained on broad data (generally using self-supervision at scale) that can be adapted (e.g., fine-tuned) to a wide range of downstream tasks". Examples of foundation models are BERT (Devlin et al., 2018) , CLIP (Radford et al., 2021a), and GPT-3 (Brown et al., 2020). So technically, the concept of foundation models is nothing new since they are based on deep neural networks and self-supervised learning, which were researched long before foundation models emerged. The vast scale and scope of foundation models developed have expanded massively. An example of this

is GPT-3, which has 175 billion parameters. So these models are enabled by the advancements in computational resources (Brown et al., 2020).

Emergence is an essential aspect of foundation models. It refers to the phenomenon where a system's behavior arises naturally rather than being explicitly designed. It is a source of scientific excitement and concern due to the potential for unexpected outcomes (Brown et al., 2020).

Homogenization is also a significant aspect of foundation models. Homogenization refers to the process of creating a unified model architecture or framework. This approach enables handling various tasks across different domains, eliminating the need for specialized models for each task. This approach generalizes learning across diverse datasets and tasks by training on vast amounts of heterogeneous data, producing a model that performs well in multiple areas. Figure 1 shows that a foundation model can learn from various data sources like images, text, speech, and other structured data sources and can be adapted to different tasks like object recognition, sentiment analysis, and image classification. It can be adapted not only to many different tasks but also to other domains such as healthcare, law, and education only to name a few (Brown et al., 2020), (Moor et al., 2023).

The healthcare domain is particularly interesting in the realm of this thesis. Although DL advances in many fields rapidly, it cannot keep up in this domain. That is primarily explained by the lack of appropriate datasets in this context (Stacke et al., 2021). Here comes transfer learning into play.

Transfer learning is a technique in ML where a model developed for one task can be adapted to solve a different but related task. So, instead of training a new model from scratch, transfer learning allows models to leverage knowledge gained from a previous task to improve performance on a new task. This is particularly useful when the designated task has limited data. Since the foundation models are pre-trained on broad data from different domains, they can be fine-tuned to a domain like healthcare with only small data through transfer learning. So this makes foundation models the perfect models to evaluate the underlying medical dataset of this thesis (Brown et al., 2020). In the following subsections, the four distinct models explored in this thesis are introduced: Dino, Dinov2, UNI, and Prov-GigaPath. Table 1 provides an overview of the evaluated model architectures, highlighting their number of parameters and the feature dimensions prior to the final classification layer. Parameter counts are expressed in millions (M) for clarity. Additionally, the used backbone for the ViTs is named (ViT-B/16, standing for the base backbone with a patch size of 16, L stands for Large, and g for giant).

### 2.5.1 Dino

Dino (Distillation with No Labels) is a foundation model used for self-supervised learning without labels. Figure 9 shows Dino´s architecture. The model uses an input picture. It is randomly transformed in two ways, resulting in two views: $x_1$ and $x_2$. These augmented views are fed into a **student** network and a **teacher** network.

| Model | Params (M) | # Output Dimension |
|---|---|---|
| Dino ViT-B/16 | 85.8 | 768 |
| Dinov2 ViT-B/14 | 86.6 | 768 |
| UNI ViT-L/16 | 303 | 1024 |
| Prov-GigaPath ViT-g/14 | 1100 | 1536 |

Table 1: Overview of the evaluated model architectures, highlighting their parameter counts, and feature dimensions before the final classification layer. Parameter counts are expressed in millions (M).

The student network, represented as $g_{\theta_s}$, processes one of the augmented views $x_1$ to produce a feature representation. This representation goes through a softmax layer, which gives the output probability distribution $p_1$ (Caron et al., 2021).

The teacher network, represented as $g_{\theta_t}$, processes the other augmented view $x_2$ of the image. It applies centering (to normalize outputs) and then passes the features through a softmax layer, producing $p_2$. The symbol "sg" (stop gradient) indicates that gradients are not backpropagated through the teacher network. This keeps the teacher network stable and helps avoid collapse (where both networks produce identical, non-informative outputs) (Caron et al., 2021).

The loss function is a cross-entropy loss between the probability distributions of the student and teacher networks: $-p_2 \log p_1$. The goal is to align the student's output distribution $p_1$ with the teacher's output distribution $p_2$, encouraging the student to mimic the teacher (Caron et al., 2021).

The teacher network's parameters $\theta_t$ are updated as an exponential moving average (EMA) of the student network's parameters $\theta_s$. This EMA update helps the teacher network evolve slowly based on the student, keeping it stable and progressively improving its representations (Caron et al., 2021).

Each network $g$ consists of a vision transformer backbone and an MLP as head in this thesis. However, they could be implemented with other architectures like a ResNet (He et al., 2015). The Dino model pre-trained on the ImageNet-1k dataset is used (Deng et al., 2009). ImageNet-1K is a widely used subset of the ImageNet database for image classification tasks. It consists of over 1.2 million labeled images divided into 1,000 classes, each representing a unique object category. The dataset was created as part of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), an annual competition that significantly advanced image recognition research. Each image in ImageNet-1K is labeled with one of the 1,000 object categories, which range from animals and everyday objects to scenes and abstract categories (Deng et al., 2009).

In summary, Dino uses a student-teacher framework, where the teacher helps guide the student's learning without labels. By augmenting the same image differently, it encourages the student to produce similar outputs for different views, improving the robustness of the network. The exponential moving average update mechanism allows the teacher to accumulate knowledge over time, creating a stronger supervisory

signal for the student. This approach enables Dino to learn meaningful representations in a self-supervised manner (Caron et al., 2021). This thesis utilizes the pre-trained student framework for further fine-tuning to the given task.

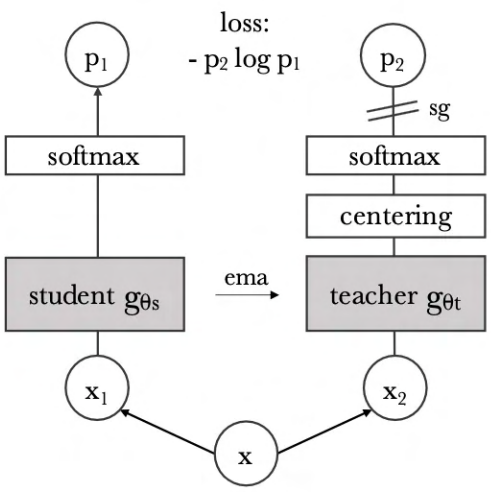


Figure 9: Simplified architecture of Dino. The teacher parameters get updated by an exponential moving average (ema), and the gradients are not backpropagated (sg=stop gradient) to keep the teacher framework stable (Caron et al., 2021).

.

### 2.5.2 Dinov2

As the name suggests, Dinov2 is the successor of Dino. Dinov2 also uses a self-supervised learning method for training vision transformers, but several architectural differences try to improve the performance, stability, and generalization (Oquab et al., 2024).

Dinov2 introduces more stable and effective training for diverse and complex data. Dinov2's loss functions improve upon Dino by focusing less on instance-level contrast and more on robust features (Oquab et al., 2024).

Dinov2 expands to support larger transformer backbones (Oquab et al., 2024). Dinov2's architecture is also optimized for efficiency, especially in handling large-scale datasets like ImageNet-21k (Deng et al., 2009) and LAION-400M (Schuhmann et al., 2021).

Dino uses fixed or learnable positional embeddings, while Dinov2 incorporates flexible positional embeddings, allowing the model to perform well across varied resolutions and input sizes without retraining (Oquab et al., 2024).

Dinov2 is trained on more diverse and larger datasets. It is optimized for multi-scale feature learning and supports larger, more diverse data, making it more robust for various downstream tasks. It is pre-trained with the LVD-142M dataset, or Large

Visual Dataset with 142 million images, specifically designed to train Dinov2 (Oquab et al., 2024). These pictures are selected for their diversity in visual content, context, and quality. Unlike web-scraped datasets like LAION-400m, LVD-142M was carefully curated to avoid noisy labels and low-quality images, which are common issues in web-scraped datasets (Oquab et al., 2024). This curation process ensures that images are diverse but also relevant and high-quality, which aids in more effective feature learning. Since the dataset is unlabeled, it allows Dinov2 to learn without supervised labels, making it more versatile for applications where annotated data is scarce. The diversity and scale of LVD-142M make it especially suitable for training a model that can generalize across various downstream tasks (Oquab et al., 2024).

### 2.5.3 UNI

UNI is a large foundation model pre-trained in histopathology images, which outperforms other encoders in different clinical tasks. It is pre-trained on the Mass-100K dataset using the Dinov2 self-supervised training algorithm (Chen et al., 2023). MASS-100K is a large-scale data set designed explicitly for self-supervised learning of medical image representations. This data set contains around 100,000 pathological images from different types of tissues and organs, offering diversity in visual patterns and textures. This diversity enables models to generalize well to other types of medical images. Although general-purpose self-supervised learning models like Dinov2 are trained on datasets such as ImageNet (Deng et al., 2009) and LAION-400m (Schuhmann et al., 2021), MASS-100K is tailored to medical data. This specialization enables models trained on MASS-100K to capture domain-specific features that would not be learned from general datasets. This makes it a promising candidate for this thesis (Chen et al., 2023).

Figure 10 gives an overview of the UNI model. An original patch (an input image) is transformed into multiple views 1 and 2. View 1 and 2 are two different random transformations of the original patch, creating varied perspectives of the same image content to improve robustness and generalization (Chen et al., 2023).

From each view, the image is divided into different types of crops:

- **Masked global crops**: These are larger patches where some regions are masked (occluded). This masking helps the model learning to predict missing parts and encourages contextual understanding.
- **Global crops**: Large, unmasked sections of the image.
- **Local crops**: Small, unmasked image sections, capturing finer details.

The cropped image patches are fed into the UNI model, which learns to extract features from each crop type. It processes each crop individually and aggregates the features to capture global and local representations (Chen et al., 2023).

The features extracted from the UNI model and the global crops are fed into a UNI teacher model. This teacher model generates high-quality, stable representations,

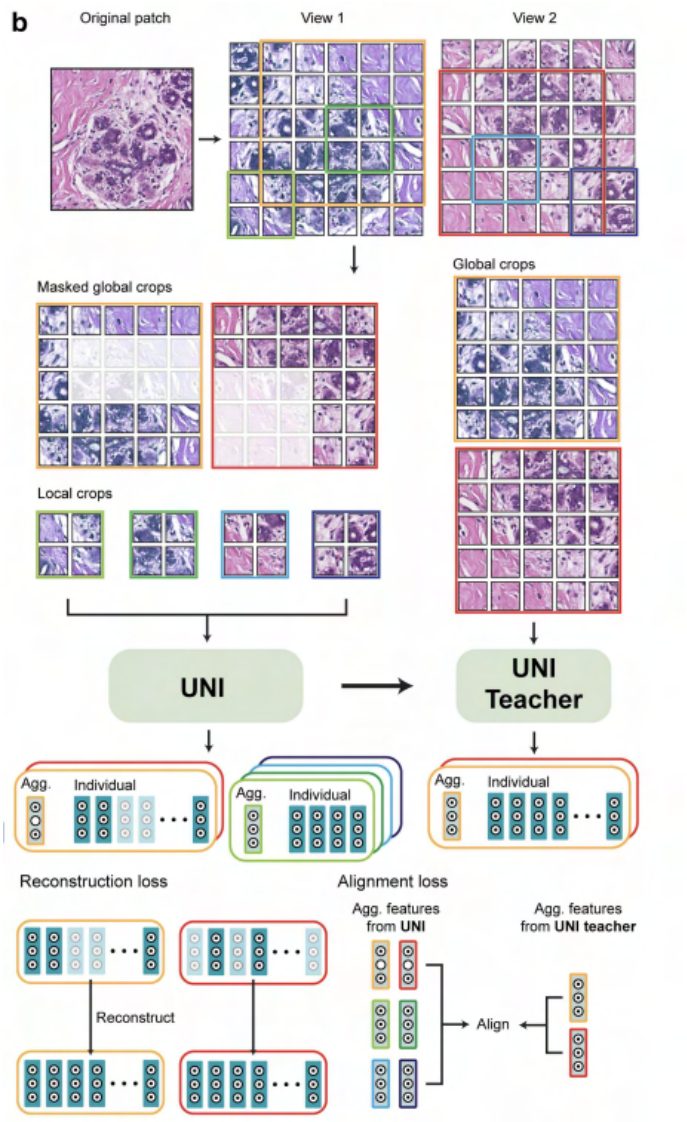


Figure 10: Architecture of UNI (Chen et al., 2023).

which guide the learning of the primary UNI model in a student-teacher fashion like Dino and Dinov2.

The model uses two different losses:

- **Reconstruction Loss**: This loss is applied to the UNI model to encourage it to reconstruct missing (masked) parts in the global crops based on the aggregated and individual features. The reconstruction task helps the model learn meaningful representations even without complete information.
- **Alignment Loss**: This aligns the aggregated features of the UNI model with those from the UNI Teacher model. The UNI model's features are adjusted to match the teacher's high-quality features, enhancing consistency and stability in feature extraction.

UNI achieves strong performance in downstream tasks without needing large labeled datasets. This is useful in fields like medical imaging, natural scene understanding, and any area where labeled data is limited and a reason it could perform well on the medical classification task of this thesis (Chen et al., 2023)

### 2.5.4 Prov-GigaPath (Prov)

The last foundation model used is Prov-GigaPath. A model pre-trained on pathology image slides. It is pre-trained on 171189 whole slides, divided into 1.3 billion image slides from Providence. Providence is a large health network in the United States. Prov-GigaPath achieves state-of-the-art performances on different digital pathology tasks, making it suitable as a model for this thesis (Xu et al., 2024).

Figure 11 shows the architecture of Prov-GigaPath. **a** shows that a histopathology slide is divided into a sequence of smaller image tiles, each of size 256×256 pixels. These tiles serve as input to a tile-level encoder, which consists of a vision transformer. The encoder processes each tile independently to generate image-level embeddings (Xu et al., 2024). The image-level embeddings from each tile go into a slide-level encoder. In this case, a LongNet model (Ding et al., 2023) is used to capture the long-range dependencies between tiles throughout the slide through dilated attention. The slide-level encoder generates slide-level embeddings for further processing (Xu et al., 2024). In **b**, the tile-level encoder is shown. It uses the Dinov2 pre-training algorithm with a student and a teacher framework as described in 2.5.2. The teacher model processes global crops of the tile without masks, while the student model processes local crops and masked versions of the global crop. A contrastive loss is applied to align the representations produced by the student model with those from the teacher model (Xu et al., 2024).

A LongNet-based decoder then processes the input embeddings with masks to reconstruct the missing or masked parts of the embeddings. The reconstruction process involves matching the target tile embeddings with the embeddings generated by the student model. Reconstruction loss is computed as the difference between the target tile embeddings and the reconstructed embeddings. This encourages the model to learn accurate representations of the tiles, even when data is missing (regions are masked), similar to the UNI framework in Section 2.5.3 (Xu et al., 2024).

The foundation models utilized in this thesis are the student models. How the models are utilized is shown in the methodology later on. All models were sourced from the "Pytorch Image Models (timm)" library (Wightman, 2019) at Huggingface [3].

## 2.6 Evaluation Metrics

Evaluation metrics play a crucial role in assessing the performance of classification models, providing quantitative insights into their predictive capabilities. Among the

[3]timm at Hugginface: https://huggingface.co/timm

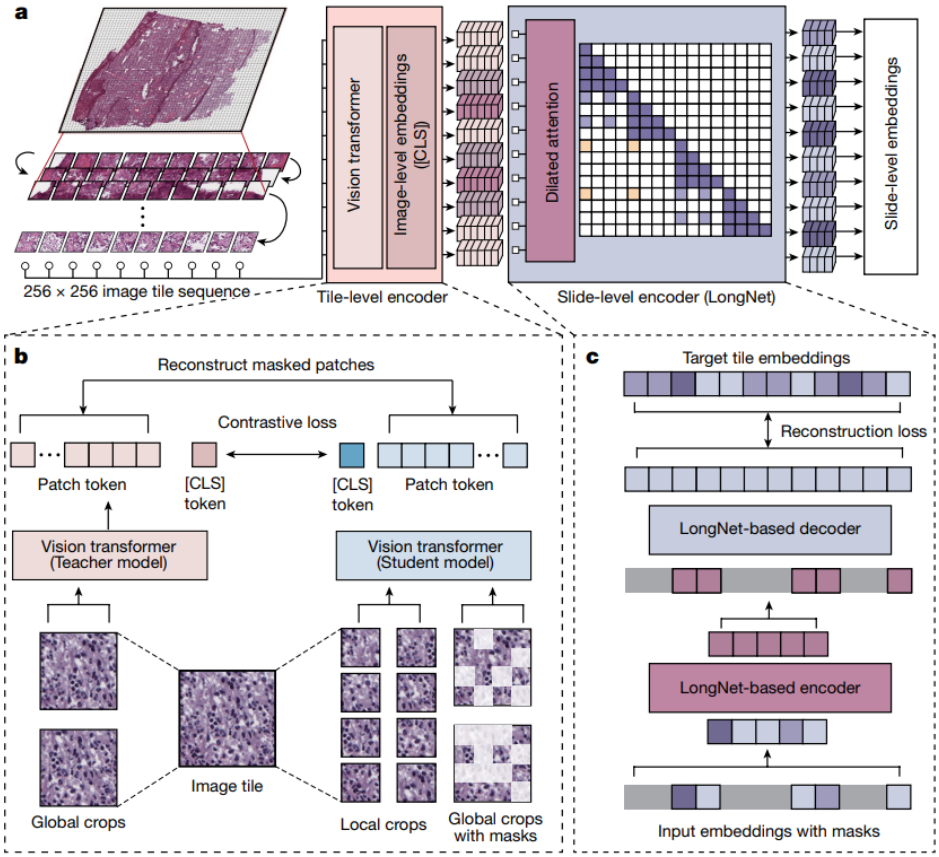


Figure 11: Flow chart of the architecture of Prov-Gigpath. **a**, Prov-GigaPath first transforms each input into 256x256 image tiles and converts each image tile into an embedding. **b**, Image tile-level pre-training using the Dinov2 learning algorithm. **c**, Slide-level pre-training with a LongNet (Xu et al., 2024).

various available metrics, accuracy, area under the curve (AUC), balanced accuracy, and Cohen's kappa are used due to their ability to capture different aspects of model performance. Each metric has unique strengths and limitations, making it essential to understand their characteristics when interpreting results or comparing models.

**Accuracy (Acc)**. The accuracy is the most intuitive metric, as it is defined as the ratio between correctly classified samples to the total number of samples shown in the following equation:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{5}$$

In the context of classification problems, TP, TN, FP, and FN refer to the classifier's outcomes compared to the actual labels.

**True Positive (TP)**: Cases where the model correctly predicts the positive class.

**True Negative (TN)**: Cases where the model correctly predicts the negative class.

**False Positive (FP)**: Cases where the model incorrectly predicts the positive class when the actual class is negative, also referenced as Type I error.

**False Negative (FN)**: Cases where the model incorrectly predicts the negative class when the actual class is positive, also referenced as Type II error.

In a multiclass classification problem, the accuracy is calculated in the same way, but by summating all classes, taking into account the true positives, false positives, true negatives, and false negatives for each class.

The accuracy provides a straightforward measure of a model's performance but may not be reliable in the presence of imbalanced datasets.

**Area Under the Curve (AUC)**. The Area under the Curve refers to the Area under the Receiver Operating Characteristic (ROC) Curve and quantifies a model's ability to distinguish between classes. The AUC measures the trade-off between True Positive Rate and False Positive Rate in different threshold values. The True Positive Rate measures the proportion of actual positive cases the model correctly identifies. It reflects the model's ability to detect positive instances. On the other hand, the False Positive Rate measures the proportion of actual negative cases that are incorrectly classified as positive by the model. The following two equations show True Positive Rate (TPR) and False Positive Rate (FPR):

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{7}$$

For multiclass classification tasks, the AUC is calculated for each class by treating it as the "positive" class and combining all other classes as the "negative" class. This process is repeated for each class, and the AUC scores for each class are then averaged.

The following values help us understand and interpret the AUC as a metric:

**AUC = 1.0**: Perfect classifier.

**AUC = 0.5**. Random guess.

**AUC < 0.5**. Worse than random guessing.

In summary, the AUC is a robust metric for class imbalance since it considers both sensitivity and specificity. It also provides a holistic view of a classifier's performance, making it a good addition to accuracy.

**Balanced Accuracy (Bal_Acc)**. Balanced accuracy is a modification of standard accuracy that accounts for class imbalance by averaging the true positive rate, as shown in this equation.

$$\text{Balanced Accuracy } = \frac{1}{C} \sum_{i=1}^{C} \frac{TP_i}{TP_i + FN_i} \tag{8}$$

Let $C$ be the total number of classes. $TP_i$ represents the True Positives for class $i$, i.e., the number of correctly predicted instances of class $i$. And $FN_i$ represents the False Negatives for class $i$, i.e., the number of instances of class $i$ that were incorrectly predicted as other classes.

As the name suggests, the balanced accuracy accounts for class imbalance, ensuring that the performance of minority classes is considered.

**Cohen's Kappa (Co)**. Cohen´s kappa $\kappa$ is a statistical measure of agreement extended to evaluate classification models. It quantifies the degree to which the model's predictions align with the ground truth beyond what is expected by random chance. Cohen´s kappa is calculated like this:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \tag{9}$$

Cohen´s kappa is the difference between the observed and expected agreement, normalized by the maximum possible agreement. The observed agreement $p_o$ is the proportion of instances where the classifier and the ground truth agree divided by the total number of samples, shown in equation 10. Equation 11 displays the Expected Agreement $p_e$. It is the proportion of agreement expected by chance, assuming a model and the ground truth are making their classifications independently.

$$p_o = \frac{\sum_{i=1}^{C} TP_i}{N} \tag{10}$$

$$p_e = \sum_{i=1}^{C} \left( \frac{TP_i + FP_i}{N} \times \frac{TP_i + FN_i}{N} \right) \tag{11}$$

$TP_i$ is the number of true positives for class $i$, $FP_i$ is the number of false positives for class $i$, $FN_i$ is the number of false negatives for class $i$, $C$ is the total number of classes, $N$ is the total number of instances (samples).

The following values help to understand and interpret the Cohen kappa as a metric:

$$\begin{aligned}
\kappa &\geq 0.81 : \text{Almost perfect agreement} \\
0.61 \leq \kappa &\leq 0.80 : \text{Substantial agreement} \\
0.41 \leq \kappa &\leq 0.60 : \text{Moderate agreement} \\
0.21 \leq \kappa &\leq 0.40 : \text{Fair agreement} \\
\kappa &< 0.20 : \text{Poor agreement} \\
\kappa &= 0 : \text{No agreement better than chance} \\
\kappa &< 0 : \text{Worse than random agreement}
\end{aligned}$$

In summary, Cohen´s kappa is also a suitable metric for imbalanced datasets and can provide a realistic assessment of model performance by accounting for the expected agreement. However, it is less intuitive than the other metrics introduced. Combining the four metrics gives a good overview of the model's overall performance.

# 3 Methodology

## 3.1 Benchmark

The benchmark evaluates model performance on the MedMNIST+ dataset using AUC, balanced accuracy, and Cohen's kappa metrics. All metrics are averaged across the 12 datasets to produce a single performance value. These same metrics are applied to assess robustness to distortions on the corrupted MedMNIST-C dataset. Additionally, this thesis includes accuracy as a supplementary metric for comparison of the results of this thesis. The technical concept and methodology for the benchmark and its associated website were collaboratively developed in a bachelor's thesis by Bachmeier (2024), where further details can be found regarding the website. This work extends the existing benchmark by introducing the capability to evaluate and compare results on the corrupted dataset version. The models introduced by Bachmeier (2024) and the models trained in this thesis work as a baseline.

The following sections outline the various training paradigms used to train the foundation models.

## 3.2 Training Methods

This topic is divided into two sub-fields. The first one (Training only the classification head) describes two training techniques that do not train the whole model but only classifiers to categorize the tasks with the features of the pre-trained backbones. The second describes two approaches that simultaneously train the entire network end-to-end on all 12 datasets. The used dataset has resolutions of 28×28, 64×64, 128×128 and 224×224. To ensure compatibility with the pre-trained models while retaining the unique characteristics of each resolution, all images were padded to 224×224 pixels using zero-padding, if necessary. That is needed since all models expect an input resolution of 224 x 224. Dinov2 is an exception since it requires images of 518×518. The images were padded to 518×518. This was done since zero-padding has minimal impact on classification accuracy while significantly decreasing training time compared to image resizing (Doerrich et al., 2024b). Zero-padding prevents neighboring zero-value input pixels from activating their corresponding convolutional units in the next layer. This minimizes the need for synaptic weight updates on outgoing connections, maintaining strong feature integrity during image reshaping (Doerrich et al., 2024b). The training was conducted on a single NVIDIA RTX™ A5000 GPU or NVIDIA L40 GPU, with the random seed set to 9930641, ensuring reproducibility.

### 3.2.1 Training only the Classification Heads

Xu et al. (2024) states the slide-level embeddings from Prov-Gigapath "can be used as features for diverse clinical applications." This is also true for the other foundation models introduced in this thesis. This work employs a training methodology where a **single** pre-trained backbone model is used to generate feature embeddings of the images. These embeddings are then fed into **12** different classification heads, each corresponding to one of the 12 tasks in the dataset. The backbone remains fixed, and only the heads are trained, allowing the model to train classifiers without retraining the whole model.

The classifiers used in this study include Support Vector Machines, LightGBM, Random Forest, k-Nearest Neighbors, and Linear Probing. For instance, when referring to the Support Vector Machine for the Dino backbone, it means that embeddings generated by the Dino model are passed through 12 different Support Vector Machines, each trained to solve one of the 12 tasks in the dataset, with training conducted across all four resolutions the dataset comes in (28×28, 64×64, 128×128, 224×224).

The five used Machine Learning classifiers are introduced in the following, starting with Support Vector Machines.

#### Support Vector Machines (SVMs)

Support Vector Machines are one of the most popular, if not the most popular, algorithms used to tackle classification. Figure 12 shows how the support vector

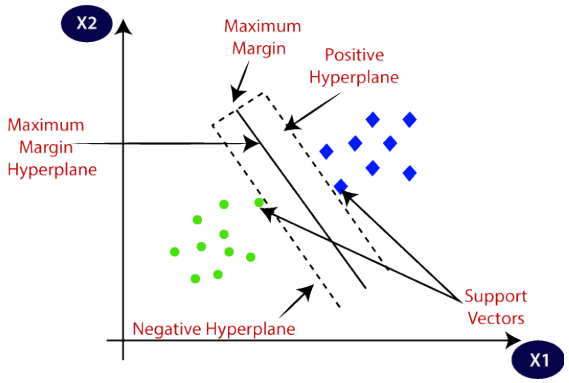


Figure 12: Explanation of support vector machines (Javatpoint, 2023).

machine for classification works. The green circles and blue squares represent two different classes. SVM aims to separate these two classes with a clear boundary.

In SVM, a hyperplane is a decision boundary that separates data points from different classes. In this 2D case, the hyperplane is a line. The SVM algorithm aims to find the best hyperplane that maximizes the margin between the two classes.

The support vectors are the data points closest to the hyperplane from both classes. These points are crucial in defining the margin and, hence, the position of the optimal hyperplane. The line in the center is the maximum margin hyperplane. This is the line that SVM selects as the decision boundary for separating the two classes with the biggest margin possible (Javatpoint, 2023), (Noble, 2006).

The dashed lines in the figure show the maximum margin, which is the distance between two hyperplanes closest to the data points from each class. The SVM algorithm maximizes this margin, ensuring the classes are as distinct as possible.

The figure shows a problem in a two-dimensional feature space. However, this also applies to the foundation models' multi-dimensional feature space. Further, the picture shows a perfect scenario where the two classes are linear and separable. However, this is most likely not the case in real-world scenarios, so the algorithm needs to address this. This is achieved by adding a "soft margin". This allows data points to cross the separating hyperplane (Noble, 2006).

Support Vector Machines have a regularization parameter **C** that controls the trade-off between maximizing the margin (generalization) and minimizing the classification error (training accuracy). A low **C** allows more misclassified points but seeks to maximize the margin between the support vectors and the decision boundary, thus prioritizing generalization and making the model less sensitive to noise but potentially underfitting. With a high **C**, the model tries to classify all training points correctly, even at the cost of having a smaller margin, which can lead to overfitting. The goal is to find the optimal value for **C** (Javatpoint, 2023).

The figure shows an SVM with a linear kernel, which means the problem is linearly separable, and the decision boundary is a straight line. Non-linear kernels exist and allow SVMs to create more complex non-linear decision boundaries by mapping data into a higher-dimensional space (Noble, 2006).

The following values for the **C**-parameter were tested: 1, 0.1, 0.01, and 0.001. The results showed that the best value was 0.01. Therefore, the SVM used in this thesis has a linear kernel and a **C** value of 0.01.

To choose a better **C**-parameter, various methods can be employed, such as grid search, random search, or the genetic algorithm (Liashchynskyi and Liashchynskyi, 2019). However, this study did not use these methods due to their time consumption. They could, nevertheless, be utilized to enhance the performance of the training algorithms further.

This also applies to the other classification heads introduced later in this chapter. A suitable hyperparameter combination was determined and used, but it could be further optimized using optimization algorithms when time is not a factor.

A non-linear kernel could potentially capture the non-linear relationships present in the data and might yield better results than a linear kernel. However, the linear kernel also provides comparable results while requiring significantly less training time, making it the more feasible choice within the thesis's time constraints.

### k-Nearest-Neighbors (kNN)

Figure 13 illustrates how the kNN algorithm works for classification. This classification is straightforward. In this example, the red squares and the green triangles represent two different classes of data points, and the yellow circle with a question mark represents a new point whose class is unknown. The goal is to classify this new point by looking at the "k" nearest data points in the feature space around it. Different distance metrics measure the distance between the data points, e.g., Euclidean distance and Manhattan distance.

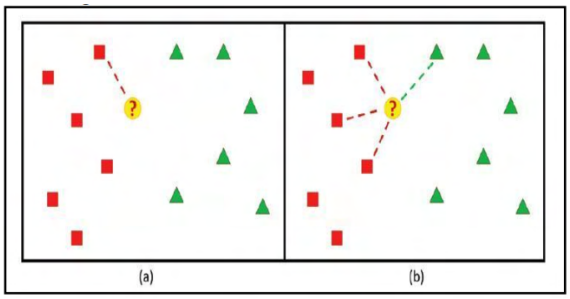


Figure 13: Illustration of the kNN algorithm. In this example, the red squares and green triangles represent two different classes of data points, and the yellow circle with a question mark represents a new point whose class is unknown (Imandoust and Bolandraftar, 2013).

Panel (a) shows the kNN-Integration for k=1 Neighbors. Here, only the single closest data point (1-Nearest Neighbor) is considered for classification. The yellow point connects to one red square, the nearest neighbor, suggesting the data point belongs to the red class.

Panel (b) shows the case for k=4. In this case, the algorithm considers the four nearest data points. The yellow point is connected to four data points: three red squares and one green triangle. With k=4, the majority class among the four nearest neighbors would determine the classification. In this case, three red squares indicate the yellow point is classified as the red class (Imandoust and Bolandraftar, 2013). This also works for non-binary classification problems, making it suitable as a classification head for the tasks of this thesis.

Doerrich et al. (2024a) explores combining kNN with large foundation models. For example, this can enhance image classification tasks by using kNN on top of foundation models. The system can adapt to new classes or domain shifts with minimal additional training, enhancing its adaptability and flexibility. The paper also states that kNN can quickly classify new or rare categories using only examples with few labels (few-shot classification). kNN is also easily interpretable, as predictions are based on specific nearby examples, making it easier to understand why a classification was made.

All these points make kNN a suitable method to try on the specific classification tasks of this paper. k is set to 11 for all for the kNN approach as suggested by Doerrich et al. (2024b).

### Random Forest (RF)

Figure 14 shows the classification of an image with a random forest. The random forest forms multiple decision trees while training. Each decision tree predicts a class for the input data based on the feature vector. And a majority vote predicts the final class.

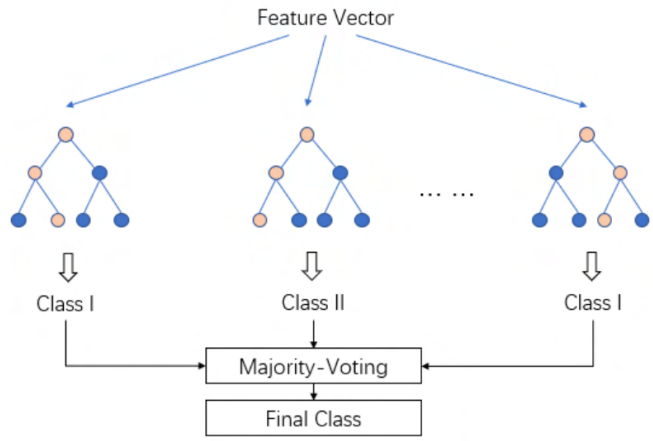


Figure 14: Illustration of the random forest algorithm (Liu et al., 2022b).

The following five steps from Azhari et al. (2019) explain how decision trees in the training stage are formed:

1. **Randomly select $k$ features from a total of $m$ features, where $k \ll m$.** At first, at each node in a decision tree, instead of considering all $m$ features, $k$ features are randomly selected as candidates for splitting. $k$ is smaller than $m$, this gives randomness and reduces overfitting.

2. **Among the $k$ features, calculate the node $d$ using the best split point**. For the selected $k$ features, the potential splits are evaluated based on a criterion such as Gini impurity. The feature and the split point that best improves the splitting criterion are chosen to define the node $d$.

3. **Split the node into daughter nodes using the best split**. The selected feature and its optimal split divide the dataset into two subsets (daughter nodes), each containing data points that match either side of the split. For example, given a feature "age" and a split point 30, the dataset is splitted into age smaller than 30 and greater than or equal to 30.

4. **Repeat steps 1 to 3 until $l$ number of nodes has been reached**.

5. **Build the forest by repeating steps 1 to 4 for $n$ trees**. This process is repeated recursively for each daughter node until a pre-defined depth($l$) is reached or other stopping conditions are met: E.g., Minimum number of samples in a node is reached, or no further improvement in the splitting criterion can be achieved.

This entire process (steps 1–4) is repeated $n$ times to build a forest of $n$ trees. Each tree is constructed using a random subset of the training data and a random subset of characteristics at each split.

As a result, you get a classification forest similar to LightGBM. The difference is in the way forests are formed.

The random forest as a classifier has high accuracy, provides insides of the significance of features and is resilient to missing values, making it versatile for real-world data and a suitable experiment for this thesis.

The following hyperparameters were used to train the different RF heads:

- **Max Depth (7)**: Limits tree depth to prevent overfitting.
- **Estimators (50)**: Number of trees; more trees improve performance but increase training time.
- **Min Samples Split (5)**: Minimum samples needed to split a node, avoiding small subgroups.
- **Min Samples Leaf (3)**: Minimum samples per leaf, reducing overfitting.

### Light Gradient Boosting Machine (LightGBM)

LightGBM is, as the name suggests, a variant of a Gradient Boosting Machine. It uses the technique of gradient boosting, which builds decision trees to improve a model's performance by sequentially reducing errors from previous trees. The term Light highlights the efficiency and speed of LightGBM. It is designed to be faster and more memory efficient. The original paper that introduced LightGBM first shows it (Ke et al., 2017): "LightGBM speeds up the training process of conventional Gradient Boosting Decision Trees by up to over 20 times while achieving almost the same accuracy."

Figure 14 shows the classification of an image with a random forest. LightGBM works after a similar concept. The input is a feature vector from the given image. In this case, the LightGBM classifier gets this feature vector from the underlying backbone, one of the foundation models.

LightGBM has multiple decision trees. Each decision tree predicts a value for the input data based on the feature vector. After all trees have made their predictions, LightGBM aggregates these predictions to determine the final class the classifier predicts and does not use a majority vote like RF (Ke et al., 2017).

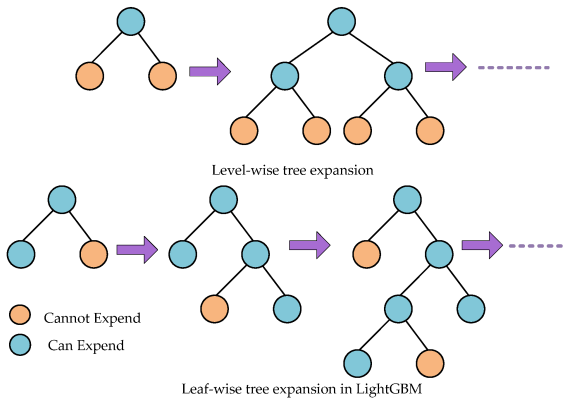


Figure 15: Forming of the decision trees in gradient boosting methods (Zhang et al., 2023b).

Figure 15 shows how decision trees are formed in classifier training. Traditional gradient boosting methods typically use level-wise growth, shown in the top part of the figure, where each level of the tree is fully expanded before moving to the next. LightGBM, on the other hand, uses leaf-wise growth. It grows the tree by splitting the leaf with the maximum potential reduction in loss. This can lead to deeper, more accurate trees with fewer splits but, at the same time, to potentially unbalanced trees. Setting different hyperparameters, like a tree's maximum depth, can prevent this.

To reduce the number of samples used in the tree construction, LightGBM also uses Gradient-based One-Side Sampling (GOSS). This technique selects a subset of data samples based on their gradient values. It keeps samples with large gradients since this indicates samples that are hard to predict and randomly samples from those with smaller gradients. By doing so, LightGBM reduces computational costs without losing much accuracy (Ke et al., 2017).

LightGBM also makes use of Exclusive Feature Bundling (EFB). LightGBM uses EFB to deal with high-dimensional data (feature vectors with many features). If two features are rarely non-zero at the same time, they are converted into a single feature, which reduces the number of features and the complexity. This helps speed up the training and reduce the memory usage (Ke et al., 2017).

LightGBM is an accurate and lightweight classifier for the given task. The LightGBM heads are trained with the following configuration:

- **Max Depth (5)**: Limits tree depth to control overfitting.
- **Num Leaves (31)**: Sets max leaves per tree; higher values increase capacity but risk overfitting.
- **Learning Rate (0.05)**: Adjusts weight updates; lower values improve precision but slow training.

- **Estimators (500)**: Number of trees; more trees enhance learning but increase training time.
- **Reg Alpha (0.1)**: L1 regularization encourages sparsity.
- **Reg Lambda (0.1)**: L2 regularization reduces overfitting.
- **Min Split Gain (0.1)**: Minimum gain for a split, avoiding insignificant splits.
- **Min Child Samples (20)**: Minimum samples per leaf to prevent overfitting.

These parameters balance complexity, regularization, and learning speed, helping to optimize the model's performance for a given dataset.

#### Linear Probing

Linear probing is a method in machine learning, especially within transfer learning, used to fine-tune a pre-trained model. It involves replacing the model's final classification layer(s) with a simple linear layer and training only this new layer while keeping the pre-trained model's weights fixed. This approach enables the model to apply its learned features to a new task without extensive retraining (Doerrich et al., 2024b). So linear probing, in the case of this thesis, is training 12 different linear layers as classifiers on the given embeddings of the different foundation models.

The following are the parameters and configurations used for training the 12 linear classifiers independently, adopted from (Doerrich et al., 2024b).

The training process is configured with specific parameters to enhance performance. A learning rate of 0.0001 is used to control the step size for parameter updates, and the batch size is set to 64. The AdamW optimizer is applied across all datasets. For loss functions, cross-entropy loss is used for binary and multiclass classification tasks, while BCEWithLogits loss is applied for the multi-label binary classification task. Sigmoid and softmax serve as activation functions, with sigmoid used for multi-label binary classification and softmax for binary and multiclass classification. Additionally, an early stopping criterion is implemented to halt training after five epochs without improvement in validation loss.

### 3.2.2 Training End-to-End

This section explains the two training methods used to train the networks in an end-to-end manner.

#### Multi-domain multi-task pre-training (mm-PT)

Pre-training on a dataset without medical images, such as ImageNet, may lack relevance for the medical domain. Instead, incrementally pre-training on a sequence of

available datasets could enable learning from multiple related datasets rather than relying on a single one. However, this approach may face challenges such as catastrophic forgetting of previous tasks. To address this issue, Woerner et al. (2024) proposes a multi-domain multi-task pre-training approach in which each model update involves sampling a batch from a randomly selected source task. This strategy aims to promote the learning of representations that are applicable across a broad range of tasks.

**Algorithm 1** Multi-domain Multi-task Pre-training

**Require:** Set $T$ of tasks, Datasets $D_t$ for tasks $t \in T$, Model $f$ with parameters $\theta$, Classification heads $l_t$ for $t \in T$

1: **while** training has not converged **do**
2: $t \leftarrow$ Sample a task $t$ from $T$ with probability $\frac{|D_t|}{\sum_{j \in T} |D_j|}$
3: $B \leftarrow$ Sample a batch from dataset $D_t$
4: $\theta \leftarrow \theta - \nabla_\theta \mathscr{L}_t(l_t(f(B;\theta)))$
5: **end while**
6: **return** $\theta$

Figure 16: Illustration of the multi-domain multi-task pre-training algorithm introduced in Woerner et al. (2024).

Figure 16 shows this multi-domain multi-task pre-training algorithm, in which the model is concurrently pre-trained on all datasets within their meta-dataset. In the case of this thesis, the dataset is replaced with the MedMNIST+ dataset (Yang et al., 2023). This indicates the model will be trained concurrently across various heterogeneous domains and tasks. The network is split into a backbone and a classification head to facilitate this. The backbone is formed by removing the last linear layer of the network, while a new linear layer is initialized as the head for each specific task. The algorithm involves randomly sampling batches from each task and training the model on these batches. Each batch is associated with the appropriate head and loss function (Woerner et al., 2024). For example, a dataset with two different subsets (sets A and B) and two batches each is given. This training technique takes a random batch from sets A and B. Since A and B are of the same size, the chance of both is 50%. For example, the algorithm chooses a batch from A. Now, the classification head (a linear layer) from A is put onto the backbone, and the model trains with this batch end-to-end with A's loss functions and activations. Now, one batch is left for A and 2 left for B. Now, the chances are 33.3% to train on the remaining batch of A and 66.6% to train on one of the two batches of B. If B is selected, the respective head is again changed to the classification head from B and the model is now trained with the loss function and activation from the classification head B. This is a simplified presentation of the workflow of this training approach. In the case of this thesis, there are 12 different datasets that differ in the number of batches, the activations for each task, and the loss of each task.

The hyperparameters for this approach are the same as those used for linear probing in section 3.2.1, with the key difference being that all 12 heads are now trained together with a single optimizer rather than with 12 independent optimizers. Additionally, a cosine annealing learning rate scheduler was used similar to Doerrich et al. (2024b).

### Multi-domain multi-task pre-training with data augmentation (mm-PT aug)

This subsection focuses on a training method were the model sees in each epoch the same number of samples for each class. That could boost the overall performance of the model. This thesis uses two data augmentation techniques to balance the datasets: Flipping and Rotating. These are two augmentation methods that do not corrupt the image. This is important since this work aims to compare the training techniques fairly. A model that trains on corrupted data would have an advantage in evaluating the robustness to distortions compared to one on a non-corrupted database. Another augmentation that would be suitable for this case would be cropping. It is not known where the important part is in each image - the part of the image that indicates the disease. It is possible to crop out the image's most important part for the classification task without knowing. This would make the image counterproductive for the model's training since the model trains on the wrong knowledge base. That is why cropping is not used. So, only flipping and rotation are used since they do not change the knowledge gained from a picture. Figure 17 shows the eight unique transformations possible by only using flipping

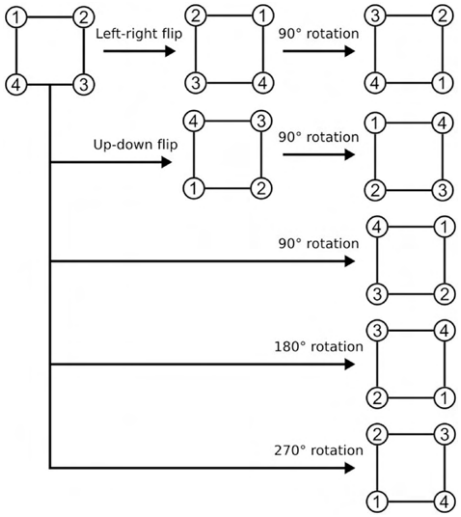


Figure 17: All possible augmentations with flipping and rotation (Sonogashira et al., 2020).

and rotation as augmentation techniques. One of these is the original image. The figure indicates how the image is augmented by referring to the corners as numbers 1 to 4.

The augmentations are only used on the training data. Since the smallest dataset has 546 training samples, we get a maximum of 4368 images for this dataset by using the augmentations. The second smallest dataset has 1080 samples, which means this dataset also has to be augmented. Every other dataset from the MedMNIST+ database has more training samples than 4368, which means no extension of the

data is needed for training. The complete MedMNIST+ dataset is introduced in the next chapter.

The training method remains the same as the multi-domain multi-task pre-training from section 3.2.2. But instead of training the model with all the data, the data is reduced to 69 batches per dataset. This approach ensures that the model processes the same number of images from each dataset in every epoch, preventing any dataset from being favored. With a batch size of 64 used in this thesis, the smallest dataset, containing 4,368 samples, results in approximately 69 batches (68.25) that are, at best, available for any dataset. For each epoch, the algorithm randomly selects batches from each task to train the specific head and its backbone, and 69 batches from each dataset are chosen randomly. The remaining samples are excluded from that epoch.

This training technique aims to improve the model's overall performance by ensuring no tasks are favored and that all tasks have an equally weighted impact on training and performance. For larger datasets, the model sees a diverse subset of data in each epoch by randomly sampling batches, which helps maintaining fairness and prevents over-reliance on any specific dataset.

Here is a simplified explanation using datasets A, B, and C:

With only 546 images, dataset A uses augmentation techniques like flipping and rotation to increase its total to 4,368 images, equivalent to 69 batches (64 images per batch). Larger datasets B and C contain more than 4,368 samples and do not require augmentation.

The model processes 69 randomly selected batches from each dataset in each epoch, ensuring equal representation. Extra samples from larger datasets B and C are excluded for that epoch, meaning the model only trains on a portion of their data while it sees dataset A fully. The validation loss is weighted to ensure that all datasets A, B, and C contribute equally to the evaluation, avoiding dominance by any single dataset.

This training technique aims to improve overall model performance by ensuring tasks from datasets A, B, and C are equally weighted.

The hyperparameters for this approach are identical to those used for linear probing in section 3.2.1. Additionally, a cosine annealing learning rate scheduler was used similar to (Doerrich et al., 2024b).

# 4 Used Dataset

This paper uses two datasets. The first one is the MedMNIST+ dataset, which includes 12 different 2D datasets of medical images. These datasets have a predefined split into Training, Validation and Test. It is used to train and evaluate the used models. The second one is the MedMNIST-C dataset. It applies different corruptions in 5 different severities on the Test Split of the MedMNIST+ dataset. This dataset is used to evaluate the robustness of the models in relation to distortion. In the following chapter the two datasets are shortly introduced.

## 4.1 MedMNIST+

MedMNIST v2 is a collection of 12 2D and 6 3D standardized biomedical image datasets, labeled and resized to resolutions of 28×28 and 28×28×28, respectively (Yang et al., 2023). The 3D images, however, are not relevant to this thesis. The MedMNIST+ collection builds on this by including higher resolutions of 64×64, 128×128, and 224×224 pixels. The dataset is diverse in size, with subsets ranging from a few hundred images to over 200,000, and features various data modalities, including Computed Tomography, Ultrasound, X-ray, and more. It supports four task types: binary classification, multi-class classification, multi-label classification, and ordinal regression (treated as multi-class classification for simplicity). The collection is standardized, with each subset pre-processed into a uniform format, and it includes colored images. Additionally, the dataset has a predefined split into training, validation, and test sets, ensuring consistency in model training across different experiments (Yang et al., 2023). These points make it a perfect dataset to evaluate the generalization potential of ML models.

The following figure provides an overview of the different datasets, which are subsequently described in detail.

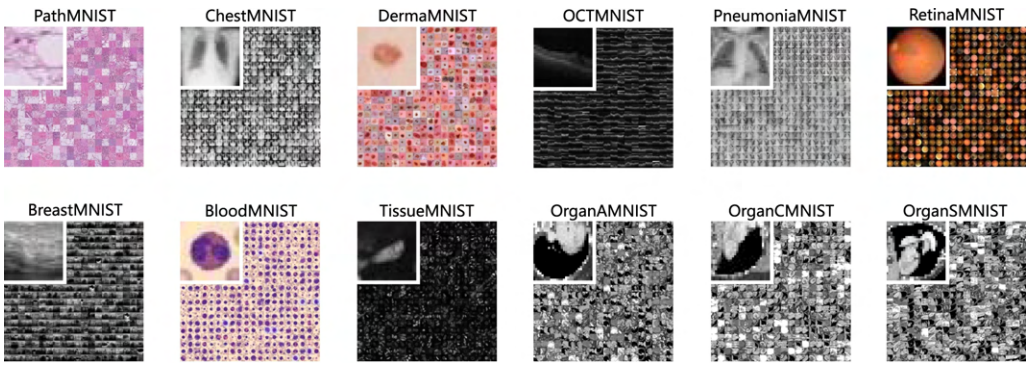


Figure 18: An overview over MedMNIST+ (Yang et al., 2023).

**PathMNIST**: The PathMNIST dataset contains 107,180 hematoxylin-eosin-stained tissue slide images (3×224×224). Tissue samples were collected from the National Center for Tumor Diseases (Heidelberg) and the University Medical Center

Mannheim, Germany. More than 100,000 image patches were generated by hand-delineating tissue regions in 86 colorectal cancer slides, supplemented by 7,180 images from 25 patients in the DACHS study. The dataset features 9 tissue classes: adipose tissue, background, debris, lymphocytes, mucus, smooth muscle, normal colon mucosa, cancer-associated stroma, and colorectal adenocarcinoma epithelium (Kather et al., 2019).

**ChestMNIST**: ChestMNIST (Wang et al., 2017), derived from the NIH ChestXRay dataset, contains 112,120 frontal-view chest X-ray images from 30,805 patients. It supports multi-label binary classification across 14 thoracic disease categories, including atelectasis, cardiomegaly, effusion, infiltration, mass, nodule, pneumonia, pneumothorax, consolidation, edema, emphysema, fibrosis, pleural thickening, and hernia. The original images have a resolution of 1024×1024 pixels and were resized to fit into the resolutions of MedMNIST+ (Wang et al., 2017), (Yang et al., 2023).

**DermaMNIST**: DermaMNIST comprises 10,015 dermatoscopic images from diverse sources, illustrating common pigmented skin lesions. It covers seven diseases: actinic keratoses, basal cell carcinoma, benign keratosis, dermatofibroma, melanocytic nevi, melanoma, and vascular skin lesions, designed for a multi-class classification task. Over 50% of diagnoses were pathology-confirmed, with others based on expert consensus or follow-up. Images collected over 20 years from Austria and Australia were digitized from various formats and cropped to focus on skin lesions (Tschandl et al., 2018).

**OctMNIST**: OctMNIST contains 109,312 optical coherence tomography images of the retina, sourced from retrospective cohorts of adult patients, aimed at studying age-related macular degeneration and diabetic macular edema. The dataset includes four diagnostic categories: choroidal neovascularization, diabetic macular edema, drusen, and normal, making it a multi-class classification task. The images, originally ranging in resolution from (384–1,536) × (277–512), were center-cropped for the usage in MedMNIST+ (Kermany et al., 2018), (Yang et al., 2023).

**PneumoniaMNIST**: PneumoniaMNIST focuses on pediatric pneumonia, containing 5,856 chest X-ray images from children. The dataset is a binary classification task, distinguishing between pneumonia and normal. The source images, with dimensions ranging from (384–2,916) × (127–2,713). The image got center-cropped to fit the resolutions of MedMNIST+ (Kermany et al., 2018), (Yang et al., 2023).

**RetinaMNIST**: The RetinaMNIST dataset focuses on diabetic retinopathy, diagnosed through retinal fundus images. It classifies diabetic retinopathy into five levels based on the International Clinical Diabetic Retinopathy scale: Grades 0 to 4, indicating increasing severity of the disease. The dataset contains 1,600 images, originally sized 3 × 1,736 × 1,824, which were center-cropped (Liu et al., 2022a), (Yang et al., 2023).

**BreastMNIST**: BreastMNIST comprises 780 grayscale ultrasound images (500 × 500 pixels) collected in 2018 from 600 women aged 25–75 at Baheya Hospital in Cairo, Egypt. Captured using LOGIQ E9 systems, the images were initially stored in DICOM format and later converted to PNG. The dataset simplifies the original three

classes (normal, benign, malignant) to distinguish between benign (or no tumor) and malignant breast tumors (Al-Dhabyani et al., 2020).

**BloodMNIST**: BloodMNIST consists of 17,092 microscopic peripheral blood cell images collected at the Core Laboratory of the Hospital Clinic of Barcelona using the CellaVision DM96 analyzer. The dataset includes eight classes: neutrophils, eosinophils, basophils, lymphocytes, monocytes, immature granulocytes, erythroblasts, and platelets. The blood samples were from individuals without infections, hematologic or oncologic diseases, or pharmacologic treatment, which could have contaminated the image (Acevedo et al., 2020).

**TissueMNIST**: TissueMNIST consists of 236,386 images of human kidney tissue, captured with an upright Leica SP8 Confocal Microscope. Grayscale images are categorized into eight types: epithelial cells of the proximal tubules, thick ascending limbs, distal convoluted tubules, and collecting duct, together with other cells such as leukocytes, podocytes, endothelial cells in glomeruli, and peritubular space cells (Ljosa et al., 2012). 2D maximum projections were generated by extracting the highest pixel value from each pixel along the axial axis to fit the dataset into MedMNIST+ (Yang et al., 2023).

**Organ(A,C,S)MNIST**: MedMNIST+ obtains the OrganMNIST datasets by utilizing 3D CT images from the Liver Tumor Segmentation Benchmark (Bilic et al., 2023). MedmNIST+ generated organ labels using bounding-box annotations for 11 body organs from a separate study. The Hounsfield units of these 3D images were converted to grayscale with an abdominal window, and 2D images were cropped from the central slices of the 3D bounding boxes in the axial, coronal, and sagittal views (Yang et al., 2023). This process resulted in the creation of the OrganAMNIST, OrganCMNIST, and OrganSMNIST datasets, where the letters A, C, and S represent the axial, coronal, and sagittal views, respectively. However the three different datasets differ in the total number of samples, as the following table shows. Table 2 summarizes the 12 datasets, including their data modality, task, number of samples, predefined split, and the number of samples in each split.

## 4.2 MedMNIST-C

Salvo et al. (2024) introduces MedMNIST-C, a corrupted version of the 12 2D datasets of the MedMNIST+ collection. The corruptions are specifically designed for each dataset to mimic the types of artifacts that may arise during image acquisition and processing, spanning five severity levels. This simulates real-world anomalies or potential shifts in the data distribution. The augmentations proposed by the paper are applied only to the test split of the MEDMNIST+ database, creating corrupted images that help to measure the robustness of distortions. The models are trained on the training data of the clean data set. The corrupted data set is used to evaluate the performance and robustness of models trained on clean data against possible real-world corruptions.

| Name | Data Modality | Task (# Classes / Labels) | # Samples | # Training / Validation / Test |
|---|---|---|---|---|
| PathMNIST | Colon Pathology | MC (9) | 107,180 | 89,996 / 10,004 / 7,180 |
| ChestMNIST | Chest X-Ray | ML (14) BC (2) | 112,120 | 78,468 / 11,219 / 22,433 |
| DermaMNIST | Dermatoscope | MC (7) | 10,015 | 7,007 / 1,003 / 2,005 |
| OCTMNIST | Retinal OCT | MC (4) | 109,309 | 97,477 / 10,832 / 1,000 |
| PneumoniaMNIST | Chest X-Ray | BC (2) | 5,856 | 4,708 / 624 / 624 |
| RetinaMNIST | Fundus Camera | OR (5) | 1,600 | 1,080 / 120 / 400 |
| BreastMNIST | Breast Ultrasound | BC (2) | 780 | 546 / 78 / 156 |
| BloodMNIST | Blood Cell Microscope | MC (8) | 17,092 | 11,959 / 1,712 / 3,421 |
| TissueMNIST | Kidney Cortex Microscope | MC (8) | 236,386 | 165,466 / 23,640 / 47,280 |
| OrganAMNIST | Abdominal CT | MC (11) | 58,850 | 34,581 / 6,491 / 17,778 |
| OrganCMNIST | Abdominal CT | MC (11) | 23,660 | 13,000 / 2,392 / 8,268 |
| OrganSMNIST | Abdominal CT | MC (11) | 25,221 | 13,940 / 2,452 / 8,829 |

Table 2: Details of the MedMNIST+ dataset, including the data modality, classification task type (with the number of classes), and data splits. (ML: Multi-Label, MC: Multi-Class, BC: Binary-Class, OR: Ordinary Regression) (Yang et al., 2023).

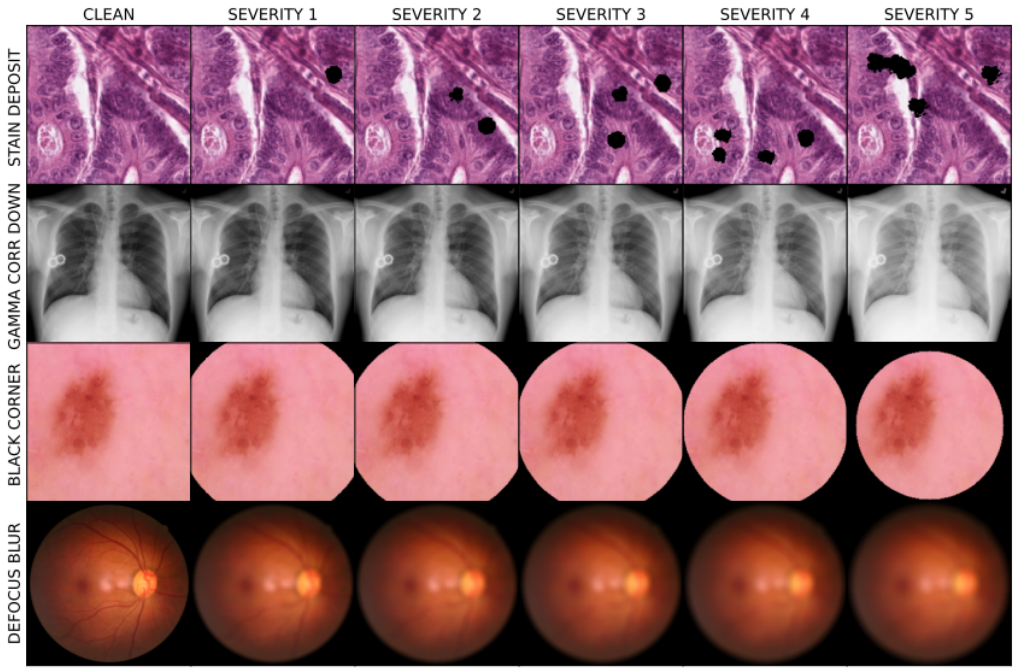


Figure 19: Four examples of MedMNIST-C. Four distinct corruptions applied to PathMNIST, ChestMNIST, DermaMNIST, and RetinaMNIST, listed from top to bottom. Severity level 1 indicates the lowest level of corruption, while severity level 5 signifies the highest level of distortion (Salvo et al., 2024).

.

Table 3 gives an overview of which corruption is applied to which dataset. The symbols [+/-] indicate whether the intensity of corruption is increased, decreased, or both, resulting in different forms of corruption. Figure 19 provides examples of four distinct corruptions applied to PathMNIST, ChestMNIST, DermaMNIST, and

| | Corruption categories | | | | |
|---|---|---|---|---|---|
| **Dataset** | **Digital** | **Noise** | **Blur** | **Color** | **Task-specific** |
| PathMNIST<br>BloodMNIST | JPEG<br>Pixelate | - | Defocus<br>Motion | Brightness[+/-]<br>Contrast[+/-]<br>Saturate | Stain deposit<br>Bubble |
| ChestMNIST<br>PneumoniaMNIST<br>OrganAMNIST<br>OrganCMNIST<br>OrganSMNIST | JPEG<br>Pixelate | Gaussian<br>Speckle<br>Impulse<br>Shot | Gaussian | Brightness[+/-]<br>Contrast[+/-] | Gamma corr.[+/-] |
| DermaMNIST | JPEG<br>Pixelate | Gaussian<br>Speckle<br>Impulse<br>Shot | Defocus<br>Motion<br>Zoom | Brightness[+/-]<br>Contrast[+/-] | Black corner<br>Characters |
| RetinaMNIST | JPEG<br>Pixelate | Gaussian<br>Speckle | Defocus<br>Motion | Brightness[-]<br>Contrast[-] | - |
| TissueMNIST | JPEG<br>Pixelate | Impulse | Gaussian | Brightness[+/-]<br>Contrast[+/-] | - |
| OCTMNIST | JPEG<br>Pixelate | Speckle | Defocus<br>Motion | Contrast[-] | - |
| BreastMNIST | JPEG<br>Pixelate | Speckle | Motion | Brightness[+/-]<br>Contrast[-] | - |

Table 3: This table summarizes the chosen types of image corruptions, each implemented at five progressively increasing severity levels. The symbols [+/-] indicate whether the intensity of corruption is increased, decreased, or varied in both directions, resulting in distinct forms of corruption (Salvo et al., 2024).

.

RetinaMNIST, listed from top to bottom. The picture also indicates that there are five different severity levels. Severity level 1 indicates the lowest corruption, while severity level 5 signifies the highest level of distortion (Salvo et al., 2024). All specified corruptions are applied to every possible resolution of the MedMNIST+ database at all five severity levels. The 28×28 resolution is a small exception. All the corruptions are used except for characters, bubble, and stain deposit, as the images are too small to apply. Characters are also not applied on images of size 64×64. Motion blur is only used for images with a size of 224×224. This is because the application programming interface introduced in Salvo et al. (2024) is primarily designed for images of size 224×224.

# 5 Experiments and Results

This section presents the results from the trained models. Table 4 shows the accuracy of all datasets if only the majority class is predicted.

| Dataset | Value |
|---|---|
| bloodmnist | 0.195 |
| breastmnist | 0.731 |
| chestmnist | 0.947 |
| dermamnist | 0.669 |
| octmnist | 0.250 |
| organamnist | 0.185 |
| organcmnist | 0.223 |
| organsmnist | 0.235 |
| patmnist | 0.186 |
| pneumoniamnist | 0.625 |
| retinamnist | 0.435 |
| tissuemnist | 0.321 |

Table 4: Accuracy values for each dataset of MedMNIST+ on the test split if only the majority class gets predicted.

The performance of the training methods and backbones this thesis implemented is now shown in greater detail, beginning with the results from the MedMNIST+ database.

## 5.1 Results for the MedMNIST+ Dataset

Table 5 shows the overall accuracies and AUCs for every training method with every backbone across the four image resolutions: 28×28, 64×64, 128×128, and 224×224. The overall value in the table is shown as the mean over all classes ± the standard deviation of the 12 different datasets. The best metric for every resolution for a training method is highlighted in bold. The best metric for a resolution for all training methods is highlighted with a background color. Dino SVM, for example, stands for Dino as the backbone and 12 different SVMs as the classification head for each of the 12 datasets and different resolutions independently. The accuracy is the mean accuracy of all 12 data modalities in the given resolution. The standard deviation is the standard deviation for all 12 datasets. The tables for all metrics for every dataset independently can be found in the Appendix A.2. The same goes for the plots shown in this section. Only a few plots are shown that give insight into the results. The Appendix A.1 contains every plot for every metric, every resolution, and every training technique.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | **79.12 ± 13.8** | **82.59 ± 13.4** | **84.21 ± 13.2** | **84.52 ± 12.3** | 91.13 ± 9.4 | 92.61 ± 9.1 | 93.69 ± 8.1 | 94.45 ± 7 |
| Dino LightGBM | 78.5 ± 13.55 | 82.3 ± 13.1 | 84.1 ± 11.9 | 84.18 ± 11.6 | **91.33 ± 9.6** | **93.18 ± 8.8** | **94.24 ± 7.4** | **94.75 ± 6.8** |
| Dino RF | 70.13 ± 15 | 73.5 ± 14.7 | 75.37 ± 14.7 | 75.89 ± 14.1 | 89.13 ± 10.1 | 88.09 ± 9.9 | 91.47 ± 8.6 | 92.1 ± 8 |
| Dino kNN | 73.61 ± 15.5 | 79.41 ± 14.4 | 81.17 ± 14.2 | 81.9 ± 12.9 | - | - | - | - |
| Dinov2 SVM | 78.55 ± 13.7 | 81.91 ± 12.7 | 83.03 ± 12.3 | 84.37 ± 11.1 | 90.73 ± 9.3 | 92.52 ± 8.5 | 93.67 ± 7.4 | 94.29 ± 6.7 |
| Dinov2 LightGBM | 76.75 ± 13.8 | 79.79 ± 12.6 | 81.57 ± 11.7 | 82.54 ± 10.9 | 89.81 ± 10.1 | 92.38 ± 8.5 | 93.65 ± 7.6 | 93.88 ± 7.1 |
| Dinov2 RF | 66.92 ± 14.5 | 69.78 ± 14.8 | 70.95 ± 15 | 72.61 ± 14 | 86.29 ± 9.7 | 88.41 ± 8.9 | 89.94 ± 8.2 | 90.32 ± 8.1 |
| Dinov2 kNN | 70.89 ± 14.5 | 74.7 ± 14.2 | 76.36 ± 14.1 | 78.13 ± 13.1 | - | - | - | - |
| UNI SVM | 78.93 ± 13.7 | 81.63 ± 12.9 | 83.3 ± 11.6 | 83.05 ± 12.6 | 90.92 ± 9.4 | 92.6 ± 8.8 | 93.69 ± 7.7 | 94.03 ± 7.2 |
| UNI LighGBM | 77.26 ± 13.9 | 80.62 ± 12.5 | 81.55 ± 11.8 | 82.55 ± 11.4 | 90.67 ± 10.0 | 92.68 ± 8.8 | 93.33 ± 8.0 | 93.85 ± 7.4 |
| Uni RF | 69.51 ± 15.2 | 72.75 ± 15.4 | 74.13 ± 15.4 | 74.21 ± 15.7 | 86.99 ± 10.3 | 89.3 ± 9.6 | 90.71 ± 8.8 | 91.06 ± 8.5 |
| Uni kNN | 71.39 ± 16 | 75.85 ± 25.5 | 77.55 ± 14.9 | 78.22 ± 14.5 | - | - | - | - |
| Prov SVM | 78.87 ± 13.7 | 82.78 ± 12.1 | 83.7 ± 12.1 | 83.93 ± 11.2 | 90.8 ± 9.5 | 92.96 ± 8.7 | 93.94 ± 7.5 | 94.18 ± 7.0 |
| Prov LightGBM | 77.42 ± 14.3 | 81.45 ± 12.5 | 81.92 ± 12.1 | 83.03 ± 11.3 | 90.44 ± 10.2 | 93.01 ± 8.7 | 94.07 ± 7.2 | 94.34 ± 6.9 |
| Prov RF | 69.56 ± 15.4 | 72.67 ± 15.7 | 74.25 ± 15.5 | 75.35 ± 15.2 | 87.21 ± 10.2 | 89.42 ± 9.4 | 90.88 ± 8.4 | 91.01 ± 8.3 |
| Prov kNN | 71.88 ± 16.2 | 76.69 ± 15.2 | 77.74 ± 14.7 | 78.27 ± 13.8 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | **78.49 ± 13.3** | **82.74 ± 12.4** | **84.51 ± 12.2** | **84.89 ± 12.1** | **91.07 ± 9.6** | **93.38 ± 8.3** | **94.59 ± 6.9** | **95.07 ± 6.4** |
| Dinov2 | 77.49 ± 13.3 | 81.26 ± 12.1 | 83.06 ± 11.9 | 83.65 ± 11 | 90.25 ± 9.6 | 92.71 ± 8.2 | 93.98 ± 7.3 | 94.44 ± 6.7 |
| UNI | 77.49 ± 13.6 | 81.29 ± 12.4 | 82.25 ± 11.5 | 82.65 ± 11.9 | 90.12 ± 10 | 92.69 ± 8.3 | 93.76 ± 7.3 | 93.97 ± 6.8 |
| Prov | 77.71 ± 13.4 | 82.11 ± 12 | 83.32 ± 11.2 | 83.75 ± 11.2 | 90.33 ± 9.9 | 93.17 ± 8.3 | 94.34 ± 6.9 | 94.37 ± 6.7 |
| **mm-PT** | | | | | | | | |
| Dino | 74.51 ± 12.6 | 81.36 ± 13 | 82.43 ± 13.1 | 83.7 ± 13.3 | 88.82 ± 10.4 | 91.2 ± 10.3 | 92.08 ± 9.9 | 92.8 ± 9.6 |
| Dinov2 | 75.17 ± 12.2 | 77.94 ± 13 | 80.28 ± 13.1 | 81.02 ± 12.8 | 89.65 ± 9.4 | 90.15 ± 10.4 | 91.19 ± 10.8 | 91.08 ± 10.6 |
| UNI | 76.1 ± 12.8 | 80.2 ± 13.7 | 82.6 ± 13.5 | 81.97 ± 13.3 | 88.63 ± 10.6 | 91.35 ± 10.2 | 92.29 ± 9.7 | 92.03 ± 10.3 |
| Prov | **78.08 ± 13.5** | **82.87 ± 12.8** | **84.88 ± 12.6** | **85.27 ± 13** | **90.0 ± 10.4** | **92.32 ± 9.5** | **92.98 ± 9.3** | **93.48 ± 9.1** |
| **mm-PT** | | | | | | | | |
| Dino | 73.75 ± 14.4 | 76.55 ± 14.6 | 76.84 ± 16 | 76.93 ± 15.6 | 88.06 ± 10.7 | 88.88 ± 12 | 89.77 ± 12.5 | 89.3 ± 12.5 |
| Dinov2 | 70.4 ± 15.2 | 73.17 ± 14.8 | 73.23 ± 15.9 | 74.7 ± 15.7 | 85.25 ± 12.3 | 87.33 ± 12.3 | 86.87 ± 12.6 | 87.94 ± 12.4 |
| UNI | 73.51 ± 15.4 | 75.89 ± 14.5 | 77.85 ± 14.7 | 77.81 ± 14.5 | 87.77 ± 11.7 | 88.8 ± 12 | 89.74 ± 12.5 | 89.53 ± 12.8 |
| Prov | **74.99 ± 15.5** | **77.05 ± 14.7** | **79.94 ± 14.6** | **81.99 ± 13.7** | **88.45 ± 11.3** | **89.26 ± 12** | **90.54 ± 12** | **92.2 ± 11** |

Table 5: Summary of benchmark results presenting the average mean and standard deviation for accuracy and the area under the receiver operating characteristic curve (AUC) across all datasets, covering all combinations of training schemes, models, and image resolutions. Since the k-NN algorithm lacks a training phase, it is not influenced by the stochasticity inherent in model training and reports only the overall accuracy. The best result for each resolution across all training schemes and models is highlighted with a background color, while the best result for each training scheme and resolution is highlighted in bold.

Linear probing emerges as the top-performing method, followed by Support Vector Machines (SVM) and LightGBM, with Dino as the best backbone overall. Table 5 highlights the following insights about model performance:

**Impact of Image Resolution**: Performance steadily improves as image resolution increases from 28×28 to 128×128 for accuracy and AUC. However, beyond 128×128, this trend plateaus, with little to no additional gains observed at 224×224.

**Classifier Performance**: SVM and LightGBM outperform kNN and random forest across most configurations. Among all training schemes, linear probing achieves the best results overall, followed by SVM and LightGBM.

**Backbone Performance:** Dino is the best-performing pre-trained backbone, while Prov is the best backbone for the mm-PT and mm-PT aug techniques.

**mm-PT Approaches**: Prov stands out as the top performer for mm-PT training schemes, achieving the highest accuracy and AUC. While most models experience performance drops when transitioning from linear probing to mm-PT, Prov is an exception, showing slight improvement in accuracy but a decline in AUC. The Prov backbone with mm-PT is the best model for accuracy in three resolutions. However, the mm-PT augmentation (mm-PT aug) approach performs significantly weaker than all other methods.

Table 6 shows similar tendencies for the balanced accuracy and Cohen´s kappa metrics:

**Impact of Image Resolution**: Performance improves steadily as image resolution increases from 28×28 to 128×128 for balanced accuracy and Cohen´s kappa. However, beyond 128×128, this trend plateaus, with minimal to no additional gains observed at 224×224.

**Classifier Performance**: SVM and LightGBM consistently outperform kNN and random forest across most configurations. Linear probing achieves the best overall results among all training schemes, closely followed by SVM and then LightGBM.

**Backbone Performance:** Dino emerges as the best-performing pre-trained backbone overall, while Prov excels as the top backbone for mm-PT and mm-PT augmentation (mm-PT aug) techniques.

**mm-PT Approaches**: Prov stands out as the top performer within mm-PT training schemes, achieving the highest balanced accuracy and Cohen's kappa. While most models show performance declines when transitioning from linear probing to mm-PT, Prov is an exception, demonstrating partial improvements in balanced accuracy and Cohen's kappa. However, the mm-PT augmentation (mm-PT aug) approach underperforms significantly compared to other methods.

The following section gives more profound insights into the different training techniques and the performance of the different datasets. Since SVM and LightGBM have proven overall that they perform much better than Random Forest and kNN, the focus is on SVM and LightGBM. For every training method, one plot for one resolution and one metric for all backbones are shown. The plots for every other

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 67.85 ± 19.7 | 72.66 ± 19.6 | **75.25 ± 18.8** | **75.62 ± 17.9** | **60.41 ± 26.5** | **65.35 ± 27.4** | **68.72 ± 26.9** | **69.48 ± 25.5** |
| Dino LightGBM | 67.14 ± 18.6 | 71.31 ± 19.8 | 73.95 ± 18.4 | 74.13 ± 17.6 | 60.06 ± 25.6 | 65.03 ± 26.9 | 68.45 ± 25.9 | 68.86 ± 25.1 |
| Dino RF | 55.12 ± 18.4 | 58.04 ± 20.6 | 60.48 ± 21.6 | 61.36 ± 21.8 | 46.83 ± 23.5 | 50.29 ± 26.0 | 53.36 ± 27.0 | 54.47 ± 27.1 |
| Dino kNN | 61.37 ± 19.8 | 68.09 ± 20.9 | 70.29 ± 21.4 | 71.3 ± 20 | 51.93 ± 26.9 | 60.99 ± 27.2 | 63.59 ± 27.8 | 65.02 ± 26.2 |
| Dinov2 SVM | 67.48 ± 18.7 | 71.49 ± 18.9 | 72.94 ± 18.3 | 74.66 ± 17.1 | 59.39 ± 26.3 | 64.63 ± 26.3 | 66.07 ± 26.5 | 69.09 ± 24.7 |
| Dinov2 LightGBM | 65.39 ± 17.9 | 68.4 ± 18.4 | 70.84 ± 17.5 | 71.89 ± 16.8 | 57.45 ± 25.2 | 61.31 ± 25.8 | 64.46 ± 24.9 | 66.08 ± 24.3 |
| Dinov2 RF | 50.99 ± 17.0 | 54.03 ± 18.7 | 55.64 ± 20.2 | 57.68 ± 19.6 | 41.74 ± 21.5 | 45.72 ± 23.3 | 47.14 ± 24.8 | 49.29 ± 24.6 |
| Dinov2 kNN | 58.03 ± 17.8 | 61.98 ± 19.5 | 64.11 ± 20.4 | 66.33 ± 19.7 | 48.38 ± 24.9 | 53.24 ± 26.3 | 55.46 ± 27.2 | 58.92 ± 25.7 |
| UNI SVM | **67.94 ± 18.7** | 71.43 ± 18.5 | 73.99 ± 17.1 | 73.96 ± 17.3 | 60.21 ± 26.0 | 63.67 ± 26.7 | 67.12 ± 25.2 | 67.53 ± 25.0 |
| UNI LightGBM | 66.12 ± 18.2 | 69.57 ± 18.0 | 70.75 ± 17.3 | 72.04 ± 17.1 | 58.41 ± 25.3 | 62.58 ± 25.7 | 64.21 ± 25.1 | 66.04 ± 24.8 |
| Uni RF | 54.49 ± 18.6 | 57.69 ± 20.8 | 59.13 ± 21.7 | 59.37 ± 22.7 | 45.72 ± 23.5 | 48.98 ± 26.8 | 51.0 ± 27.4 | 50.95 ± 28.2 |
| Uni kNN | 58.79 ± 19.4 | 64.21 ± 19.9 | 66.12 ± 20.5 | 66.55 ± 20.6 | 49.7 ± 25.2 | 55.91 ± 26.9 | 59.02 ± 26.6 | 59.88 ± 26.5 |
| Prov SVM | 67.94 ± 18.9 | **72.78 ± 18.1** | 74.33 ± 17.8 | 75.01 ± 16.8 | 59.64 ± 26.6 | 65.48 ± 26.3 | 67.89 ± 25.6 | 68.64 ± 24.6 |
| Prov LightGBM | 66.61 ± 18.6 | 70.69 ± 18.1 | 71.29 ± 17.7 | 72.91 ± 17.1 | 58.39 ± 25.9 | 64.22 ± 25.6 | 65.04 ± 25.2 | 67.16 ± 24.5 |
| Prov RF | 54.56 ± 19.0 | 57.54 ± 21.0 | 59.4 ± 21.9 | 60.97 ± 22.0 | 45.32 ± 24.0 | 48.65 ± 27.5 | 51.34 ± 27.4 | 53.75 ± 27.1 |
| Prov kNN | 59.31 ± 19.8 | 64.45 ± 20.6 | 65.76 ± 21.2 | 66.69 ± 20 | 49.71 ± 26.2 | 57.0 ± 26.9 | 58.32 ± 27.3 | 60.02 ± 25.6 |
| **Linear Probing** | | | | | | | | |
| Dino | **66.8 ± 19.2** | **73.27 ± 18.2** | **75.23 ± 18** | **76.27 ± 17.6** | **59.45 ± 25.9** | **65.91 ± 26.5** | **69.04 ± 26.1** | **70.02 ± 25.6** |
| Dinov2 | 66.66 ± 18.6 | 70.53 ± 18.3 | 73.42 ± 17.7 | 74.13 ± 16.2 | 57.59 ± 25.8 | 63.5 ± 26 | 66.7 ± 25.9 | 67.74 ± 24.8 |
| UNI | 66.1 ± 18.2 | 71.33 ± 17.6 | 72.74 ± 16.6 | 73.13 ± 16.7 | 57.84 ± 25.7 | 63.25 ± 26.4 | 65.4 ± 25 | 66.59 ± 25 |
| Prov | 66.55 ± 18.5 | 72.06 ± 17.7 | 74.05 ± 16.8 | 74.58 ± 16.9 | 57.76 ± 26.2 | 64.56 ± 26 | 67.18 ± 24.9 | 68.23 ± 24.7 |
| **mm-PT** | | | | | | | | |
| Dino | 61.5 ± 18.5 | 71.25 ± 19.6 | 72.2 ± 19.5 | 74.45 ± 19.4 | 51.54 ± 26.1 | 63.17 ± 27.8 | 63.93 ± 28.5 | 67.37 ± 27.8 |
| Dinov2 | 65.0 ± 71.3 | 66.68 ± 18.7 | 70.37 ± 19.3 | 70.45 ± 19.6 | 56.55 ± 23.4 | 58.09 ± 26 | 61.89 ± 27.2 | 61.65 ± 28.9 |
| UNI | 63.91 ± 17.7 | 69.11 ± 20.7 | 71.92 ± 20.9 | 70.34 ± 21.2 | 55.06 ± 25.6 | 60.29 ± 29.5 | 63.42 ± 30.6 | 61.38 ± 31.9 |
| Prov | **65.76 ± 20.8** | **72.34 ± 19.4** | **75.13 ± 19.3** | **75.18 ± 20.6** | **57.8 ± 27.1** | **64.12 ± 29.2** | **67.69 ± 28.4** | **67.24 ± 30.6** |
| **mm-PT aug** | | | | | | | | |
| Dino | 62.66 ± 18.9 | 65.01 ± 19.6 | 65.42 ± 21.3 | 65.52 ± 21.3 | 53.38 ± 0.25 | 56.79 ± 0.26 | **57.15 ± 0.28** | 56.63 ± 0.28 |
| Dinov2 | 57.27 ± 18.7 | 61.91 ± 19.6 | 61.31 ± 19.8 | 62.99 ± 20.2 | 48.59 ± 23.3 | 52.11 ± 25.8 | 52.16 ± 26 | 54.86 ± 26.2 |
| UNI | 62.32 ± 19.2 | 65.02 ± 19.6 | 67.03 ± 19.6 | 66.12 ± 20.1 | 53.96 ± 24.8 | 56.33 ± 25.6 | 59.05 ± 26.6 | 58.68 ± 26.6 |
| Prov | **62.84 ± 20.7** | **65.17 ± 20.5** | **69.85 ± 21.2** | **73.27 ± 19** | **54.5 ± 26.6** | **57.03 ± 26.7** | **62.53 ± 27.2** | **65.24 ± 27.2** |

Table 6: Summary of benchmark results presenting the average mean and standard deviation for balanced accuracy and Cohen´s kappa across all datasets, covering all combinations of training schemes, models, and image resolutions. Furthermore, since k-NN directly uses embeddings and labels for classification, it does not provide a reliable AUC score. The best result for each resolution across all training schemes and models is highlighted with a background color, while the best result for each training scheme and resolution is highlighted in bold.

resolution and metric and the missing plots for not-shown classifiers can be found in Appendix A.2.

### 5.1.1 Training without the Backbone

#### Support Vector Machines (SVMs)

This section shows the results of the SVM as a classifier in more detail. Figure 20 shows the accuracies for Dino, Dinov2, Uni, and Prov as backbone with an SVM-classifier.

The x-axis represents the different datasets, while the y-axis corresponds to their respective accuracy values. The color coding provides additional information: green denotes the accuracy of the training data, orange represents the validation data, and blue corresponds to the test data. Since test data accuracy is of primary interest, the blue points are annotated with their actual values, rounded to two decimal places.

Each subplot's title specifies the backbone model used and the average test accuracy across 12 classifications. For example, the label TestAcc_Dino indicates using Dino as the backbone, with a mean test accuracy of 84.21% and a standard deviation of 13.2%. Dino exhibits the highest test accuracy among the backbones, while other configurations demonstrate slightly lower performance.

Accuracy values vary substantially between tasks, with some datasets showing significant discrepancies between training, validation, and test accuracies. Across most tasks, the four backbones yield comparable results, except for breastMNIST and retinaMNIST, where Dino outperforms the others by at least three percentage points.

Additionally, the figure highlights two key trends: training accuracies (green line) tend to be consistently higher, whereas test accuracies (blue line) often exhibit more significant variability.

#### LightGBM

This section shows the results of LightGBM as a classifier in more detail. Figure 21 shows the accuracies for Dino, Dinov2, Uni, and Prov as backbone with a LightGBM-classifier.

The highest test accuracy is achieved using the Dino backbone, with a mean accuracy of 84.14% and a standard deviation of 11.9%. Other configurations exhibit lower performance, reducing at least two percentage points, similar to the performance observed with the SVM classifier. Accuracy values vary considerably across tasks, with notable gaps between training, validation, and test accuracies. These gaps are more pronounced compared to those observed when using the SVM classifier.

For most datasets, the accuracies across tasks are comparable among the four backbones, with Dino consistently demonstrating the best performance in most cases.

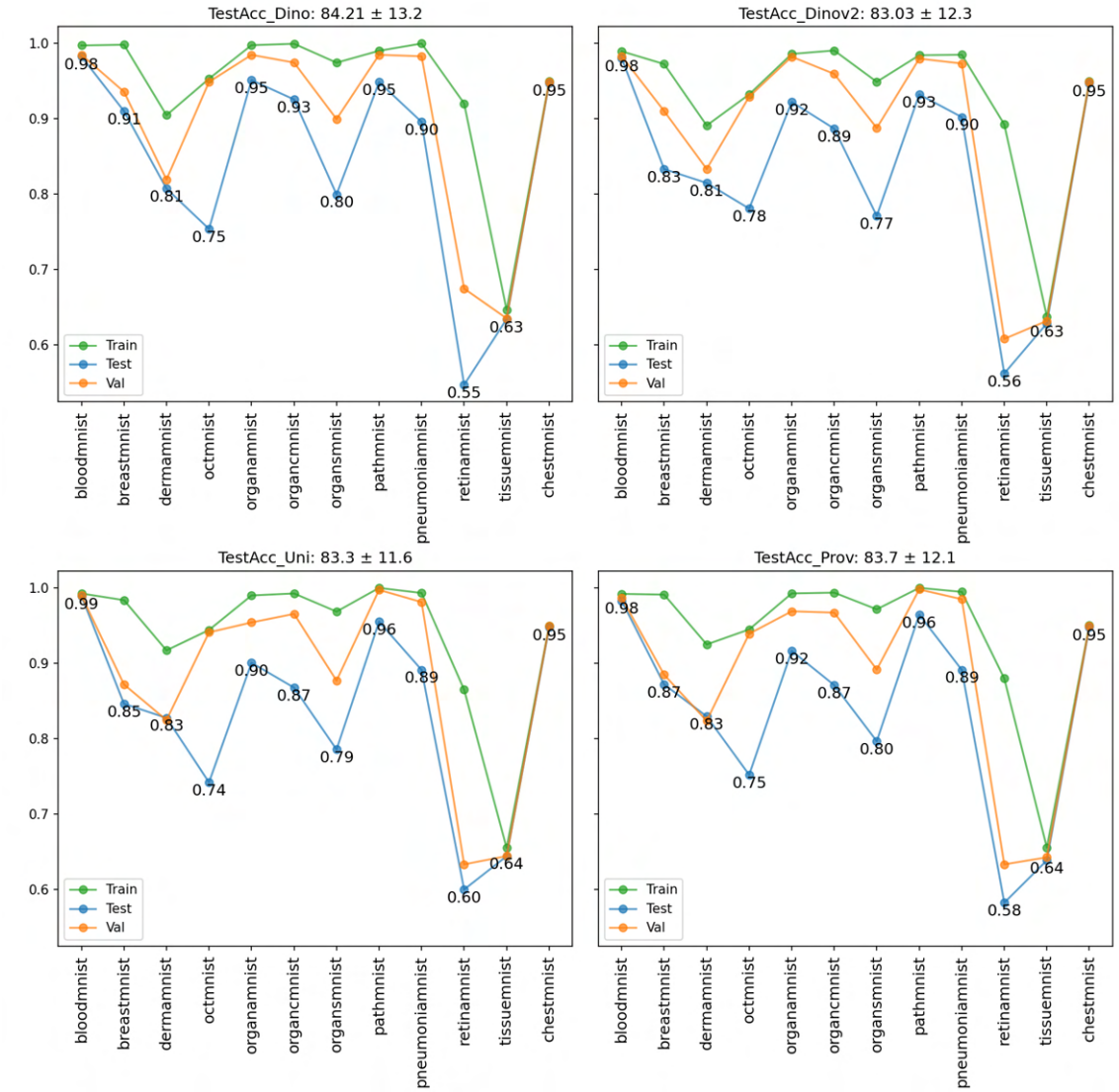


Figure 20: Accuracies for SVMs in 128×128 size for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated.

### Linear Probing

The following examines the AUC as an example metric for linear probing in depth. Figure 22 shows the AUC for Dino, Dinov2, Uni, and Prov as the backbone with a linear layer as a classifier. None of the trained models trained for the maximum number of epochs since the early stopping criterion was triggered every time before reaching the 100th epoch.

Again, the different splits are shown in green (train), orange (validation), and blue (test).

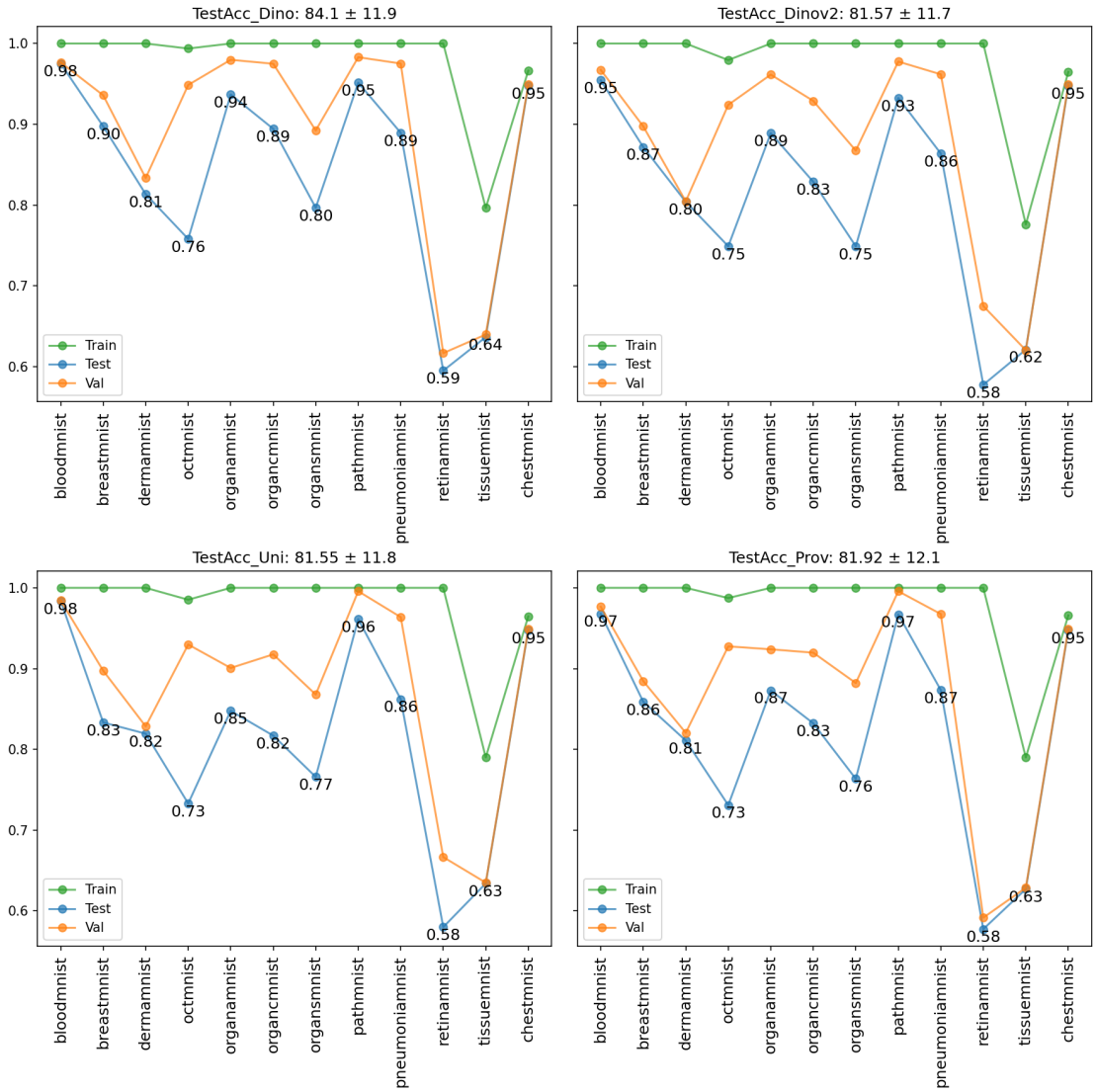


Figure 21: Accuracies for LightGBM in 128×128 size for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated.

The Dino backbone achieves the highest mean AUC, averaging 93.38% with a standard deviation of 8.3% across all 12 tasks. The other backbones exhibit slightly lower average AUC values, but the difference is marginal.

The orange (validation) and blue (test) lines are closely aligned, suggesting that the models perform similarly on validation and test data. For most datasets, the training performance closely matches the validation and test sets. However, exceptions include breastMNIST, dermaMNIST, retinaMNIST, and chestMNIST, where the training AUC is consistently higher than the validation and test AUC for all backbones.

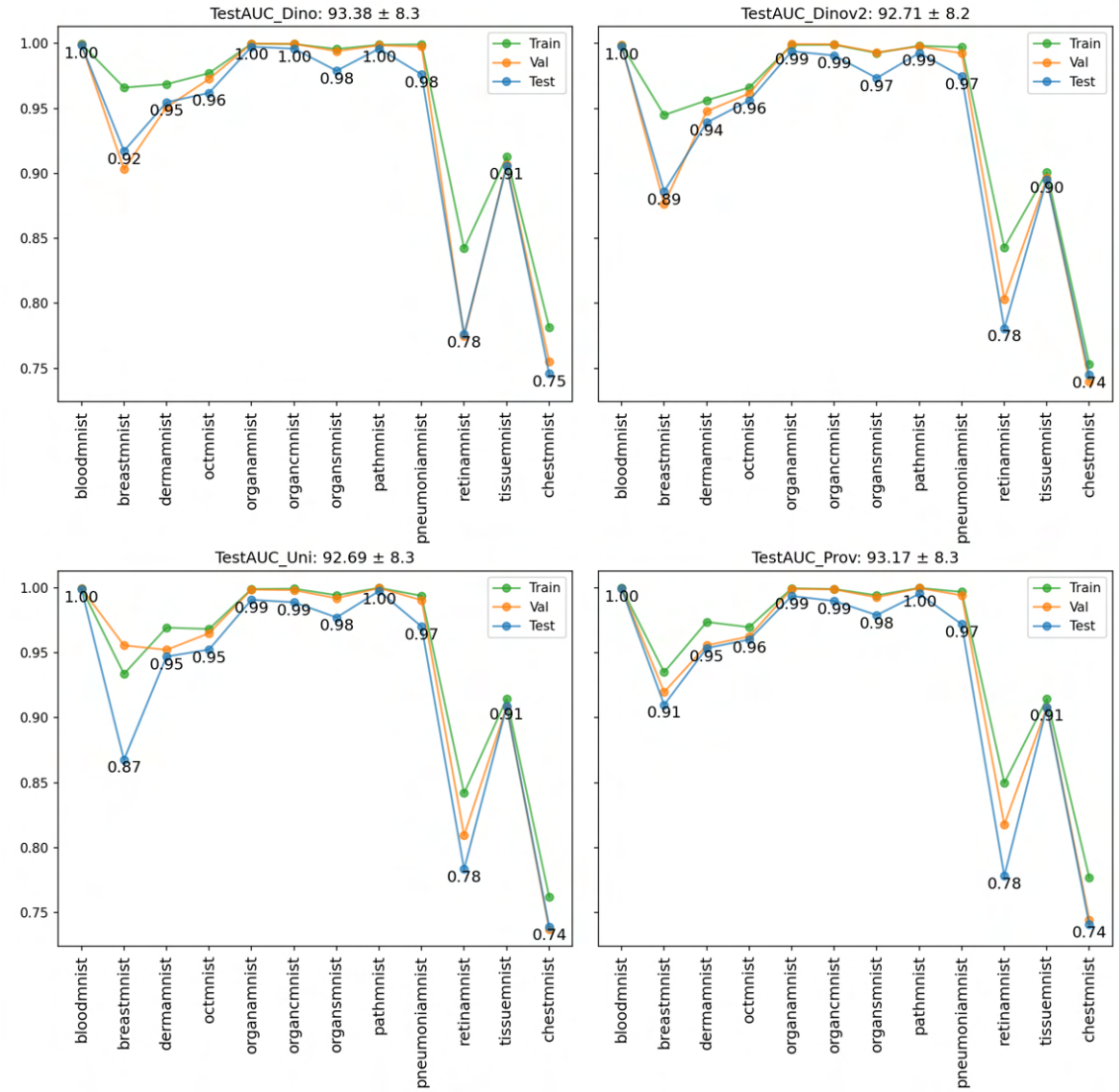


Figure 22: AUCs for linear probing in 64×64 resolution for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated.

Overall, the models demonstrate comparable performance across the various tasks, with minimal variation between backbones.

For the subsequent discussion on multi-domain multi-task pretraining, the balanced accuracies for the 224×224 resolution are also presented in figure 23. In this context, the Dino backbone consistently outperforms the others by at least 1.5 percentage points. Notably, the validation set often performs better than the test set.

This metric also highlights significant discrepancies between training and validation/test performances for specific datasets such as breastMNIST, dermaMNIST, and retinaMNIST, similar to the trends observed in AUC. Additionally, it reveals

instances where test set performance falls below validation performance, as observed for datasets like octMNIST, organCMNIST, organAMNIST, and organSMNIST.

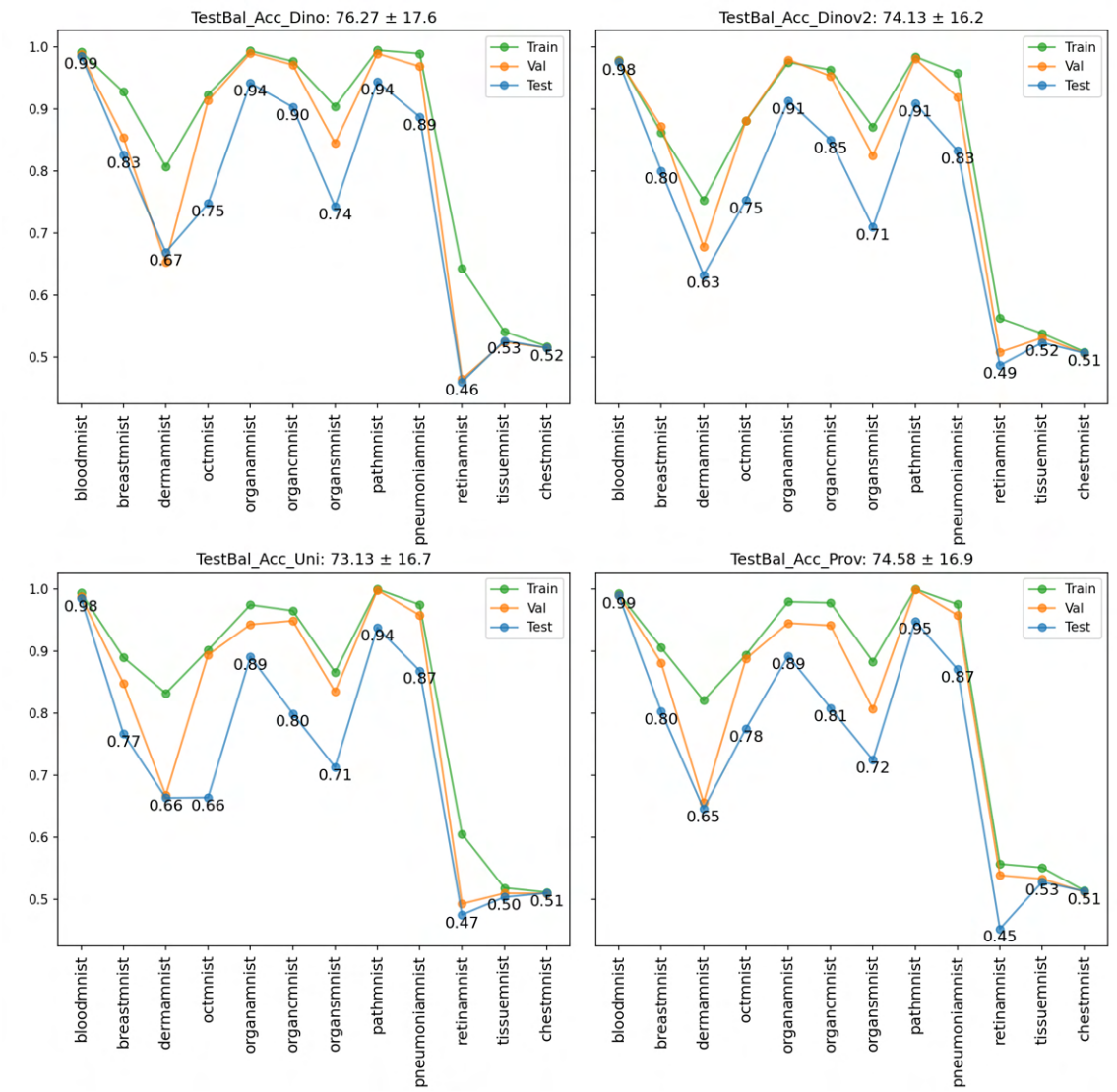


Figure 23: Balanced accuracies for linear probing in 224×224 resolution for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated.

### 5.1.2 Training End-to-End

#### Multi-domain multi-task pre-training

The following figure 24 shows the balanced accuracies for multi-domain multi-task pre-training in the 224×224 resolution. Early stopping is triggered every time after 3 to 15 epochs, depending on the resolution and the backbone.

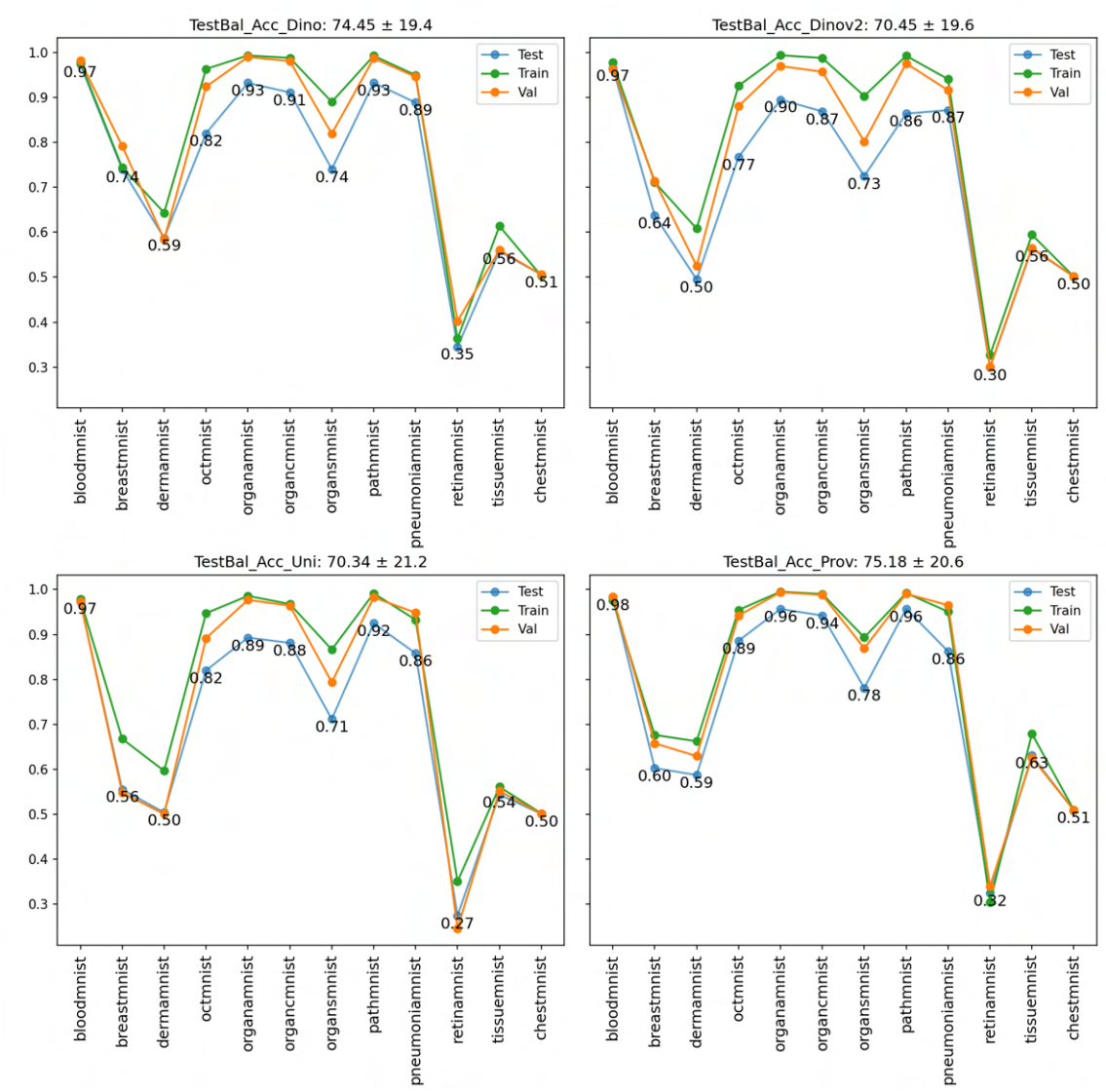


Figure 24: Balanced accuracies for multi-domain multi-task pre-training in 224×224 resolution for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated.

The models demonstrate varying performance depending on the backbone used. Prov achieves the highest balanced accuracy, with an average of 75.18% and a stan-

dard deviation of 20.6%, followed by Dino, with an average balanced accuracy of 74.45% and a standard deviation of 19.4%. In contrast, Dinov2 and Uni exhibit significantly lower performance.

Balanced accuracies vary markedly across tasks, with substantial gaps between training, validation, and test sets. Among these, test data consistently shows the lowest performance. Notably, there is considerable variation in performance across datasets. Prov performs best in most categories, such as organAMNIST, organCMNIST, and organSMNIST, but struggles with breastMNIST classification, where Dino demonstrates superior performance.

Overall, the performance of smaller datasets like breastMNIST and retinaMNIST drops, while the biggest datasets, octMNIST, and tissueMNIST, improve compared to linear probing.

#### Multi-domain multi-task pre-training with data augmentation

Figure 25 shows the balanced accuracies for multi-domain multi-task pre-training with data augmentations in the 224×224 resolution. Early stopping is triggered every time after 9 to 25 epochs, depending on the resolution and the backbone.

The models exhibit significant performance differences depending on the chosen framework. Prov emerges as the best-performing model, achieving a balanced accuracy of 73.27% with a standard deviation of 19%. Uni ranks second with a balanced accuracy of 66.12% and a standard deviation of 20.1%. The results again highlight the discrepancies between the performance of the training, validation, and test data.

Prov dominates in 9 of 12 tasks, often by a substantial margin. The exceptions are bloodMNIST, breastMNIST, and chestMNIST. Although Prov performs nearly on par with the best models for chestMNIST and bloodMNIST, it falls short by three percentage points on breastMNIST, consistent with its underperformance in breastMNIST classification observed during multi-domain multi-task pre-training.

Compared to multi-domain multi-task pre-training, improvements are observed for breastMNIST and retinaMNIST across all backbones, while chestMNIST performance remains unchanged. Prov also shows enhanced performance on dermaMNIST. However, for most other datasets, balanced accuracies decline relative to the multi-domain multi-task pre-training results, most of the time even by a lot.

## 5.2 Results for the MedMNIST-C Dataset

This section presents the results of the models evaluated on MedMNIST-C. All models were trained on the MedMNIST+ dataset and are now evaluated on the same test splits but with the corrupted versions from MedMNIST-C to assess their robustness to distortions. Additional plots for the different training schemes' performance and other resolutions can be found in the Appendix A.3. Table 7 shows the general accuracies and AUCs on the corrupted dataset for every training method

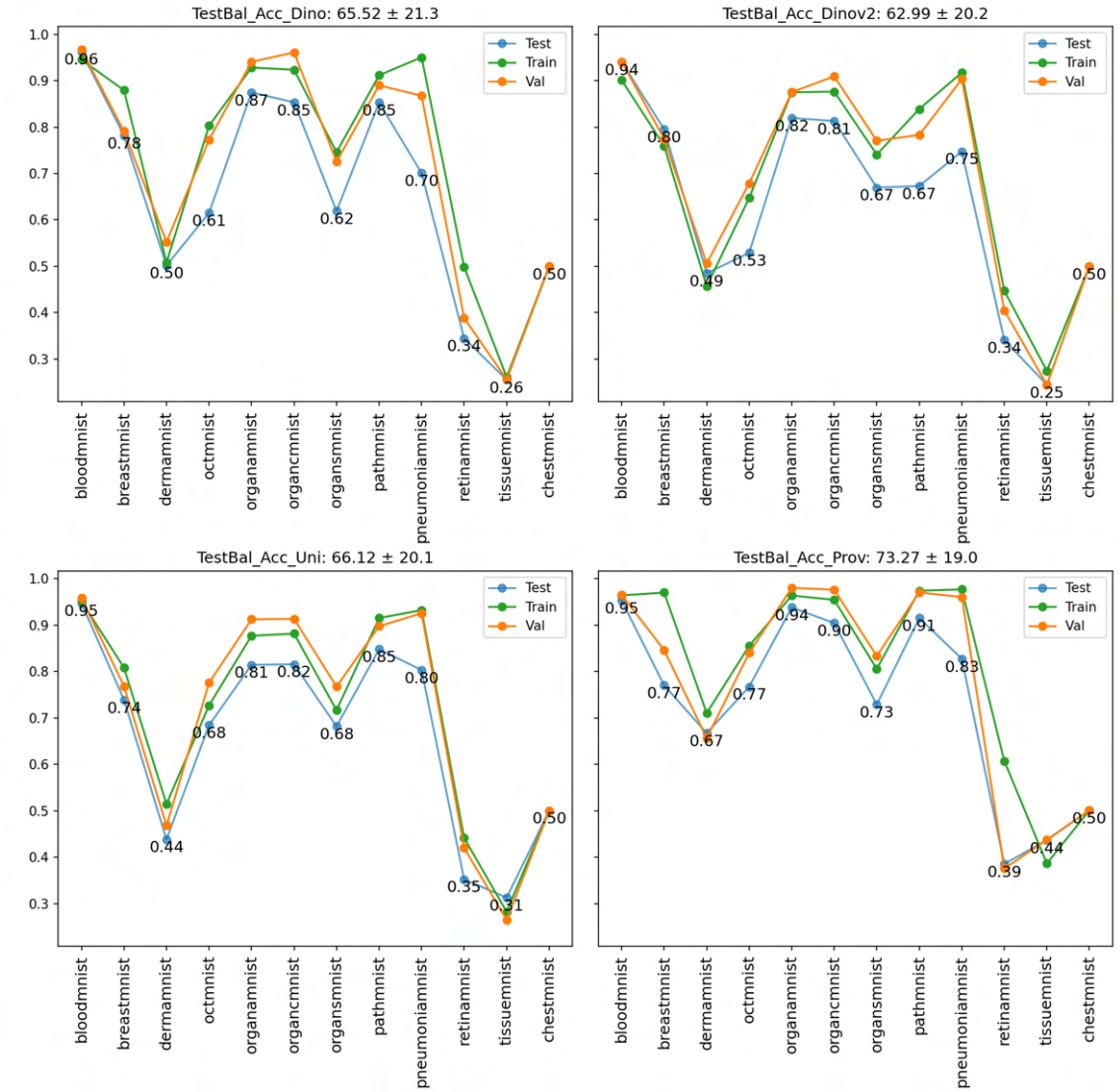


Figure 25: Balanced accuracies for multi-domain multi-task pre-training with data augmentation in 224×224 resolution for every backbone. Training, validation, and test performance are indicated in green, orange, and blue. The title of each plot shows the average and standard deviation over all 12 datasets. The test data is annotated.

with every backbone in the four image resolutions. The overall value in the table is shown as the mean over all classes ± the standard deviation. The best metric for every resolution for a training method is highlighted in bold. The best metric for a resolution overall training methods is highlighted with a background color. kNN and random forest were not evaluated since they performed worse than LightGBM and SVM on MedMNIST+.

Table 7 reveals several significant findings. It summarizes benchmark results showing the mean ± standard deviation for accuracy and AUC across corrupted datasets,

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 65.16 ± 16.9 | 71.8 ± 14.8 | 74.01 ± 15.3 | 73.06 ± 16.2 | **84.44 ± 11.6** | 87.84 ± 10.8 | 89.46 ± 10.6 | 89.74 ± 9.8 |
| Dino LightGBM | **65.66 ± 16.3** | **71.97 ± 14** | **74.47 ± 14.6** | **74.25 ± 15.6** | 84.43 ± 11.3 | **88.37 ± 10** | **90.2 ± 9.9** | **90.21 ± 9.9** |
| Dinov2 SVM | 65.34 ± 15.9 | 68.34 ± 14.7 | 73.52 ± 15.6 | 75.83 ± 15.4 | 83.8 ± 11.4 | 86.03 ± 10.9 | 88.67 ± 10.6 | 89.78 ± 9.8 |
| Dinov2 LightGBM | 63.78 ± 15.7 | 67.01 ± 14.4 | 72.59 ± 14.6 | 74.43 ± 14.4 | 82.65 ± 11.2 | 85.07 ± 10.8 | 88.51 ± 10.6 | 89.9 ± 9.9 |
| UNI SVM | 63.41 ± 16.1 | 65.47 ± 15.9 | 67.19 ± 16.6 | 65.37 ± 17.7 | 83.31 ± 11.2 | 84.88 ± 11.5 | 86.0 ± 10.9 | 85.4 ± 11 |
| UNI LighGBM | 61.67 ± 14.7 | 65.2 ± 15.5 | 65.7 ± 16.1 | 66.4 ± 17.2 | 82.01 ± 11.3 | 84.83 ± 11.1 | 84.82 ± 10.7 | 85.62 ± 10.7 |
| Prov SVM | 64.4 ± 16.1 | 67.59 ± 15.3 | 69.19 ± 15.1 | 68.71 ± 15.2 | 82.53 ± 11.0 | 85.86 ± 10.7 | 86.83 ± 10.3 | 86.53 ± 10.2 |
| Prov LightGBM | 62.35 ± 15.0 | 65.97 ± 15.0 | 67.15 ± 15.7 | 67.59 ± 15.4 | 92.12 ± 0.09 | 94.13 ± 0.09 | 94.55 ± 0.08 | 95.37 ± 0.07 |
| **Linear Probing** | | | | | | | | |
| Dino | **65.63 ± 15.7** | **71.8 ± 14.9** | **74.27 ± 15.4** | 74.11 ± 16.9 | **85.64 ± 10.5** | **89.02 ± 9.7** | **90.95 ± 9.3** | **91.04 ± 9** |
| Dinov2 | 65.48 ± 15.4 | 67.9 ± 14.7 | 74.27 ± 15.0 | **76.15 ± 15.0** | 84.37 ± 10.6 | 86.76 ± 10.1 | 89.86 ± 9.8 | 91.0 ± 9.1 |
| UNI | 63.47 ± 15.3 | 65.79 ± 15.4 | 67.62 ± 16.4 | 67.95 ± 16.9 | 84.05 ± 10.3 | 86.43 ± 10.3 | 87.39 ± 10.0 | 86.39 ± 10.4 |
| Prov | 64.34 ± 15.2 | 67.65 ± 14.9 | 69.74 ± 15.1 | 68.43 ± 16 | 84.3 ± 10.3 | 87.61 ± 10.1 | 88.57 ± 9.6 | 87.88 ± 9.6 |
| **mm-PT** | | | | | | | | |
| Dino | 63.0 ± 14.9 | 69.27 ± 12.2 | 72.79 ± 12.5 | 74.43 ± 12.8 | 82.86 ± 10 | 86.64 ± 9.6 | 88.35 ± 9.9 | 89.33 ± 9.8 |
| Dinov2 | 62.46 ± 15.4 | 64.84 ± 13.9 | 68.66 ± 12.2 | 71.29 ± 12.3 | 82.27 ± 9.5 | 84.29 ± 9.4 | 86.51 ± 10.1 | 86.74 ± 9.9 |
| UNI | 65.16 ± 15.3 | 68.44 ± 15.3 | 73.73 ± 12.7 | 73.79 ± 12.1 | 83.5 ± 10.4 | 86.35 ± 9.4 | 89.14 ± 9.7 | 88.99 ± 9.9 |
| Prov | **68.69 ± 14.6** | **73.91 ± 12** | **78.1 ± 12.1** | **80.73 ± 12.9** | **85.03 ± 10.6** | **88.57 ± 9.3** | **90.64 ± 9.8** | **91.91 ± 9.6** |
| **mm-PT aug** | | | | | | | | |
| Dino | 63.06 ± 16.3 | 63.22 ± 14 | 66.47 ± 14 | 67.37 ± 14.8 | 82.87 ± 10.6 | 83.03 ± 11 | 84.96 ± 11.8 | 85.11 ± 12.2 |
| Dinov2 | 61.33 ± 16.6 | 61.53 ± 14.3 | 63.09 ± 15.1 | 64.07 ± 14.6 | 80.81 ± 11.6 | 82.25 ± 11.3 | 82.48 ± 11.7 | 83.27 ± 11.6 |
| UNI | 63.21 ± 16.4 | 63.26 ± 14 | 65.59 ± 14.2 | 67.72 ± 14.2 | 82.41 ± 11.3 | 83.25 ± 11.3 | 84.46 ± 11.8 | 85.27 ± 12.2 |
| Prov | **64.91 ± 15.9** | **65.91 ± 13.6** | **70.78 ± 13.4** | **75 ± 15.2** | **83.49 ± 11.3** | **84.57 ± 11** | **87.05 ± 11.8** | **89.54 ± 11.7** |

Table 7: Summary of benchmark results presenting the average mean and standard deviation for accuracy and the area under the receiver operating characteristic curve (AUC) across the 12 corrupted datasets of MedMNIST-C, covering all combinations of training schemes, models, and image resolutions. The best result for each resolution across all training schemes and models is highlighted with a background color, while the best result for each training scheme and resolution is highlighted in bold. kNN and random forest were not evaluated because they performed worse than LightGBM and SVM on MedMNIST+.

covering all training schemes, models, and image resolutions. The best result per resolution is highlighted with a background color, while the best per training scheme and resolution is in bold. kNN and random forest were excluded due to inferior performance compared to LightGBM and SVM on the clean dataset.

**Impact of Image Resolution**: Performance steadily improves as image resolution increases from 28×28 to 128×128 for accuracy and AUC. However, beyond 128×128, this trend plateaus, with little to no additional gains observed at 224×224 for most of the pre-trained backbone. Dinov2 is an exception since it shows noticeably better performance on 224×224 than on 128×128.

**Classifier Performance**: Among all training schemes, linear probing, SVM, and LightGBM are at the same level for accuracy, while linear probing shows better results for the AUC.

**Backbone Performance:** Dino is the best performing pre-trained backbone for the image resolutions, while Dinov2 is on eye level for 128×128 and surpasses Dino on 224×224. Prov is the best backbone for the mm-PT and mm-PT aug techniques.

**mm-PT Approaches**: Prov consistently stands out as the top performer for mm-PT training schemes, achieving the highest accuracy across all resolutions and the best AUC at 224×224. Mm-PT training shows clear improvements with higher resolutions, solidifying its position as the leading approach for accuracy with the Prov backbone. While mm-PT achieves comparable AUC to the Dino backbone under linear probing, Dino outperforms in AUC at 28×28, 64×64, and 128×128 resolutions. However, at 224×224, the Prov backbone with mm-PT surpasses Dino in AUC. In contrast, the mm-PT augmentation (mm-PT aug) approach performs significantly weaker, with Prov still maintaining dominance in this context.

Table 8 shows the overall balanced accuracies and Cohen's kappa values on the corrupted MedmNIST-C dataset for each training method with every backbone in the four image resolutions: 28×28, 64×64, 128×128, and 224×224. The total value in the table is shown as the mean over all classes ± the standard deviation. The best metric for every resolution for a training method is highlighted in bold. The best metric for a resolution for all training methods is highlighted with a background color. Since there was no time to evaluate all models, kNN and random forest were not evaluated because they performed worse than LightGBM and SVM on the clean dataset.

Table 8 presents several key insights. The Dino backbone delivers the best overall performance across all classifier-based training methods among the pre-trained models. For both balanced accuracy and Cohen's kappa, the performance of linear probing is better than LightGBM and SVM.

Table 8 supports the findings from the accuracy and AUC for the performance on the MedMNIST-C dataset. It summarizes benchmark results showing the mean ± standard deviation for balanced accuracy and Cohen´s kappa across the corrupted datasets, covering all training schemes, models, and image resolutions. The best result per resolution is highlighted with a background color, while the best per training scheme and resolution is in bold. kNN and random forest were excluded due to inferior performance compared to LightGBM and SVM on the clean dataset.

**Impact of Image Resolution**: Performance steadily improves as image resolution increases from 28×28 to 128×128 for balanced accuracy and Cohen´s kappa. However, beyond 128×128, this trend plateaus, with little to no additional gains observed at 224×224 for most of the pre-trained backbone. Dinov2 is an exception since it shows noticeably better performance on 224×224 than on 128×128.

**Classifier Performance**: Among all training schemes, linear probing slightly outperforms SVM, while LightGBM falls somewhat off.

**Backbone Performance**: Dino is the best performing pre-trained backbone for the image resolutions, while Dinov2 is on eye level for 128×128 and surpasses Dino on 224×224. Prov and UNI as backbones fall off in comparison.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | **55.28 ± 17.7** | **61.51 ± 18.4** | **64.93 ± 19** | 64.51 ± 19.4 | **43.01 ± 23** | **50.46 ± 24.8** | **54.57 ± 25.9** | 53.6 ± 26.1 |
| Dino LightGBM | 54.19 ± 17 | 59.98 ± 17.9 | 62.99 ± 18.5 | 62.72 ± 19 | 42.45 ± 22 | 49.9 ± 24.1 | 53.55 ± 25 | 53.14 ± 25.6 |
| Dinov2 SVM | 54.76 ± 17 | 57.85 ± 16.8 | 64.23 ± 18.3 | **66.48 ± 19** | 42.44 ± 22.1 | 46.24 ± 22.6 | 53.86 ± 25.9 | **57.49 ± 26.1** |
| Dinov2 LightGBM | 51.58 ± 15.8 | 54.62 ± 15.9 | 61.64 ± 17.7 | 63.46 ± 18.3 | 38.83 ± 20.6 | 42.31 ± 22.0 | 51.77 ± 24.6 | 54.38 ± 25.3 |
| UNI SVM | 52.88 ± 15.9 | 55.9 ± 16.2 | 57.62 ± 17.7 | 56.64 ± 18.8 | 40.04 ± 21 | 42.2 ± 23.2 | 45.03 ± 23.6 | 43.09 ± 25.7 |
| UNI LighGBM | 50.45 ± 15 | 53.3 ± 16.4 | 54.23 ± 17.4 | 54.49 ± 18.8 | 37.25 ± 19.7 | 40.17 ± 22.4 | 41.53 ± 23.3 | 41.61 ± 24.9 |
| Prov SVM | 54.04 ± 16.1 | 58.27 ± 16.4 | 59.94 ± 17.3 | 59.71 ± 17.7 | 41.23 ± 21.7 | 45.26 ± 23.4 | 47.86 ± 24.0 | 47.69 ± 24.2 |
| Prov LightGBM | 51.13 ± 14.7 | 54.65 ± 15.4 | 55.53 ± 17.1 | 55.25 ± 18.1 | 37.98 ± 19.4 | 42.0 ± 21.3 | 43.5 ± 22.8 | 43.08 ± 24.0 |
| **Linear Probing** | | | | | | | | |
| Dino | **55.29 ± 16.7** | **62 ± 17** | **65.15 ± 18.3** | 66.16 ± 18.8 | **43.67 ± 21.5** | **50.9 ± 24.1** | **55.15 ± 25.2** | 55.62 ± 26.4 |
| Dinov2 | 53.88 ± 16.0 | 56.84 ± 16.0 | 64.63 ± 17.6 | **66.67 ± 17.9** | 41.43 ± 20.9 | 44.85 ± 22.1 | 54.35 ± 25.4 | **57.49 ± 25.7** |
| UNI | 51.89 ± 15.4 | 55.62 ± 15.5 | 56.87 ± 17 | 57.71 ± 18.1 | 938.83 ± 20.3 | 41.75 ± 23.1 | 43.76 ± 23, 4 | 44.92 ± 25.3 |
| Prov | 53.37 ± 15.3 | 57.91 ± 15.5 | 60.55 ± 16.8 | 60.11 ± 17.5 | 40.84 ± 20.1 | 45.04 ± 22.5 | 48.36 ± 23.9 | 48.27 ± 24.5 |
| **mm-PT** | | | | | | | | |
| Dino | 51.04 ± 14.9 | 59.16 ± 15.2 | 61.81 ± 15.7 | 65.87 ± 16.4 | 37.67 ± 20.1 | 47.54 ± 21.4 | 50.81 ± 23.4 | 55.53 ± 24.3 |
| Dinov2 | 52.59 ± 15.2 | 54.28 ± 14.1 | 59.21 ± 14.5 | 61.18 ± 15.4 | 40.35 ± 19 | 41.88 ± 18.8 | 47.23 ± 21.3 | 49.86 ± 23 |
| UNI | 53.54 ± 15.2 | 57.52 ± 15.4 | 62.87 ± 17.1 | 62.06 ± 17.2 | 41.55 ± 21.2 | 45.41 ± 21.9 | 52.31 ± 25.1 | 51.21 ± 25.3 |
| Prov | **56.43 ± 18.2** | **62.97 ± 15.9** | **68.27 ± 17.3** | **70.8 ± 19.2** | **45.49 ± 22.9** | **52.52 ± 24** | **58.55 ± 25.7** | **61.75 ± 27.6** |
| **mm-PT aug** | | | | | | | | |
| Dino | 52.71 ± 17.1 | 52.71 ± 14.1 | 55.48 ± 16.3 | 56.41 ± 17.3 | 40.42 ± 21.2 | 40.15 ± 18.6 | 43.53 ± 21.4 | 44.04 ± 23.8 |
| Dinov2 | 48.85 ± 15.9 | 51.2 ± 14.2 | 52.14 ± 15.1 | 53.54 ± 15.4 | 36.85 ± 18.7 | 38.19 ± 18.3 | 39.61 ± 19.5 | 41.5 ± 20 |
| UNI | 52.83 ± 16.7 | 53.33 ± 14.3 | 55.75 ± 14.9 | 56.77 ± 16.5 | 40.83 ± 20.4 | 40.64 ± 19.3 | 43.8 ± 20.6 | 45.41 ± 22.2 |
| Prov | **53.68 ± 17.7** | **55.09 ± 15.5** | **61.09 ± 17.3** | **66.29 ± 18.4** | **41.98 ± 22** | **43.13 ± 20.8** | **50.47 ± 22.3** | **55.98 ± 26** |

Table 8: Summary of benchmark results presenting the average mean and standard deviation for balanced accuracy and Cohen´s kappa across the 12 corrupted datasets of MedMNIST-C, covering all combinations of training schemes, models, and image resolutions. The best result for each resolution across all training schemes and models is highlighted with a background color, while the best result for each training scheme and resolution is highlighted in bold. Since there was no time to evaluate all models, kNN and random forest were not evaluated because they performed worse than LightGBM and SVM on the clean dataset.

**mm-PT Approaches**: For mm-PT training schemes, Prov consistently excels, achieving the highest balanced accuracy and Cohen's kappa across all resolutions. It outperforms all other training schemes in every category, solidifying its position as the top-performing model on MedMNIST-C. Mm-PT training demonstrates clear improvements with higher resolutions, further reinforcing the dominance of the Prov backbone. In contrast, the mm-PT augmentation (mm-PT aug) approach shows significantly weaker performance, though Prov remains the dominant backbone.

Since balanced accuracy at the 224×224 resolution is the most relevant metric for the corrupted dataset, we focus our analysis exclusively on this metric.

Figure 26 compares the balanced accuracies of two of the best training methods on MedMNIST-C - Dino linear probing and Prov mm-PT — at a resolution of 224×224. The top plots show results for the Dino backbone, while the bottom plots present those for the Prov backbone. Performance on the clean MedMNIST+ dataset is

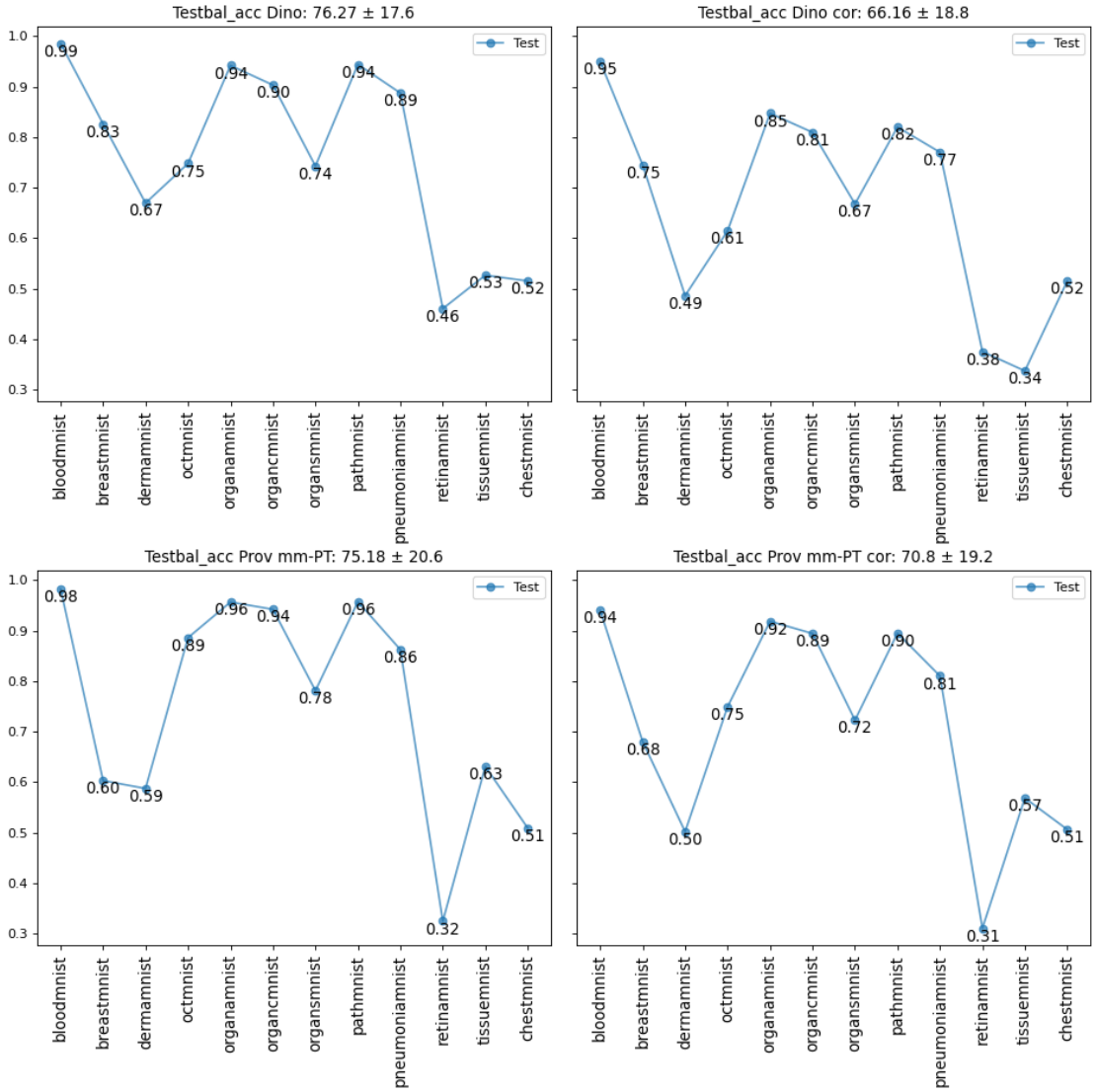


Figure 26: Balanced accuracies for two of the best training methods at a resolution of 224×224 on the two datasets are illustrated. The top plots represent Dino Linear Probing, while the bottom plots correspond to Prov mm-PT. Performance on MedMNIST+ is shown in the left column, whereas performance on MedMNIST-C is depicted in the right column.

displayed on the left, and performance on the corrupted MedMNIST-C dataset is shown on the right.

The figure reveals that the Dino backbone with linear probing achieves noticeably better performance on non-corrupted datasets such as breastMNIST, dermaMNIST, retinaMNIST, and pneumoniaMNIST, which are the four smallest datasets in terms of training samples. ChestMNIST shows similar performance for both backbones. The Prov backbone with the mm-PT training scheme outperforms Dino for all other datasets.

Prov with mm-PT dominates in nearly all categories when evaluating the corrupted dataset. Notably, it performs significantly better on the two largest datasets regarding training samples, tissueMNIST and octMNIST. However, the performance of the two smallest datasets, breastMNIST and retinaMNIST, is considerably worse compared to the pre-trained Dino backbone. For chestMNIST and dermaMNIST, the performance is almost identical between the two methods, with chestMNIST maintaining the same balanced accuracy as observed on the clean dataset.

| | bloodmnist | breastmnist | chestmnist | dermamnist | octmnist | otganamnist | organcmnist | organsmnist | pathmnist | pneumoniamnist | retinamnist | tissuemnist |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **JPEG** | $96.02_{\pm 3.1}$ | $60.04_{\pm 1.7}$ | $50.76_{\pm 0.1}$ | $52.65_{\pm 4.2}$ | $82.50_{\pm 6.4}$ | $95.55_{\pm 0.2}$ | $93.88_{\pm 0.4}$ | $77.73_{\pm 0.3}$ | $88.63_{\pm 6.3}$ | $84.10_{\pm 1.4}$ | $32.23_{\pm 0.4}$ | $55.38_{\pm 6.2}$ |
| **Pixelate** | $97.77_{\pm 0.5}$ | $77.38_{\pm 2.5}$ | $50.68_{\pm 0.3}$ | $55.58_{\pm 2.0}$ | $71.66_{\pm 9.5}$ | $95.48_{\pm 0.2}$ | $94.25_{\pm 0.1}$ | $78.12_{\pm 0.2}$ | $94.17_{\pm 1.7}$ | $82.18_{\pm 1.8}$ | $31.35_{\pm 0.8}$ | $61.77_{\pm 1.5}$ |
| **Gaussian Noise** | - | - | $50.32_{\pm 0.3}$ | $53.51_{\pm 6.2}$ | - | $93.49_{\pm 2.4}$ | $91.72_{\pm 2.4}$ | $74.52_{\pm 3.1}$ | - | $85.34_{\pm 0.4}$ | $34.24_{\pm 0.8}$ | - |
| **Speckle Noise** | - | $61.08_{\pm 0.7}$ | $50.36_{\pm 0.3}$ | $51.97_{\pm 6.7}$ | $73.32_{\pm 11.5}$ | $93.62_{\pm 2.6}$ | $91.98_{\pm 2.7}$ | $74.80_{\pm 3.5}$ | - | $85.24_{\pm 1.1}$ | $32.40_{\pm 0.6}$ | - |
| **Impulse Noise** | - | - | $50.27_{\pm 0.3}$ | $53.11_{\pm 5.9}$ | - | $94.60_{\pm 0.9}$ | $93.00_{\pm 1.1}$ | $76.08_{\pm 1.8}$ | - | $84.66_{\pm 0.5}$ | - | $54.50_{\pm 2.8}$ |
| **Shot Noise** | - | - | $50.12_{\pm 0.2}$ | $48.33_{\pm 6.6}$ | - | $94.40_{\pm 1.5}$ | $92.68_{\pm 1.5}$ | $75.37_{\pm 2.8}$ | - | $85.24_{\pm 1.1}$ | - | - |
| **Defocus Blur** | $90.59_{\pm 7.8}$ | - | - | $43.40_{\pm 4.6}$ | $82.62_{\pm 4.8}$ | - | - | - | $73.69_{\pm 17.0}$ | - | $28.28_{\pm 1.4}$ | - |
| **Motion Blur** | $94.05_{\pm 3.8}$ | $71.84_{\pm 4.4}$ | - | $45.56_{\pm 6.1}$ | $67.82_{\pm 12.7}$ | - | - | - | $83.71_{\pm 8.4}$ | - | $26.74_{\pm 1.2}$ | - |
| **Gaussian Blur** | - | - | $51.01_{\pm 0.0}$ | - | - | $83.06_{\pm 7.1}$ | $79.14_{\pm 9.7}$ | $58.95_{\pm 7.3}$ | - | $72.97_{\pm 6.7}$ | - | $60.89_{\pm 1.4}$ |
| **Zoom Blur** | - | - | - | $51.05_{\pm 2.5}$ | - | - | - | - | - | - | - | - |
| **Brightness Up** | $93.97_{\pm 4.0}$ | $58.16_{\pm 1.9}$ | $50.71_{\pm 0.1}$ | $47.49_{\pm 7.5}$ | - | $86.42_{\pm 6.1}$ | $82.11_{\pm 8.2}$ | $61.20_{\pm 8.9}$ | $88.95_{\pm 3.4}$ | $77.35_{\pm 7.5}$ | - | $57.32_{\pm 2.6}$ |
| **Brightness Down** | $87.17_{\pm 14.2}$ | $78.28_{\pm 3.5}$ | $50.50_{\pm 0.2}$ | $41.51_{\pm 13.2}$ | - | $90.07_{\pm 4.2}$ | $87.03_{\pm 4.0}$ | $73.06_{\pm 3.3}$ | $92.64_{\pm 1.4}$ | $72.79_{\pm 8.9}$ | $30.96_{\pm 1.7}$ | $50.16_{\pm 7.7}$ |
| **Contrast Up** | $97.10_{\pm 0.9}$ | - | $50.72_{\pm 0.1}$ | $54.94_{\pm 3.4}$ | - | $90.96_{\pm 3.9}$ | $87.31_{\pm 4.2}$ | $68.87_{\pm 5.6}$ | $92.08_{\pm 3.3}$ | $89.05_{\pm 0.6}$ | - | $58.16_{\pm 4.9}$ |
| **Contrast Down** | $94.01_{\pm 6.6}$ | $69.21_{\pm 5.8}$ | $50.63_{\pm 0.2}$ | $48.26_{\pm 8.5}$ | $71.58_{\pm 11.5}$ | $93.93_{\pm 1.3}$ | $91.69_{\pm 1.8}$ | $76.22_{\pm 1.9}$ | $90.50_{\pm 4.0}$ | $73.27_{\pm 10.7}$ | $31.94_{\pm 2.6}$ | $56.44_{\pm 6.1}$ |
| **Saturate** | $97.36_{\pm 0.7}$ | - | - | - | - | - | - | - | $94.63_{\pm 1.8}$ | - | - | - |
| **Stain Deposit** | $91.16_{\pm 3.4}$ | - | - | - | - | - | - | - | $91.01_{\pm 1.9}$ | - | - | - |
| **Bubble** | $94.91_{\pm 1.6}$ | - | - | - | - | - | - | - | $94.48_{\pm 1.1}$ | - | - | - |
| **Gamma Corr. Up** | - | - | $50.80_{\pm 0.1}$ | - | - | $91.84_{\pm 3.1}$ | $89.34_{\pm 3.1}$ | $74.61_{\pm 2.8}$ | - | $89.36_{\pm 0.7}$ | - | - |
| **Gamma Corr. Down** | - | - | $50.70_{\pm 0.2}$ | - | - | $90.65_{\pm 6.8}$ | $88.44_{\pm 7.8}$ | $69.91_{\pm 9.0}$ | - | $71.67_{\pm 11.9}$ | - | - |
| **Black Corners** | - | - | - | $47.51_{\pm 1.9}$ | - | - | - | - | - | - | - | - |
| **Characters** | - | - | - | $57.56_{\pm 0.9}$ | - | - | - | - | - | - | - | - |

Table 9: A detailed summary of benchmark results showing balanced accuracy for each dataset in MedMNIST-C and each type of corruption, with the average and standard deviation calculated across all severity levels. The results are based on the best training scheme: Prov mm-PT at resolution 224×224.

Table 9 emphasizes that different types of corruptions affect datasets in varying ways, with the extent of the impact mainly depending on the characteristics of the specific dataset. For instance, a corruption that results in a significant performance decline in one dataset might have only a minor effect on another. For example, when comparing bloodMNIST and breastMNIST, Brightness Down is the worst-performing corruption for bloodMNIST but the best-performing one for breastMNIST. This pattern is consistent across all datasets. Notably, corruption has no observable impact on chestMNIST. The varying severities have different impacts on performance. For example, in the case of bloodMNIST, the standard deviation for the "Contrast Up" corruption is 0.9%, whereas for the "Brightness Down" corruption, the standard deviation is 14.2%. This demonstrates a significant difference in how the severity

levels influence the dataset. The tables for the five different severities independently can be found in the appendix A.4.

# 6 Comparison and Discussion

This section provides context for the results and begins the discussion with the results of the clean dataset. Since the results show that Random Forest and kNN are significantly weaker, they get no further attention.

## 6.1 Performance on MedMNIST+

#### Different performance on the 12 different tasks

The trends observed across different training methods suggest that some datasets are inherently easier to classify while others pose more significant challenges. For example, BloodMNIST and PathMNIST consistently demonstrate high performance across all metrics and training schemes. In contrast, RetinaMNIST represents the opposite extreme, exhibiting the lowest performance among the datasets. This disparity may be attributed to the limited number of training samples in RetinaMNIST and its focus on ordinal regression, a unique characteristic of the dataset. Various performance levels can be influenced by several factors, including the number of training samples, the nature of the data modalities, the underlying tasks, and, most significantly, the varying difficulty levels of these tasks.

#### Majority class vote for chestMNIST

ChestMNIST yields some interesting results. The dataset achieves an accuracy of approximately 95% (the same as a classifier that would only predict the majority class), a balanced accuracy of around 50%, and a Cohen´s kappa close to 0 for all training schemes and resolutions. Notably, it is also the only dataset unaffected by corruption. These findings suggest that the model predominantly predicts the majority class, resulting in high overall accuracy. However, this behavior indicates that the classifier fails to differentiate between classes as intended.

#### Improvement with higher resolution

The results indicate that the model performance improves as the image resolution increases from 28×28 to 64×64, and 128×128. However, this improvement levels off when increasing the resolution from 128×128 to 224×224. That is because an image size of 224×224 is generally better for classification tasks than 28×28. The larger size retains more detailed features and finer spatial information, allowing the model to better capture the complexities of the data. The smaller 28×28 images may lose essential details, reducing the ability to distinguish between different classes, especially for tasks that require recognizing intricate patterns or textures.

### Influence of the backbone

Dino is pre-trained on a broad range of data, which likely enhances its generalization capabilities. In contrast, Prov and UNi are pre-trained on medical images, which may give them a deeper understanding of domain-specific features in the medical field but could limit their adaptability to other medical data modalities or more general datasets. Additionally, Dinov2, which has a higher input resolution, required more zero padding, and this additional padding could negatively influence its performance.

### Comparison between linear probing and SVM and LightGBM

As the results show, linear probing slightly outperforms both SVMs and LightGBM. This is primarily because linear probing exhibits the least overfitting, with smaller gaps between training and validation performance than the other models.

Foundation models map input images to a high-dimensional feature space, enabling classifiers to extract patterns. However, SVMs with linear kernels and linear classifiers struggle to capture the non-linear relationships in this space. Despite their strength in handling high-dimensional data and maximizing margins, the linear kernel limits SVMs to linear patterns, underutilizing rich feature representations.

In contrast, LightGBM is designed to model non-linear relationships, leveraging gradient boosting to combine multiple weak learners (trees) into a strong predictive model. However, its performance in this context falls short compared to SVM and linear probing, primarily due to insufficient hyperparameter tuning and a tendency to overfit. LightGBM has the potential to outperform linear classifiers and SVMs when adequately fine-tuned. Optimizing hyperparameters such as learning rate, tree depth, and regularization techniques can significantly reduce overfitting and improve its generalization ability, enabling it to better capture the complex, non-linear patterns within the feature space.

LightGBM's overfitting is evident in its significantly better performance on the training data compared to validation and test sets, particularly on smaller datasets like breastMNIST, retinaMNIST, and dermaMNIST. These datasets, being among the smallest in MedMNIST, are more susceptible to overfitting. For example, breastMNIST and retinaMNIST, the two smallest datasets, see LightGBM overfitting most severely. In contrast, although the linear SVM and the linear classifier cannot model non-linear relationships effectively, they exhibit smaller training-validation gaps, benefiting from their more straightforward structure and fewer degrees of freedom. However, if such patterns exist, their linear nature can limit their ability to fully exploit the rich, non-linear patterns in the feature space generated by foundation models.

Since LightGBM shows the most significant overfitting, effective hyperparameter optimization could enhance its performance, allowing it to outperform SVM and linear classifiers. Additionally, experimenting with SVM using non-linear kernels

is worthwhile, as they better capture non-linear patterns in the data, leading to improved results.

### Comparison between linear probing and multi-domain multi-task pre-training

When comparing linear probing with multi-domain multi-task pre-training as training methods, linear probing outperforms in 3 out of 4 backbones: Dino, Dinov2, and Uni. Prov is a notable exception, as it shows partially better results with multi-domain multi-task pre-training across various metrics.

An analysis of the metrics reveals a notable trend: The two largest training datasets, octMNIST and tissueMNIST, demonstrate performance improvements across all backbones. In contrast, three of the four smallest datasets — breastMNIST, dermaMNIST, and retinaMNIST — show significant performance declines. This indicates that larger datasets disproportionately influence the backbone's training in multi-domain multi-task pre-training. In comparison, smaller datasets contribute minimally or not at all, resulting in an imbalance in performance. This occurs because the larger datasets receive significantly more attention during training.

For Dinov2 and UNI, the performance of all other datasets worsens, while for Dino, the results remain relatively stable. Prov, however, shows improvements across most datasets, making it the best-performing model under the multi-domain multi-task pre-training scheme. This success may be attributed to Prov's significantly larger number of parameters, which likely enables it to better leverage the information provided by the mm-PT training technique.

### Comparison between multi-domain multi-task pre-training and multi-domain multi-task pre-training with data augmentation

When comparing the two training methods, mm-PT outperforms mm-PT aug in nearly every category. Mm-Pt aug performs significantly worse for the three largest datasets: tissueMNIST, octMNIST, and pathMNIST. Although most other datasets, except for retinaMNIST and breastMNIST, also perform worse, the decline is less pronounced. Notably, breastMNIST, the smallest dataset, performs better across all backbones, and retinaMNIST, the second smallest dataset, shows slight improvement for all backbones except Dino. This indicates a reversed weighting effect in mm-PT, where smaller datasets receive too much emphasis while larger datasets receive too little.

In addition, the training process itself contributes to this imbalance. The model processes only 69 batches (4,416 images) per dataset per epoch and trains for 9 to 25 epochs – depending on the backbone and resolution – before early stopping is triggered. As a result, larger datasets like octMNIST and pathMNIST are not fully utilized, which explains their poorer performance under this training scheme.

An attempt was made to weight each dataset equally in mm-PT, ensuring that every dataset contributed the same to the model's learning. However, this approach yielded similar results to mm-PT aug, placing excessive emphasis on the smaller datasets and thus reducing the performance of the bigger datasets. This also shows that different datasets require different weights to achieve optimal performance when training end-to-end.

## 6.2 Performance on MedMNIST-C

### Influence of image size

When analyzing the clean dataset, the improvement from 128×128 to 224×224 resolution was minimal, if any. However, the difference between the two resolutions was observable for the corrupted datasets, with the 224×224 resolution performing notably better, especially with the mm-PT approach from Prov. This is likely because corruptions significantly impact smaller resolutions, as the alterations are more pronounced in smaller images. Larger image sizes, such as 224×224, allow for more detailed feature extraction, making them more effective for classification, mainly when dealing with corrupted data.

### Influence of the applied corruptions

The impact of different corruptions on performance varies depending on the dataset. This is because datasets can differ significantly in their modality, structure, and the nature of the information they contain. Each dataset interacts uniquely with various corruptions, leading to differences in how much a particular corruption affects its performance. This variation could be influenced by factors such as the dataset's inherent complexity, the robustness of the features it relies on, or its sensitivity to particular perturbations. For example, datasets with fine-grained textures may be more susceptible to blurring, as this corruption removes critical high-frequency details. Conversely, datasets that rely more on color or coarse shapes might be more affected by color distortions or pixelation.

### Influence of the different severities of the corruptions

The varying severities of corruption can have a different impact on performance because the extent of the corruption directly affects the degree to which critical information in the dataset is altered. The corruption may only slightly distort the data at low severities, potentially allowing the model to recover or ignore minor changes, resulting in minimal performance loss. However, as the severity increases, the corruption becomes more pronounced, leading to more significant disruptions in the data, which can hinder the model's ability to interpret or classify the information correctly.

For example, in image datasets, a low-level blur may only obscure fine details, but a higher-level blur could distort key features essential for classification or segmentation. Similarly, with pixelation or color distortions, low severity might only affect a small portion of the image. In contrast, higher severities could result in substantial loss of critical visual cues, making it harder for the model to differentiate between categories.

The model's robustness and ability to generalize also play a role. Models that learn invariant features may perform reasonably well under moderate corruption levels. Still, as the severity of corruption surpasses the model's capacity to handle distortions, its performance degrades sharply. Thus, understanding the relationship between corruption severity and model performance is crucial for evaluating how resilient a model is to real-world challenges.

### Influence of the backbone

Dino demonstrates the best performance when comparing pre-trained models where only the classifier is fine-tuned. This superiority can be attributed to two key reasons:

Strong Backbone Performance on MedMNIST+: The Dino backbone already exhibits the best performance on clean datasets, providing a solid foundation for the classifier to build upon. This indicates that Dino's pre-trained features are the most transferable and robust across the different tasks.

Diverse pre-training data: Dino is pre-trained on a wide variety of data, which likely improves its generalization capabilities and robustness to different corruptions. Dinov2 is also trained on diverse data but with higher input size, making its impact more apparent at higher resolutions. Prov and UNi, on the other hand, are pre-trained exclusively on medical images, potentially giving them a firmer grasp of medical domain-specific features but possibly limiting their adaptability to other medical data modalities or more general datasets.

### Influence of training the backbone

When examining the impact of training the backbone, the Prov backbone demonstrates superior performance with the mm-PT training scheme, surpassing all other models. This is attributed to the Prov backbone's ability to learn more effectively under this specific training method compared to alternative backbones. The mm-PT training scheme appears particularly well-suited to unlocking the potential of the Prov backbone, allowing it to develop stronger feature representations.

Training the backbone improved the overall performance and significantly enhanced the model's robustness to distortions. By fine-tuning the backbone, the model becomes better equipped to handle various data corruptions or perturbations, ensuring more consistent performance across diverse and noisy datasets. This highlights the

importance of aligning the training strategy with the backbone architecture to maximize learning efficacy and resilience to challenging data conditions.

Prov with the mm-PT training scheme outperforms the pre-trained Dino backbone, which ranks as the second-best model across 9 out of the 12 datasets. An exception is ChestMNIST, where classification relies solely on predicting the majority class, resulting in no performance differences between the models. On the other hand, BreastMNIST and RetinaMNIST exhibit poorer performance. This is likely due to their small dataset sizes, which limited the attention they received during training with the mm-PT scheme, preventing the model from fully leveraging its potential on these datasets.

# 7 Limitations and Future Work

This section discusses the methods' limitations and proposes ideas to address these limitations. It begins with suggestions for improving training schemes that incorporate future embeddings of pre-trained models.

## 7.1 Training Methodology

Due to time constraints, all training schemes were trained and evaluated using a single random seed. However, submissions to the benchmark introduced in this thesis will require at least three seeds, as utilizing multiple seeds reduces variance in the results and facilitates a more reliable comparison of the models.

### 7.1.1 Training only the Classification Heads

Because of the limited time and since the results of this thesis should only function as a reference point, the thesis opted to find a hyperparameter configuration that works well for every dataset. However, since the different datasets are very diverse regarding the classification task, the data modalities, the number of samples, and the balancing of the different classes within the individual datasets, they need different hyperparameters to optimize the results. So, to improve the performance of all datasets, every dataset needs its own individual hyperparameter optimization. As mentioned, various methods, such as Grid Search, can be used to identify better hyperparameter settings.

Since the training data for isolated datasets is limited and the classes are often imbalanced, expanding the datasets within this training scheme is a logical step. This approach increases the volume and balance of the training data, ensuring more diverse datasets. As a result, it can enhance the overall performance of individual models.

### 7.1.2 Training End-to-End

The results from the multi-domain multi-task pre-training approaches, with and without data augmentations, indicate that larger datasets either receive too much or too little weight. Some datasets are more straightforward to classify and require less weight, while others are more challenging and require more weight. Therefore, it would be beneficial to assign the weights of the dataset independently under the supervision of a supervisor. Although this approach could improve the overall performance of the models, it demands a deeper understanding of the datasets and their interactions, making it a time-consuming attempt for optimization.

Another potential limitation of this training method is using a single optimizer for all 12 datasets combined. When the model is exposed to a sample from dataset

A, optimized through the chosen optimizer, and then sees a significant amount of data from other datasets before returning to dataset A, the optimizer and learning rate may no longer be suitable for adjusting the classification head of dataset A. A possible solution would be to use 12 separate optimizers for the 12 different classification heads, allowing the learning rate and optimizer to be tailored to the specific data modalities of each head.

### 7.1.3 Combination of both Approaches

Another possibility for improving the performance of the models could be a combination of both approaches. The backbones of the multi-domain multi-task pre-training are used to train 12 new classifiers independently on the feature space of these models. That gives the advantage of having a backbone already trained on the training data and the advantage that the 12 different classification heads can train independently. This has the advantage that classification heads can train a different number of epochs since they are not forced to learn at the same speed. In addition, different classification heads can be trained in this setting in different hyperparameter settings, which can also be a way to improve performance.

## 7.2 Usage of an additional Metric to evaluate the Robustness to Distortions

The metrics utilized in this thesis primarily assess the overall performance of the models, which is suitable for evaluating the characteristics of the clean dataset. However, it is valuable to examine the model performance on the corrupted dataset in relation to their performance on the clean dataset. For example, if two models (A and B) achieve the same balanced accuracy, such as 75%, on the corrupted dataset, they may appear equally robust to distortions. However, when considering their respective performances on the clean dataset, 85% balanced accuracy for model A versus 75% for model B, it becomes evident that model B demonstrates greater robustness to distortions, as its performance is less affected by the introduction of corruption. This is essential for evaluating the model's performance, as the goal is usually to achieve minimal performance degradation under distribution shifts.

Hendrycks and Dietterich (2019) and Salvo et al. (2024) introduce two metrics that solve this problem by comparing the robustness of the model to the robustness of an AlexNet. Additionally, a metric is introduced to measure robustness by comparing the performance of corrupted data relative to clean data: The metric uses the balanced error, which is calculated by 1 - 1-balanced accuracy. The evaluation begins with the calculation of the clean, balanced error, $\mathrm{BE}_f^{\mathrm{clean}}$, for each model $f$ using the corresponding MedMNIST+ test set. Following this, the balanced error, $\mathrm{BE}_{f,s,c}$, is determined for each corruption $c \in C_d$ and severity level $s$ (ranging from 1 to 5). Here, $C_d$ represents the set of all corruptions associated with a dataset $d$ (e.g., $C_{\mathrm{blood}} = \{\mathrm{JPEG}, \ldots, \mathrm{bubble}\}$). The errors are then averaged across severity

levels and normalized using AlexNet errors to produce $\mathrm{BE}_{f,c}$. This process, defined in Equation 12, accounts for the various effects of corruption on classification performance. The final balanced error $\mathrm{BE}_f$ is calculated by averaging $\mathrm{BE}_{f,c}$ on all corruptions (Salvo et al., 2024).

$$\mathrm{BE}_{f,c} = \frac{\sum_{s=1}^{5} \mathrm{BE}_{f,s,c}}{\sum_{s=1}^{5} \mathrm{BE}_{\mathrm{AlexNet},s,c}} \tag{12}$$

To assess performance degradation under distribution shifts, the relative balanced error (rBE) is introduced. This metric evaluates robustness by quantifying the performance drop in relation to the clean test set. The calculation of $\mathrm{rBE}_{f,c}$ extends Equation 12 by subtracting the clean performance:

$$\mathrm{rBE}_{f,c} = \frac{\sum_{s=1}^{5} (\mathrm{BE}_{f,s,c} - \mathrm{BE}_{f}^{\mathrm{clean}})}{\sum_{s=1}^{5} (\mathrm{BE}_{\mathrm{AlexNet},s,c} - \mathrm{BE}_{\mathrm{AlexNet}}^{\mathrm{clean}})} \tag{13}$$

The general value $\mathrm{rBE}_f$ is obtained by averaging $\mathrm{rBE}_{f,c}$ across all corruptions. The metrics are averaged across labels for multi-label tasks, such as ChestMNIST, to ensure consistency (Salvo et al., 2024).

These metrics would be suitable for evaluating the robustness of the models, but they were not in this thesis´s time budget since an additional AlexNet model needs to be trained and evaluated with the same training methodologies.

# 8 Conclusion

This thesis introduced a novel benchmark for MedMNIST+, an extension to the MedMNIST dataset family, designed to facilitate standardized and efficient evaluation of medical image classification tasks. By incorporating diverse imaging modalities, expanded task definitions, and state-of-the-art baseline results across multiple models and methods, the MedMNIST+ benchmark provides a comprehensive platform for assessing model performance in medical scenarios. Through this benchmark, we aim to accelerate the development and comparison of machine learning methods in the medical imaging domain, fostering reproducible research and driving progress toward practical clinical applications.

Evaluating various machine learning approaches on foundation models established a baseline for the challenge. The pre-trained models achieved the highest overall performance, while the mm-PT and mm-PT aug training methodologies fell short of surpassing them. Among the models, the Dino backbone delivered the best results, benefiting from pre-training on various tasks, whereas Prov and UNI, pre-trained on specific medical domains, lagged behind.

Additionally, a novel benchmark for MedMNIST-C was introduced to evaluate model robustness on corrupted data and distorted medical images, extending the MedMNIST dataset family. MedMNIST-C facilitates a standardized and efficient assessment of model performance under challenging conditions by simulating real-world degradation through diverse corruption types and severity levels. This benchmark aims to advance the development of robust machine learning methods for medical imaging, fostering reproducible research and improving reliability in practical clinical scenarios.

A baseline for this challenge was established by evaluating the machine learning approaches trained on MedMNIST+. The mm-PT training methodology, which trains the model end-to-end, demonstrated the highest robustness to distortions, outperforming the pre-trained models when using the Prov backbone. This highlights that a well-trained backbone, exposed to the data during training, exhibits greater resilience to distortions. The Dino and Dinov2 backbones demonstrated superior performance among the pre-trained models, benefiting from their pre-training across various tasks. In contrast, Prov and UNI, pre-trained on specific medical domains, showed significantly lower robustness to corrupted inputs.

In conclusion, the introduction of the MedMNIST+ and MedMNIST-C benchmarks represents a significant step forward in evaluating medical image classification and robustness in the face of real-world data challenges. These benchmarks provide a comprehensive, standardized platform for assessing model performance across various tasks, fostering reproducible research and accelerating advancements in the field. The insights gained from the baseline evaluations further highlight the importance of pre-training and task diversity in enhancing model performance and robustness, paving the way for developing more reliable machine learning methods for practical clinical applications.

# A Appendix

## A.1 Every Plot for the Performance on MedMNIST+

This section of the appendix has every plot for every metric, every backbone, every training method and every resolution on MedMNIST+.

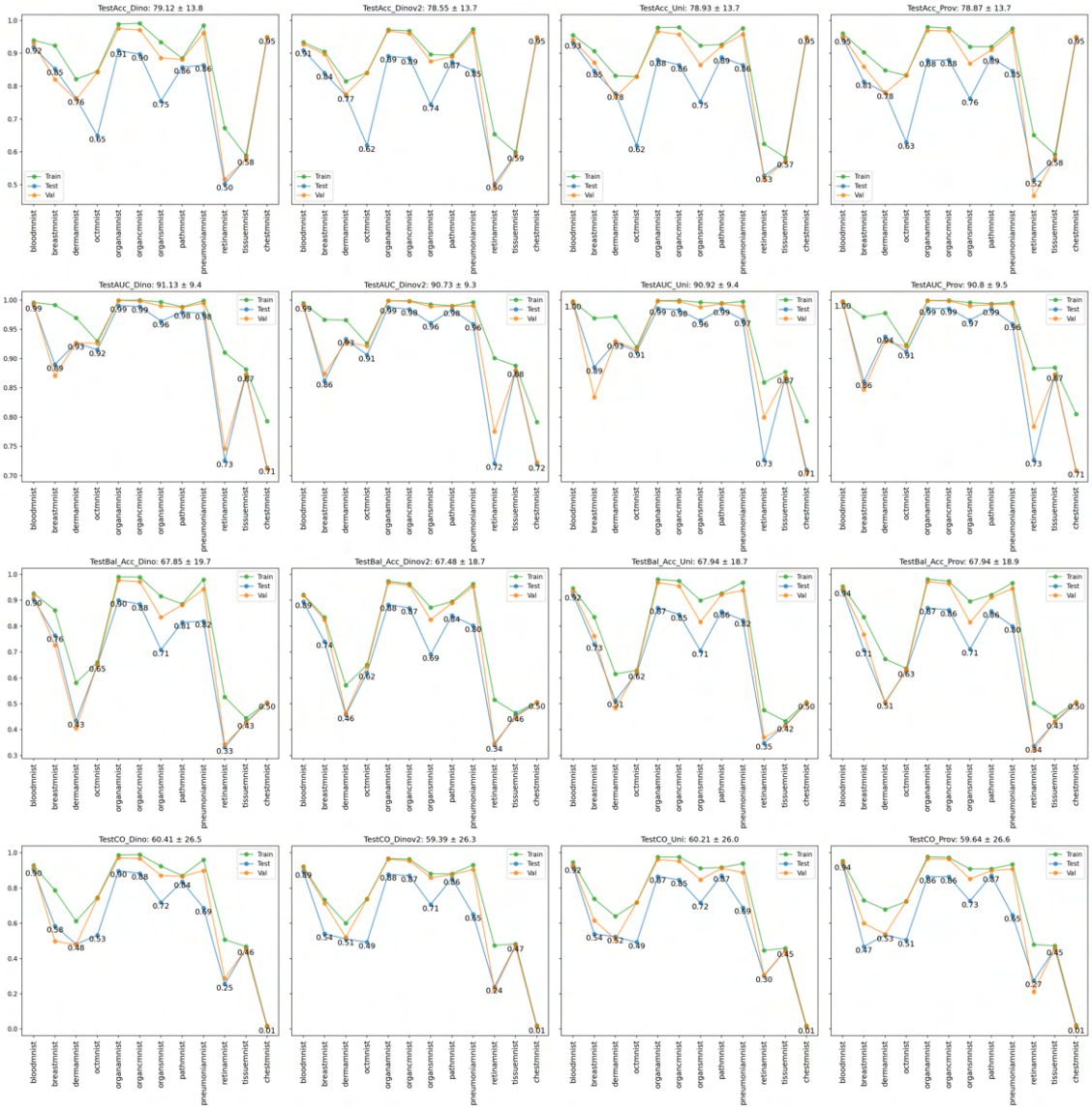


Figure 27: Every metric for 28×28 resolution with SVM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

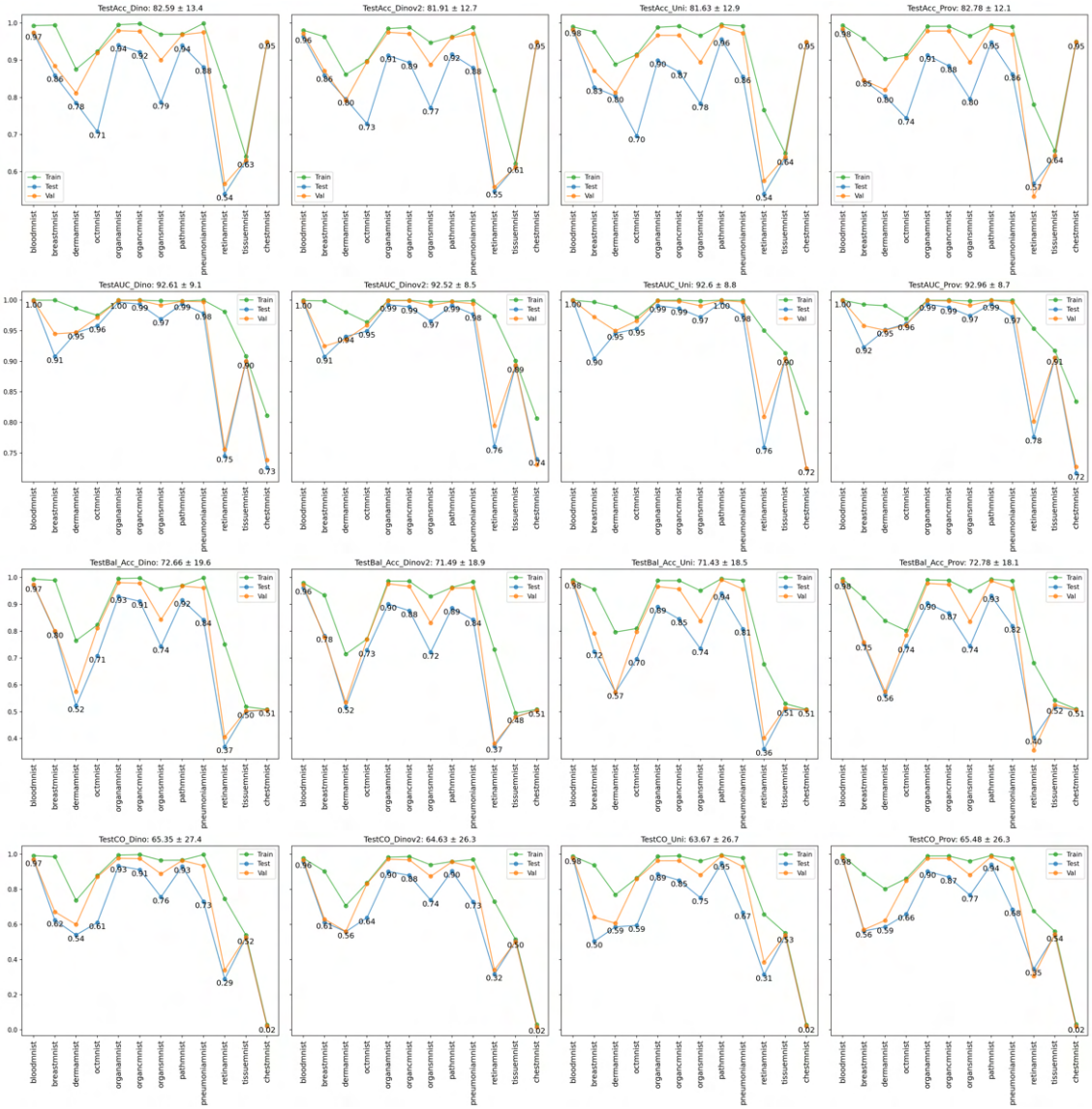


Figure 28: Every metric for 64×64 resolution with SVM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

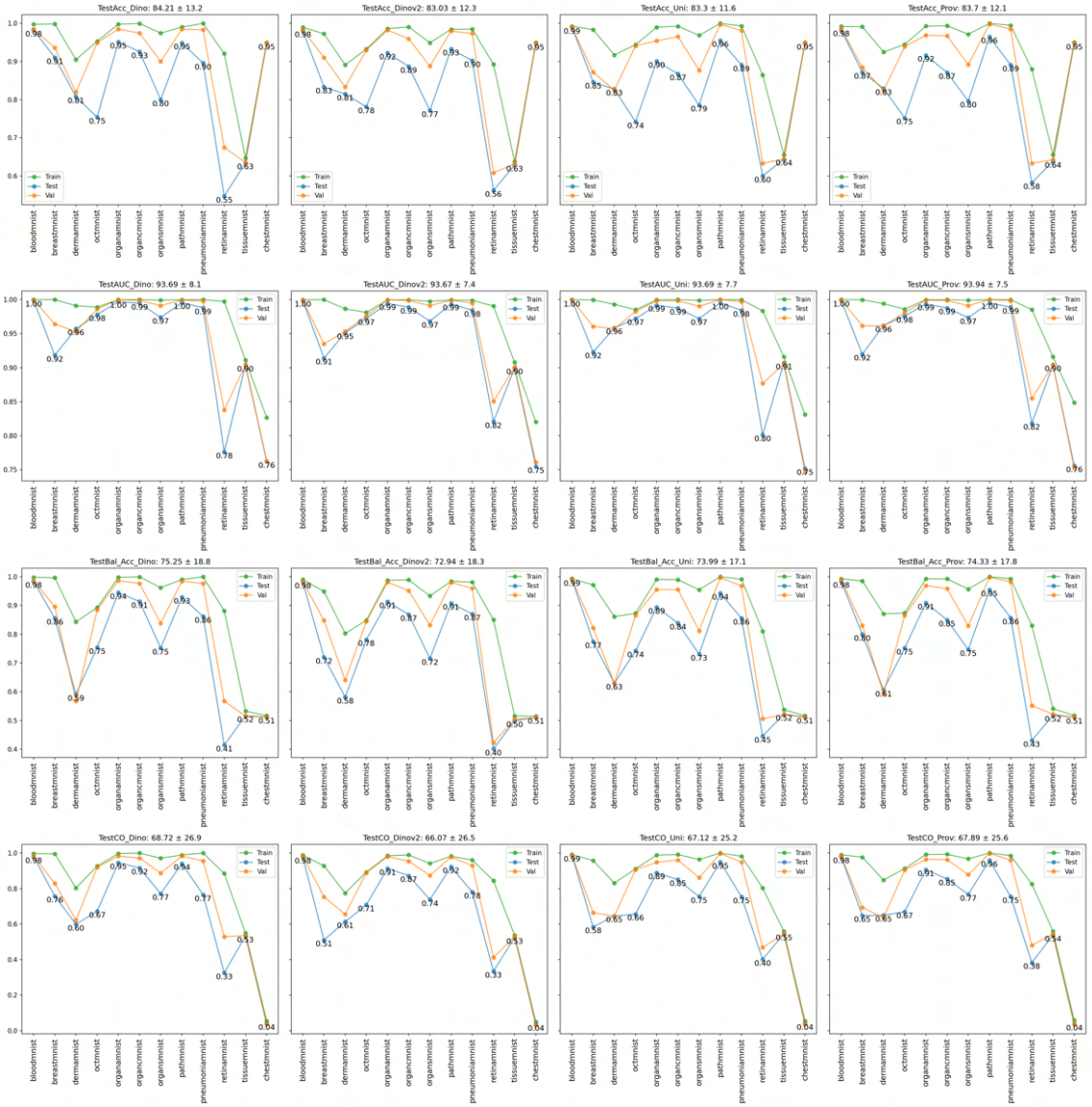


Figure 29: Every metric for 128×128 resolution with SVM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

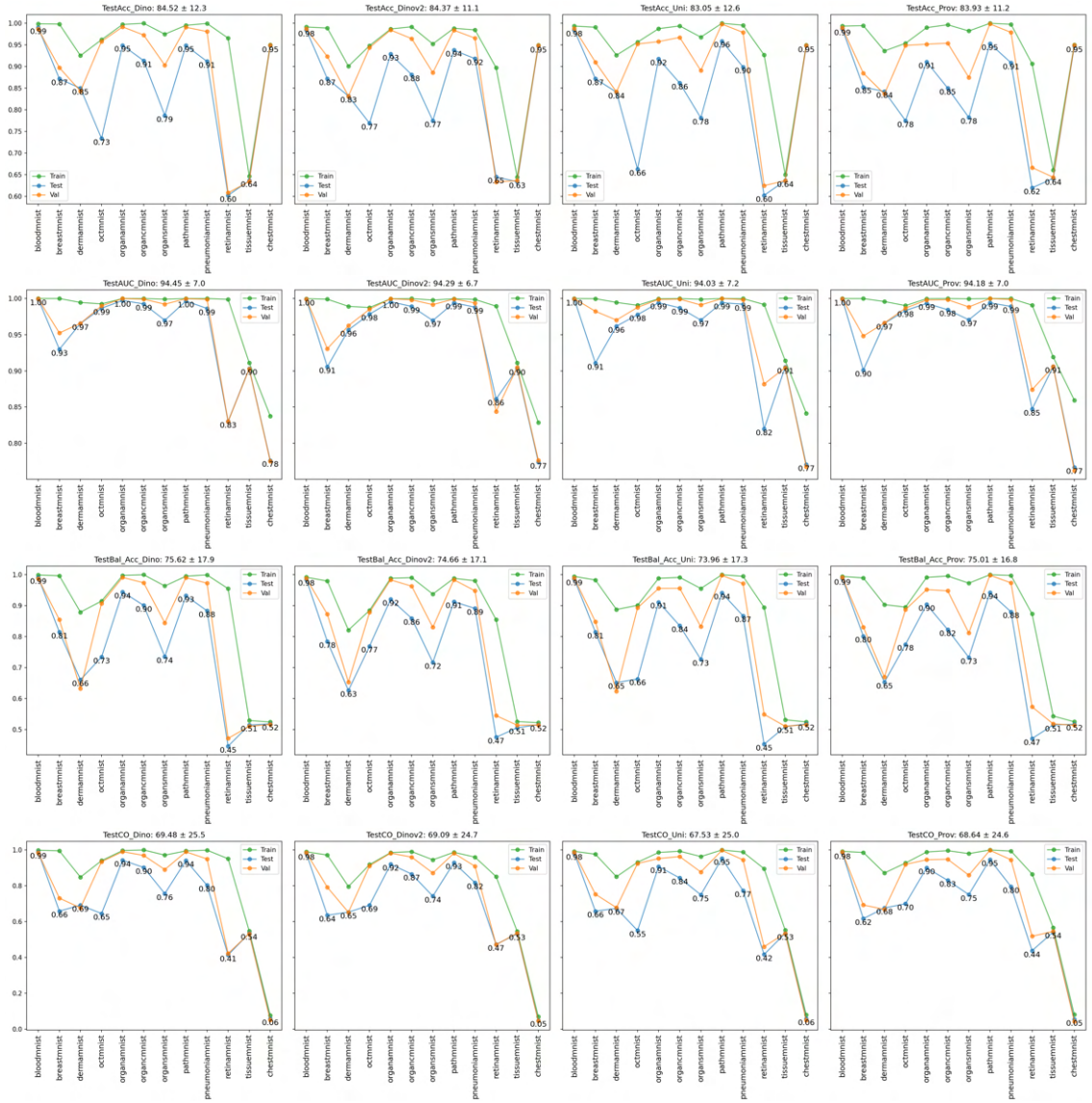


Figure 30: Every metric for 224×224 resolution with SVM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

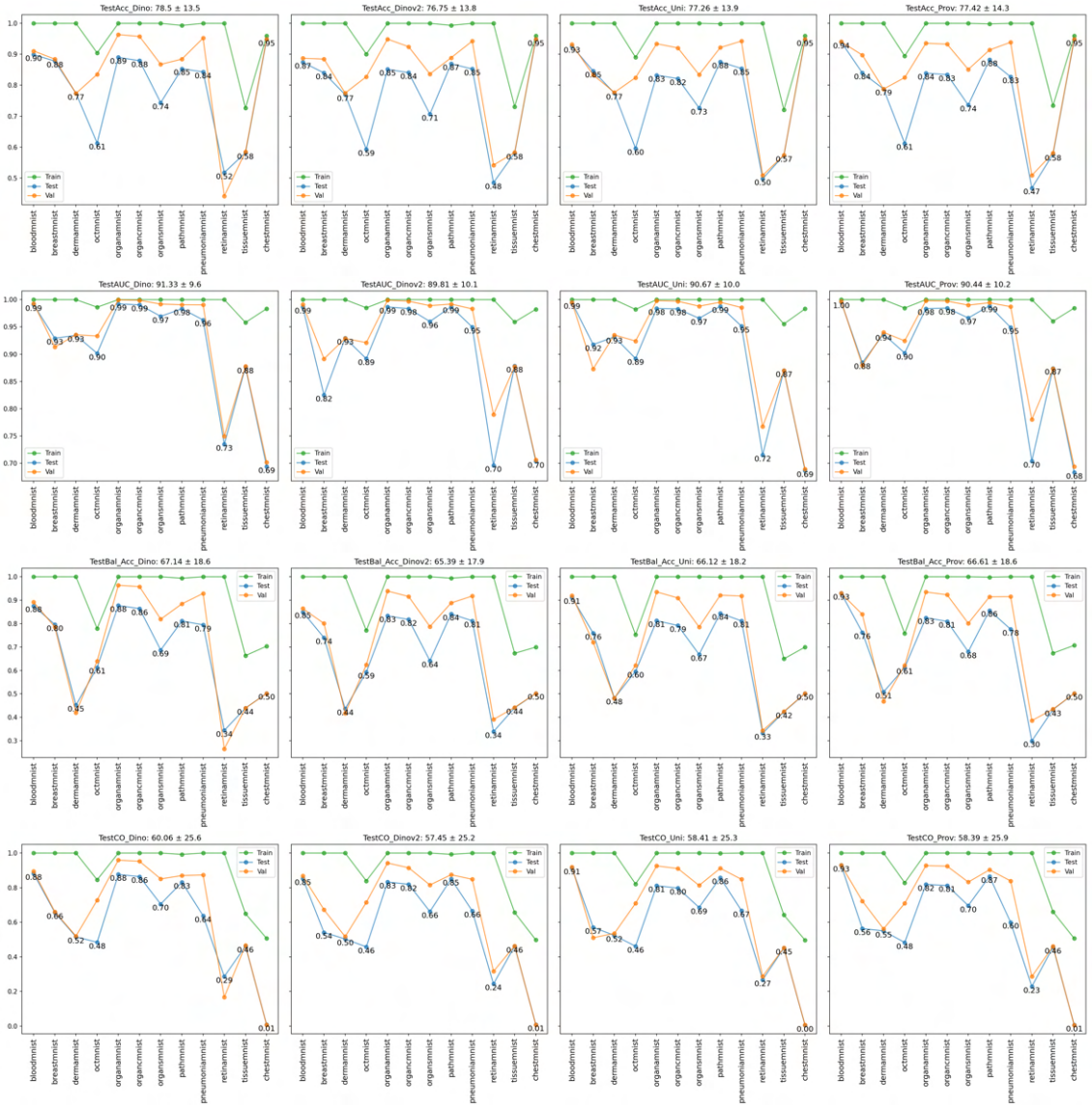


Figure 31: Every metric for 28×28 resolution with LightGBM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

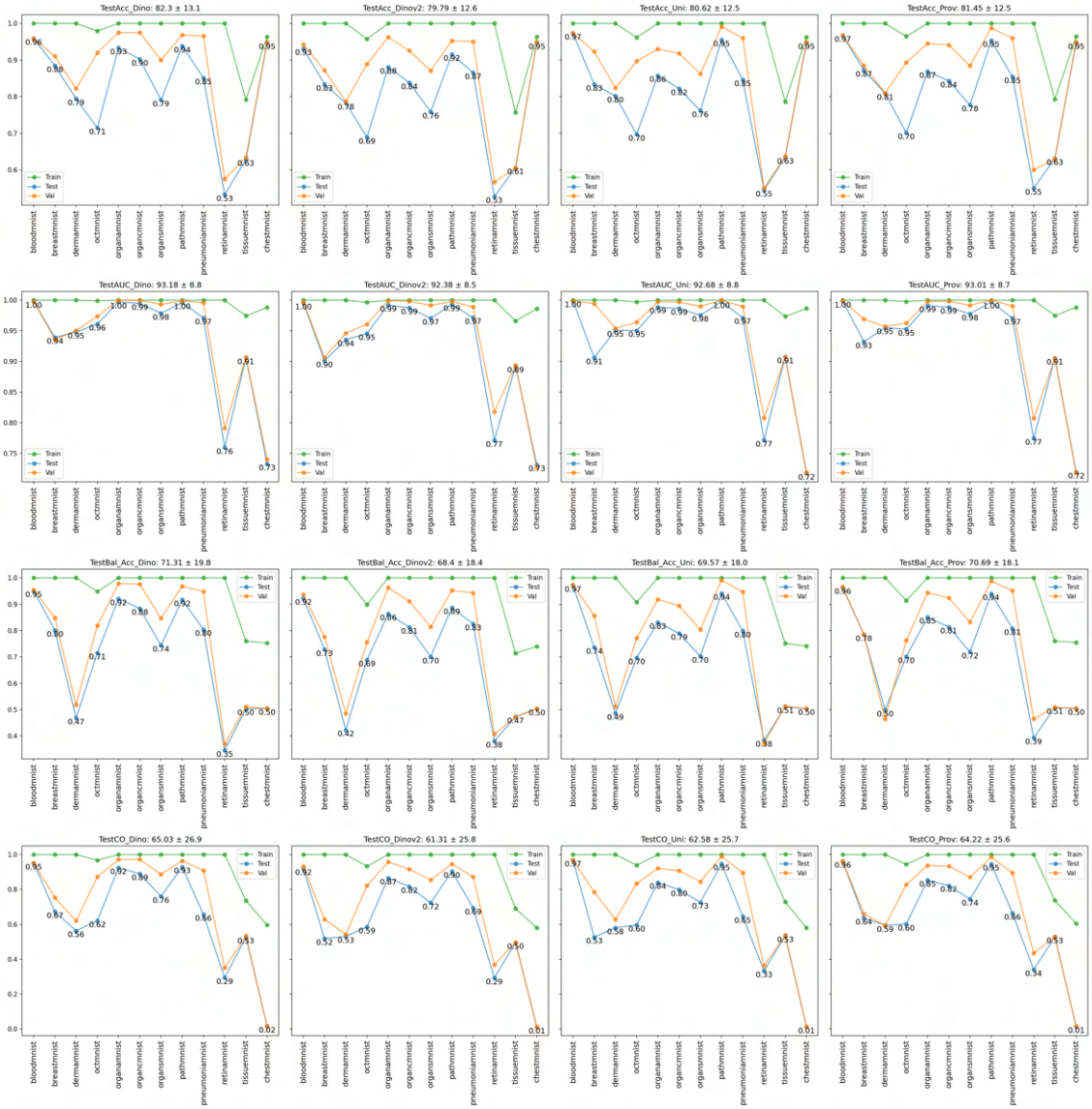


Figure 32: Every metric for 64×64 resolution with LightGBM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

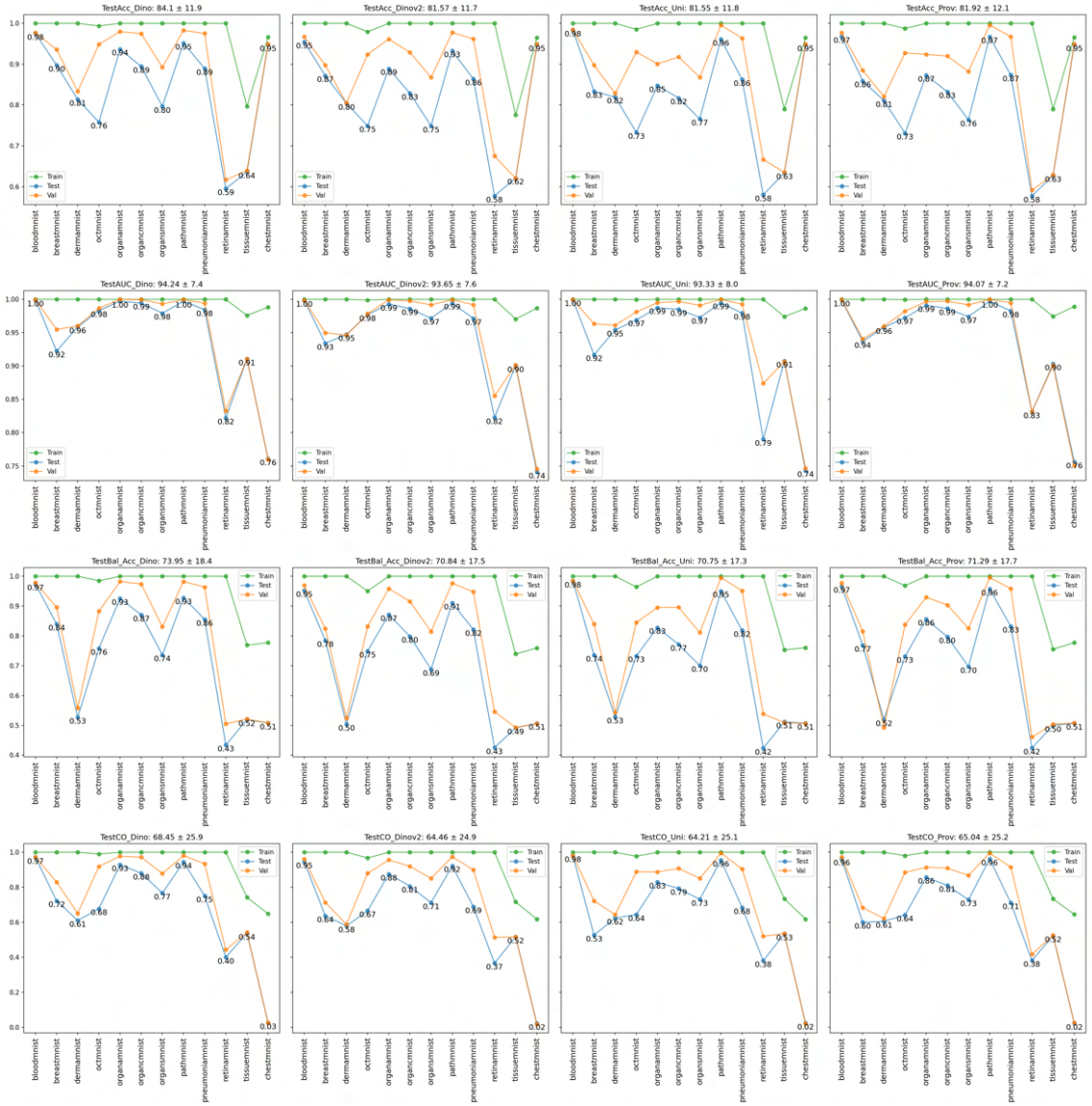


Figure 33: Every metric for 128×128 resolution with LightGBM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

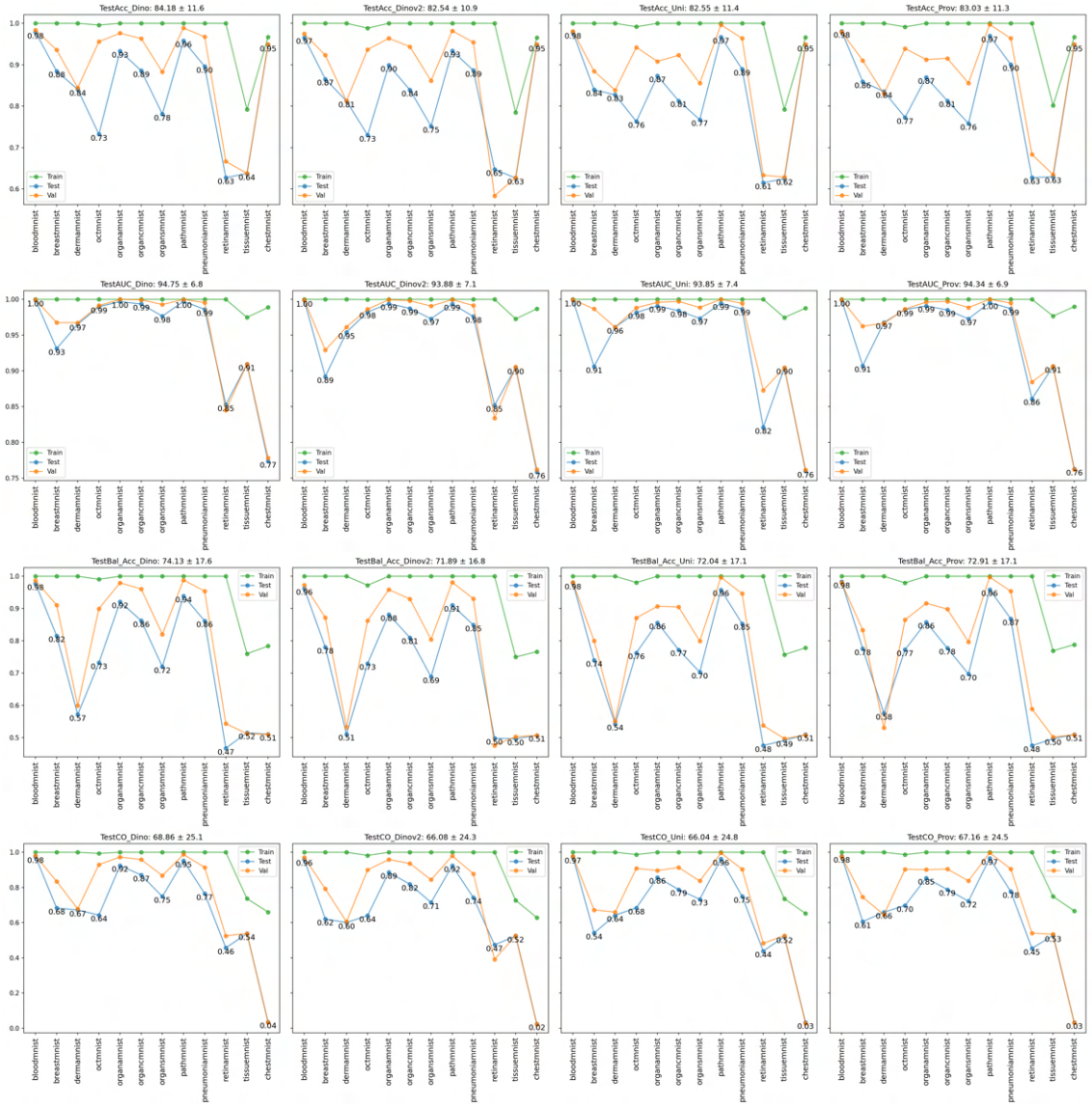


Figure 34: Every metric for 224×224 resolution with LightGBM as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

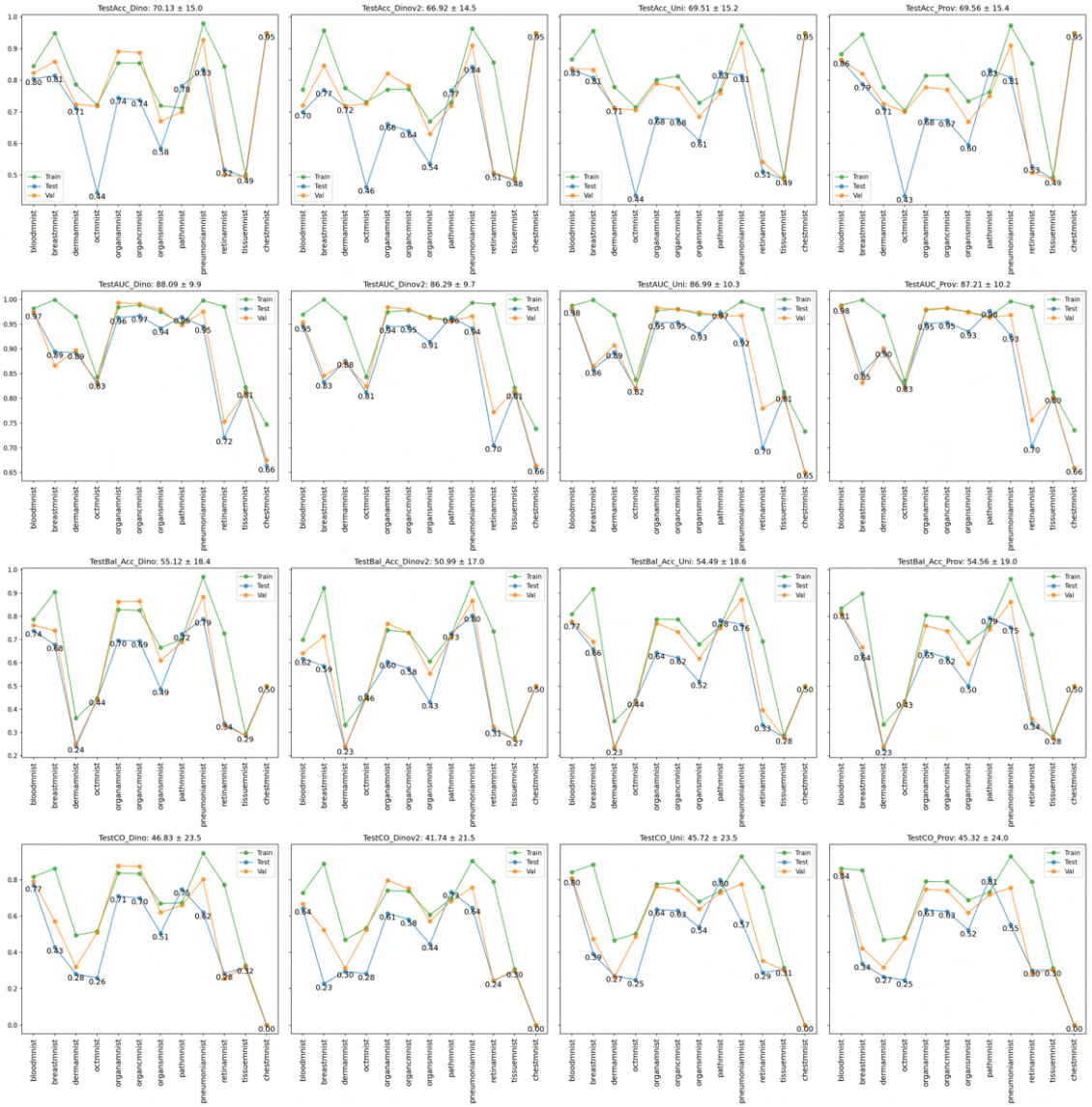


Figure 35: Every metric for 28×28 resolution with random forest as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

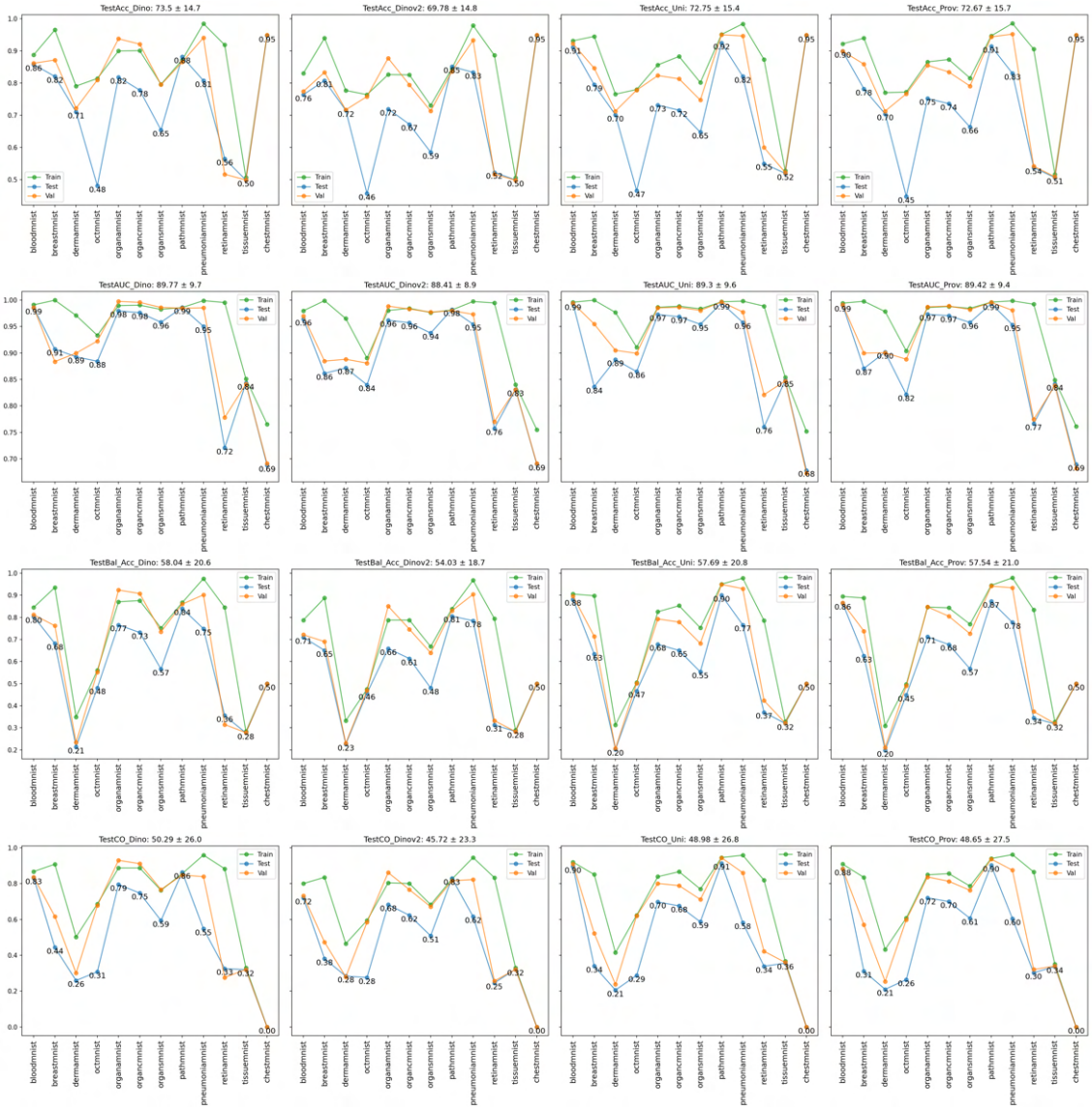


Figure 36: Every metric for 64×64 resolution with random forest as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

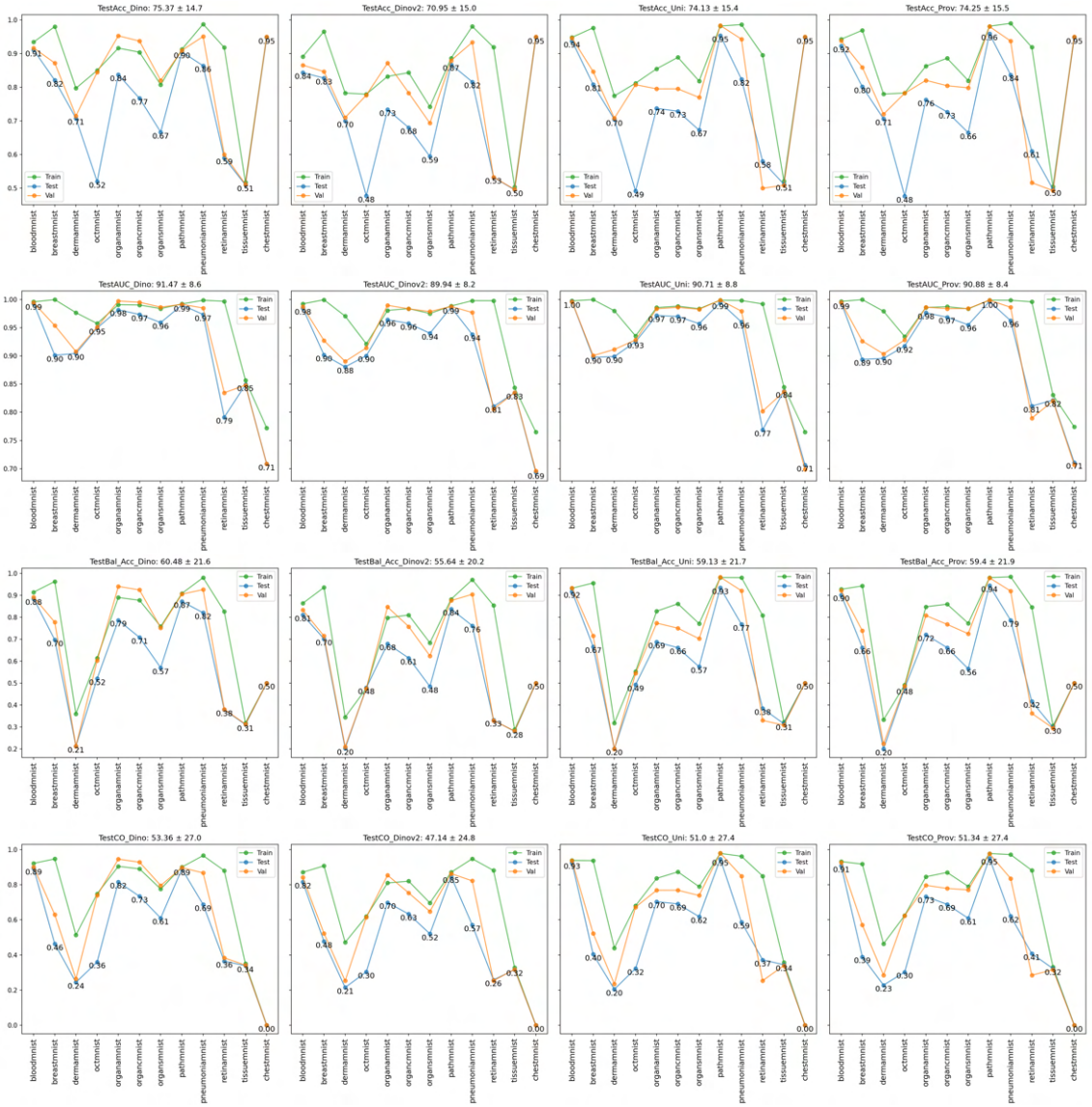


Figure 37: Every metric for 128×128 resolution with random forest as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

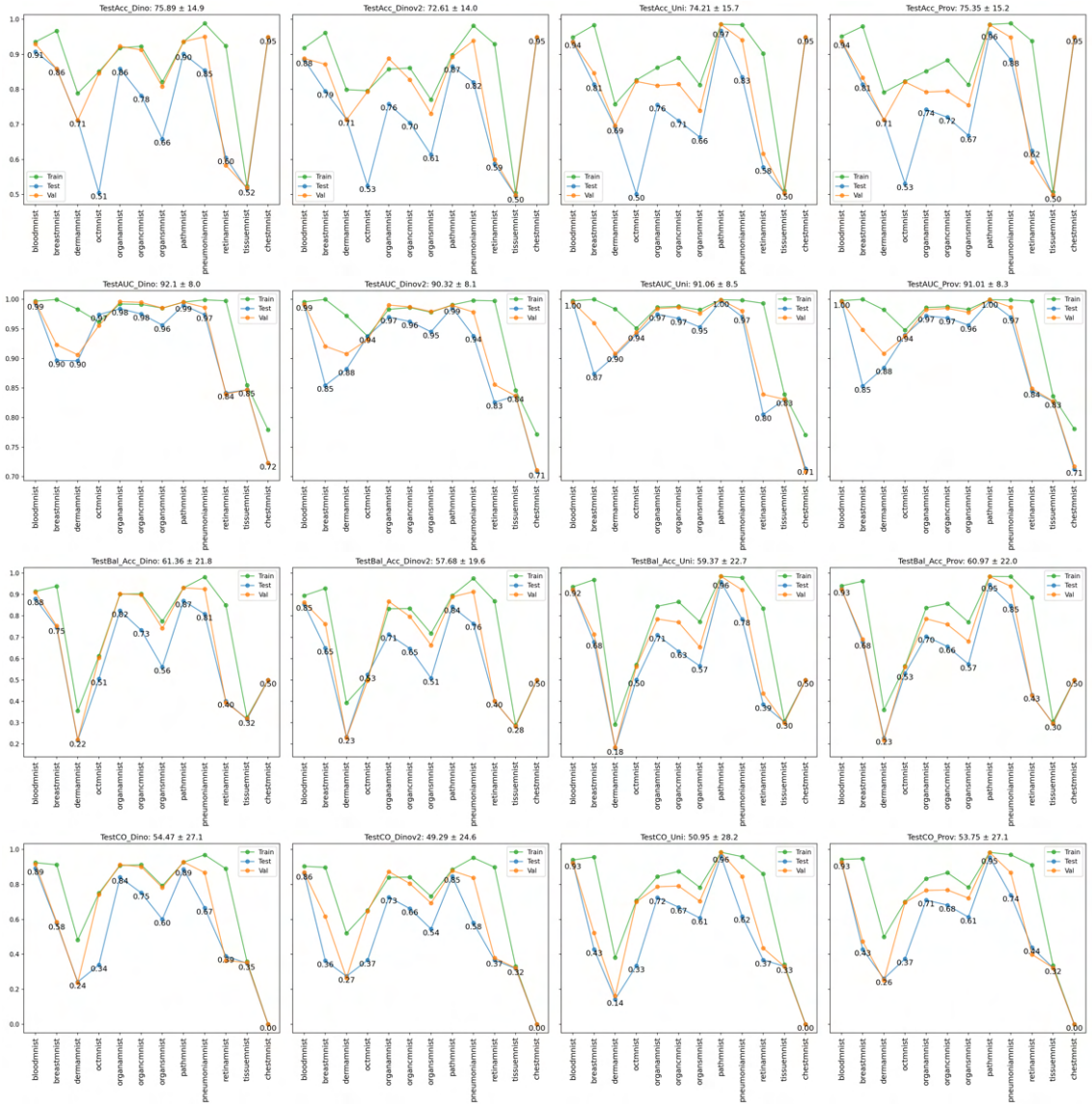


Figure 38: Every metric for 224×224 resolution with random forest as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

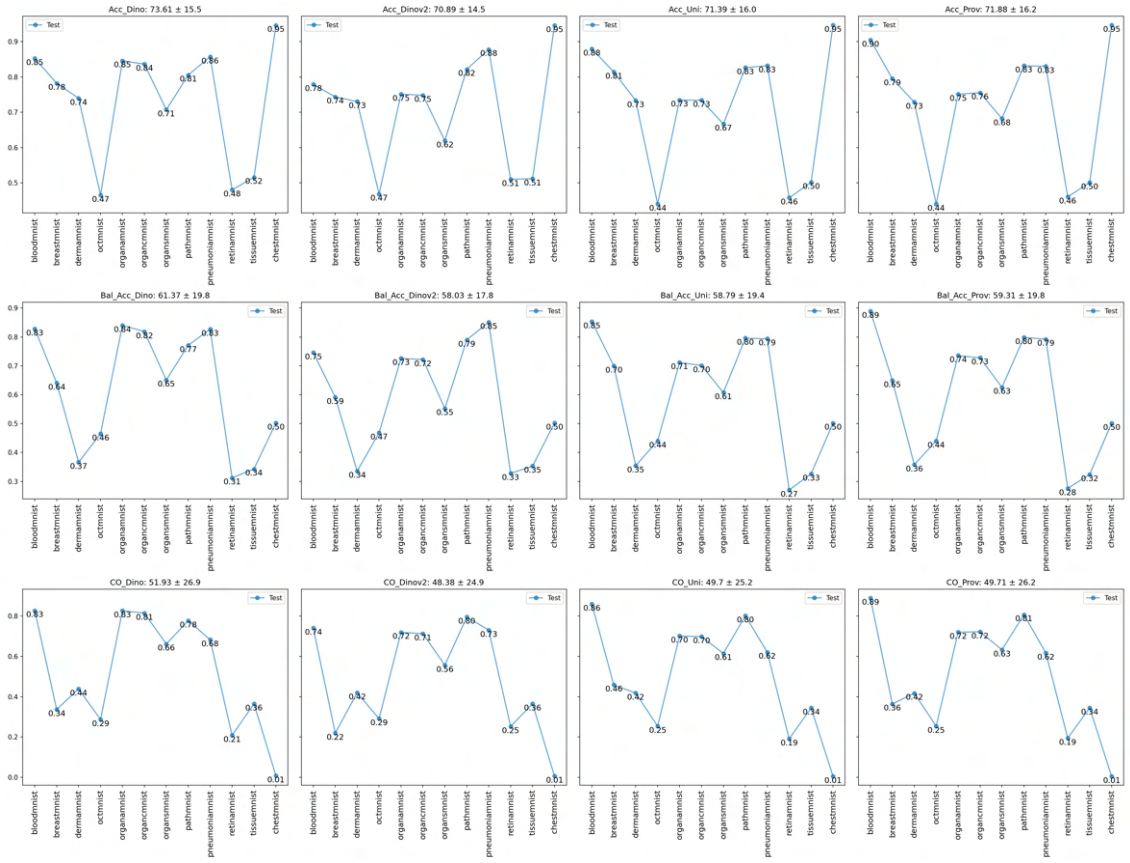


Figure 39: Every metric for 28×28 resolution with kNN as classification head on the pre-trained backbones.Test performance indicated in blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

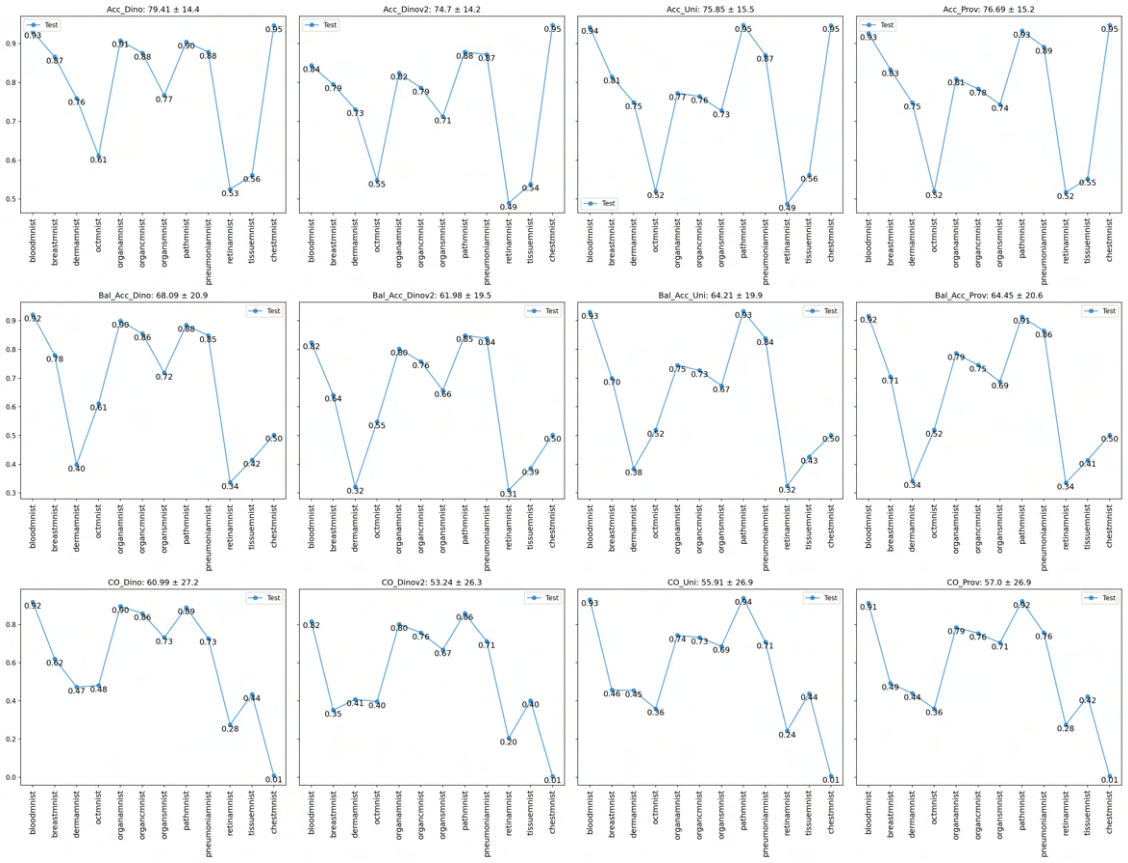


Figure 40: Every metric for 64×64 resolution with kNN as classification head on the pre-trained backbones. Test performance indicated in blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

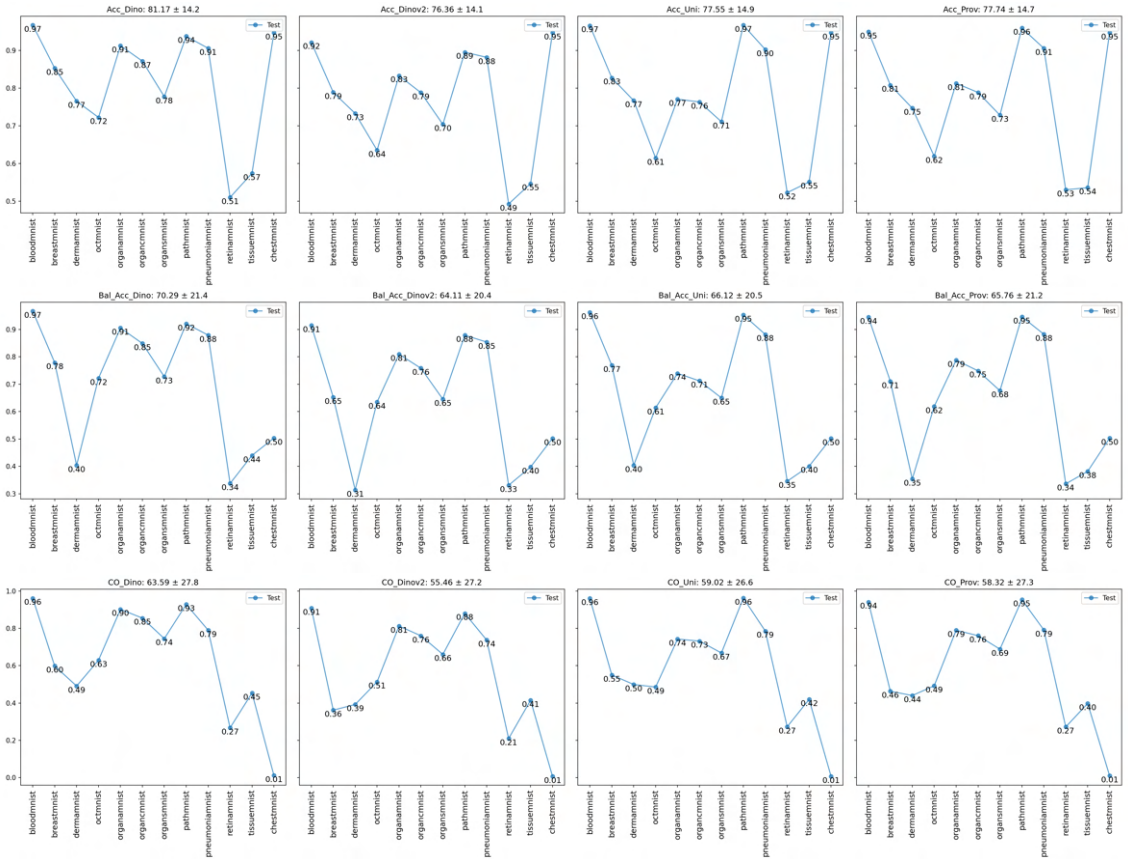


Figure 41: Every metric for 128×128 resolution with kNN as classification head on the pre-trained backbones. Test performance indicated in blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

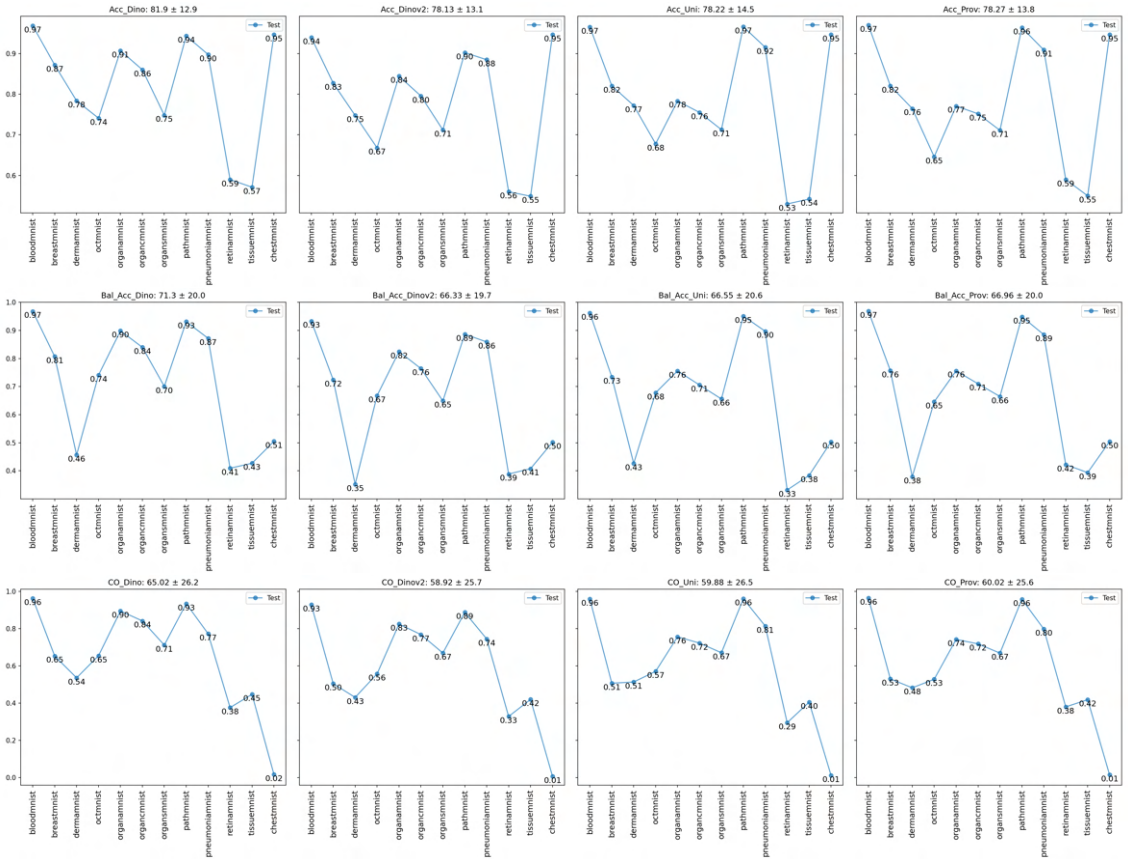


Figure 42: Every metric for 224×224 resolution with KNN as classification head on the pre-trained backbones. Test performance indicated in blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

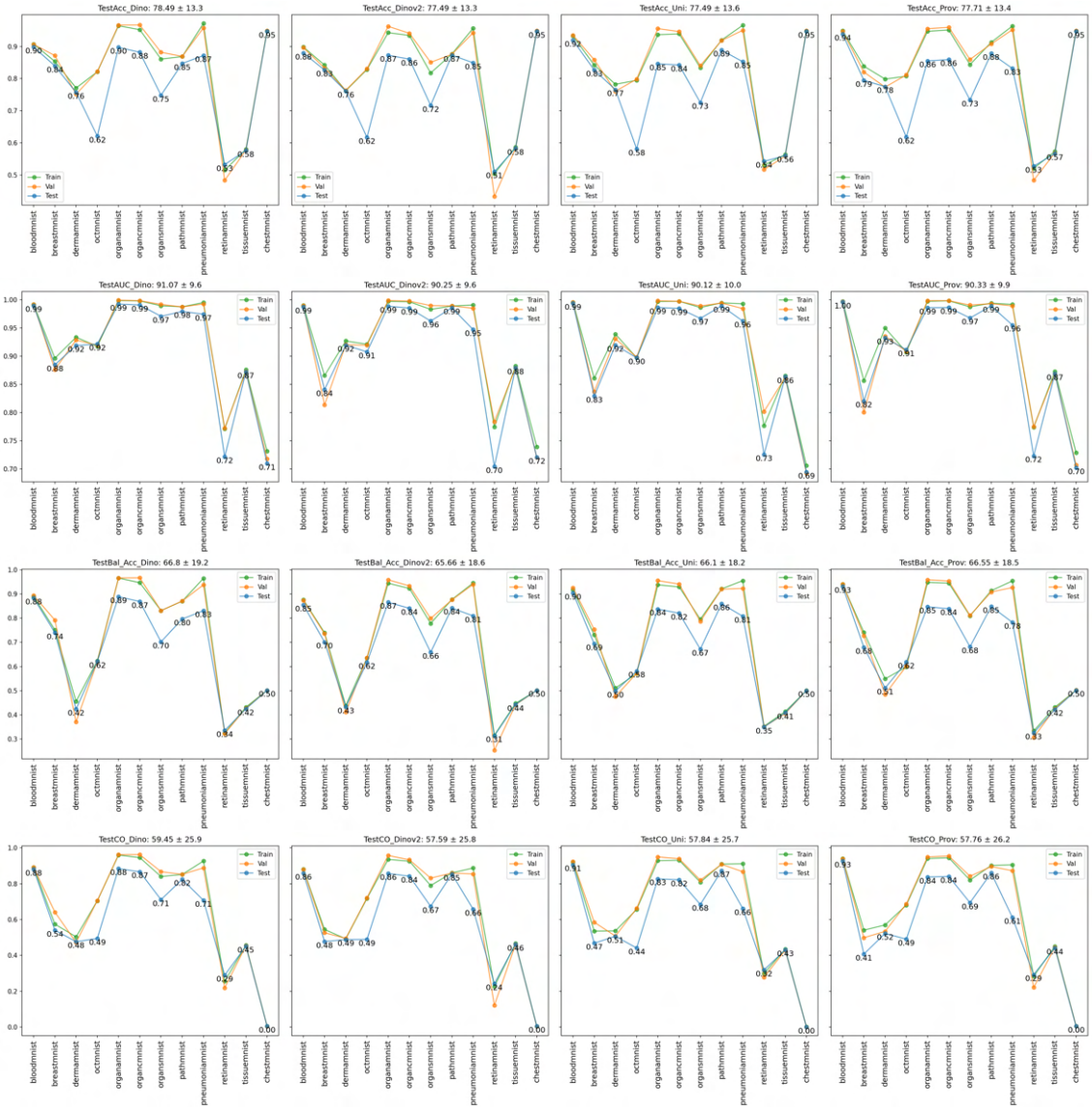


Figure 43: Every metric for 28×28 resolution with a a linear classifier as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

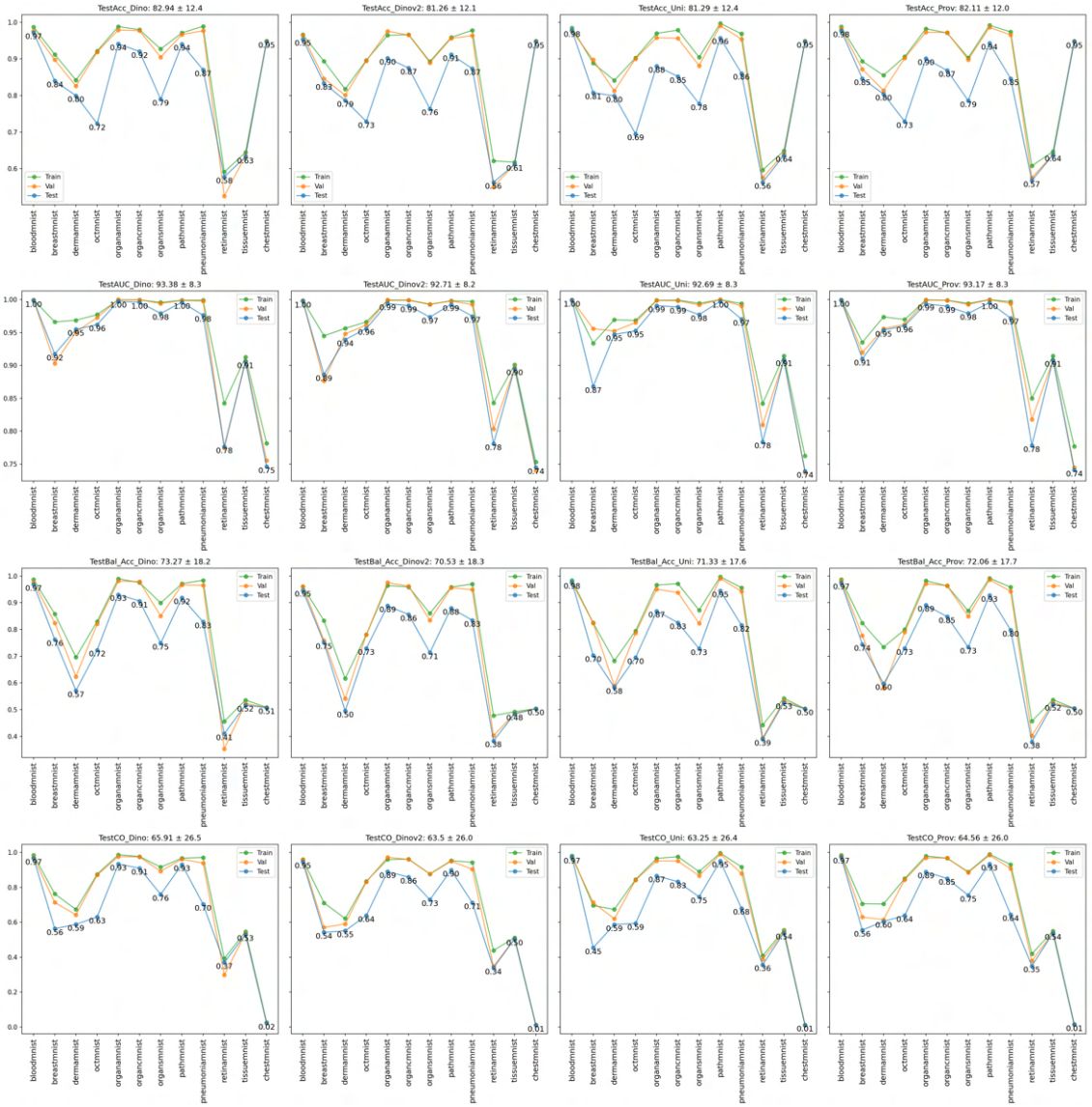


Figure 44: Every metric for 64×64 resolution with a linear classifier as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

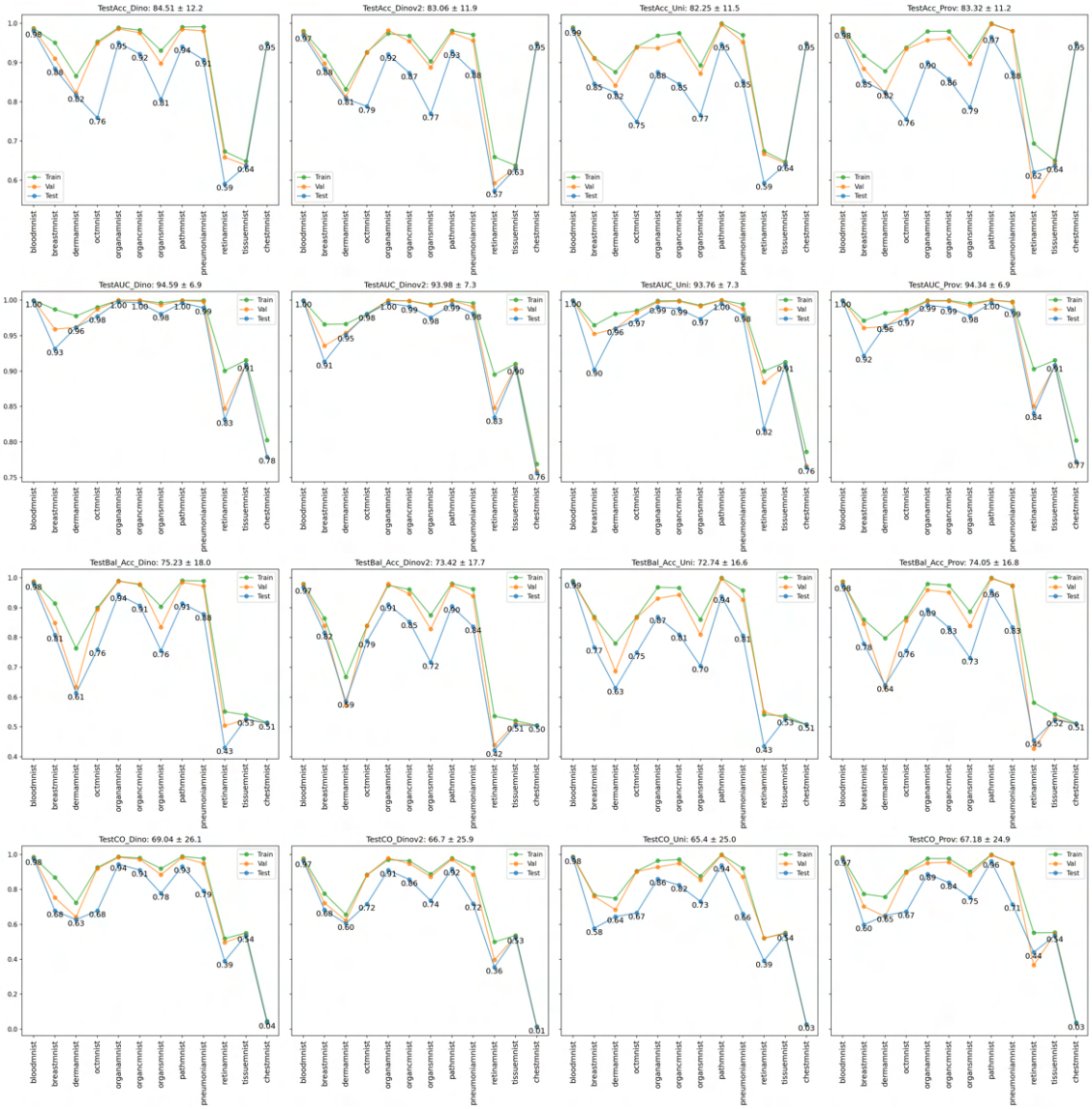


Figure 45: Every metric for 128×128 resolution with a linear classifier as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

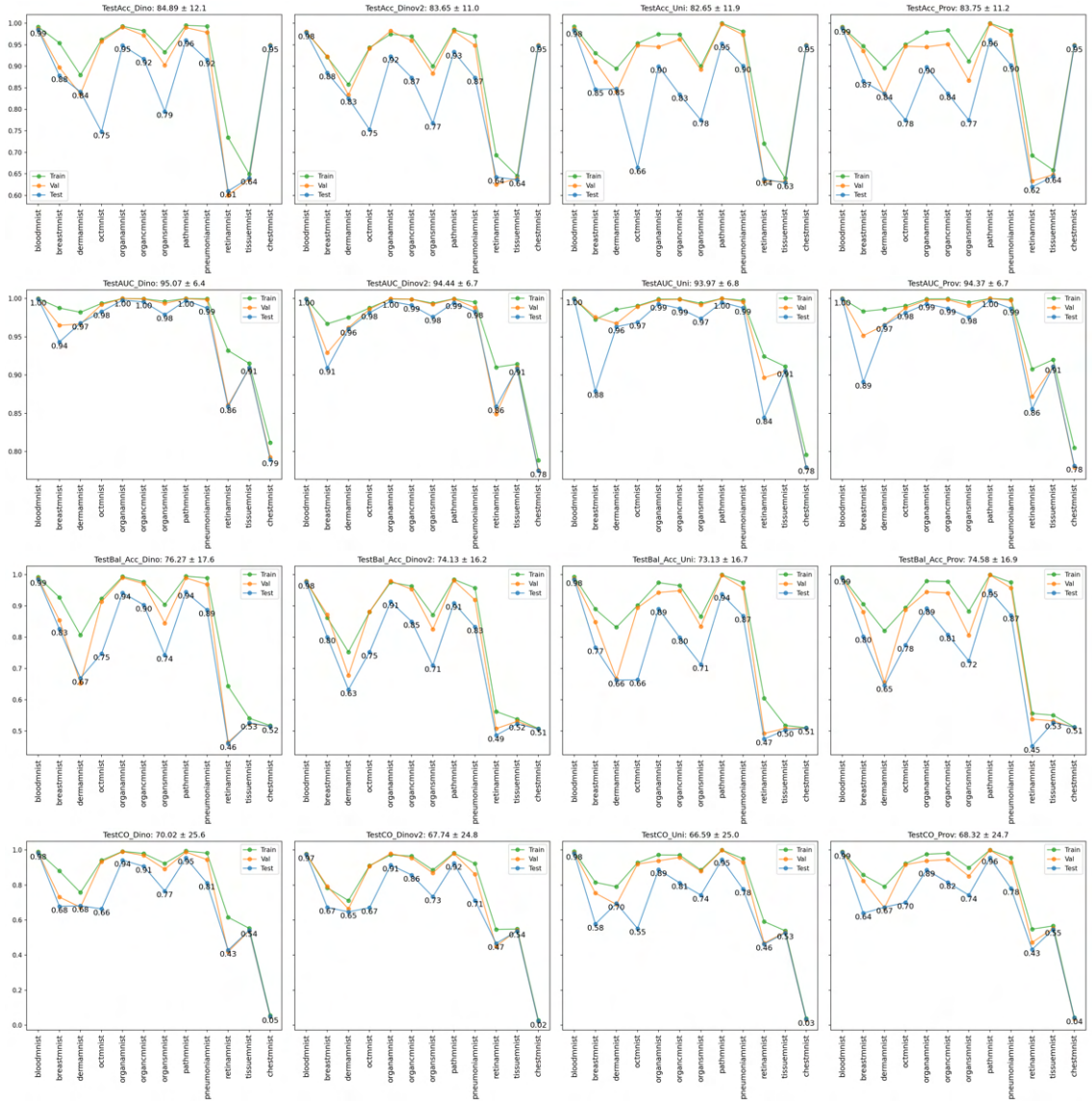


Figure 46: Every metric for 224×224 resolution with a linear classifier as classification head on the pre-trained backbones. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

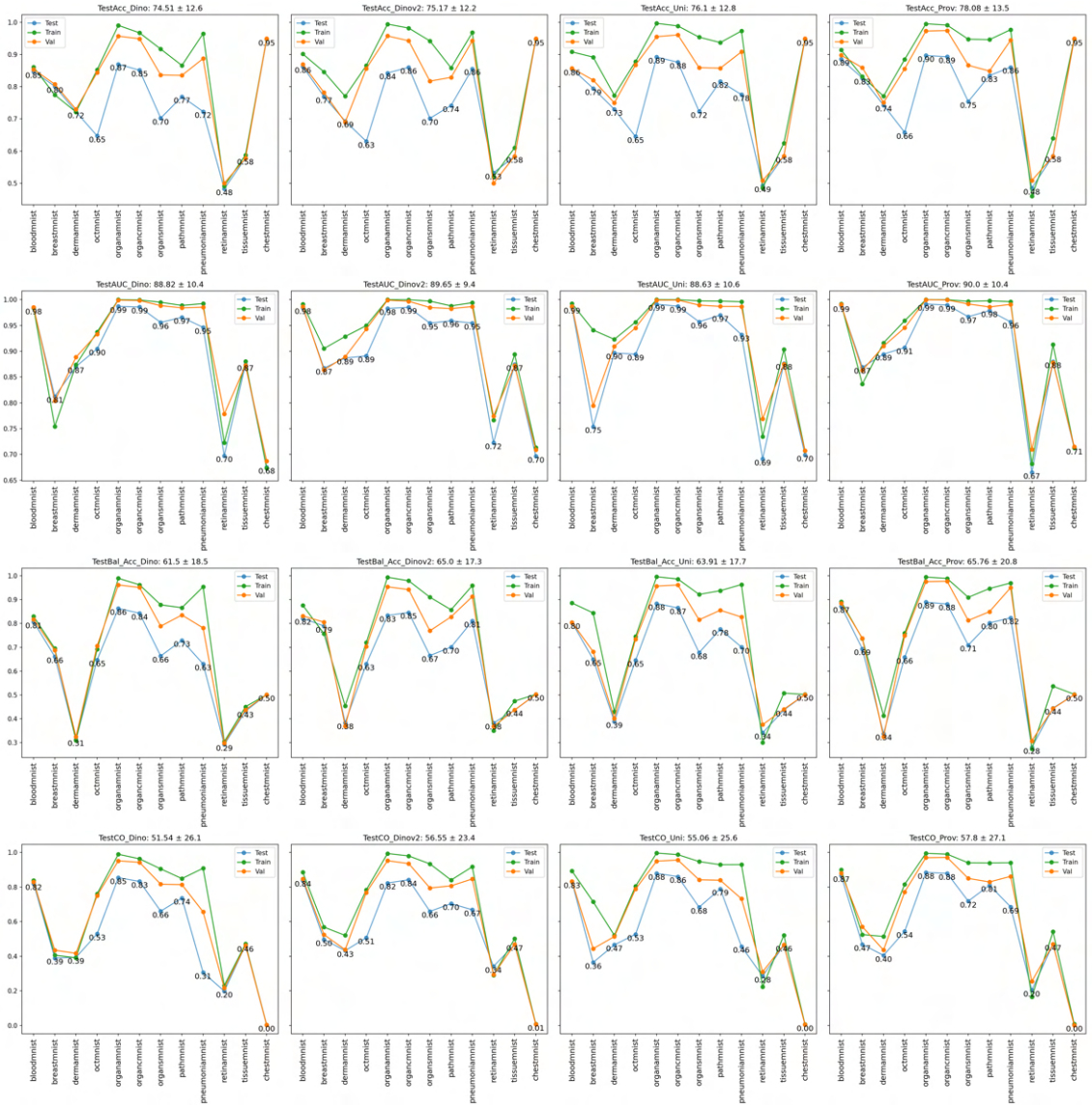


Figure 47: Every metric for 28×28 resolution with multi-domain multi-task pre-training for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

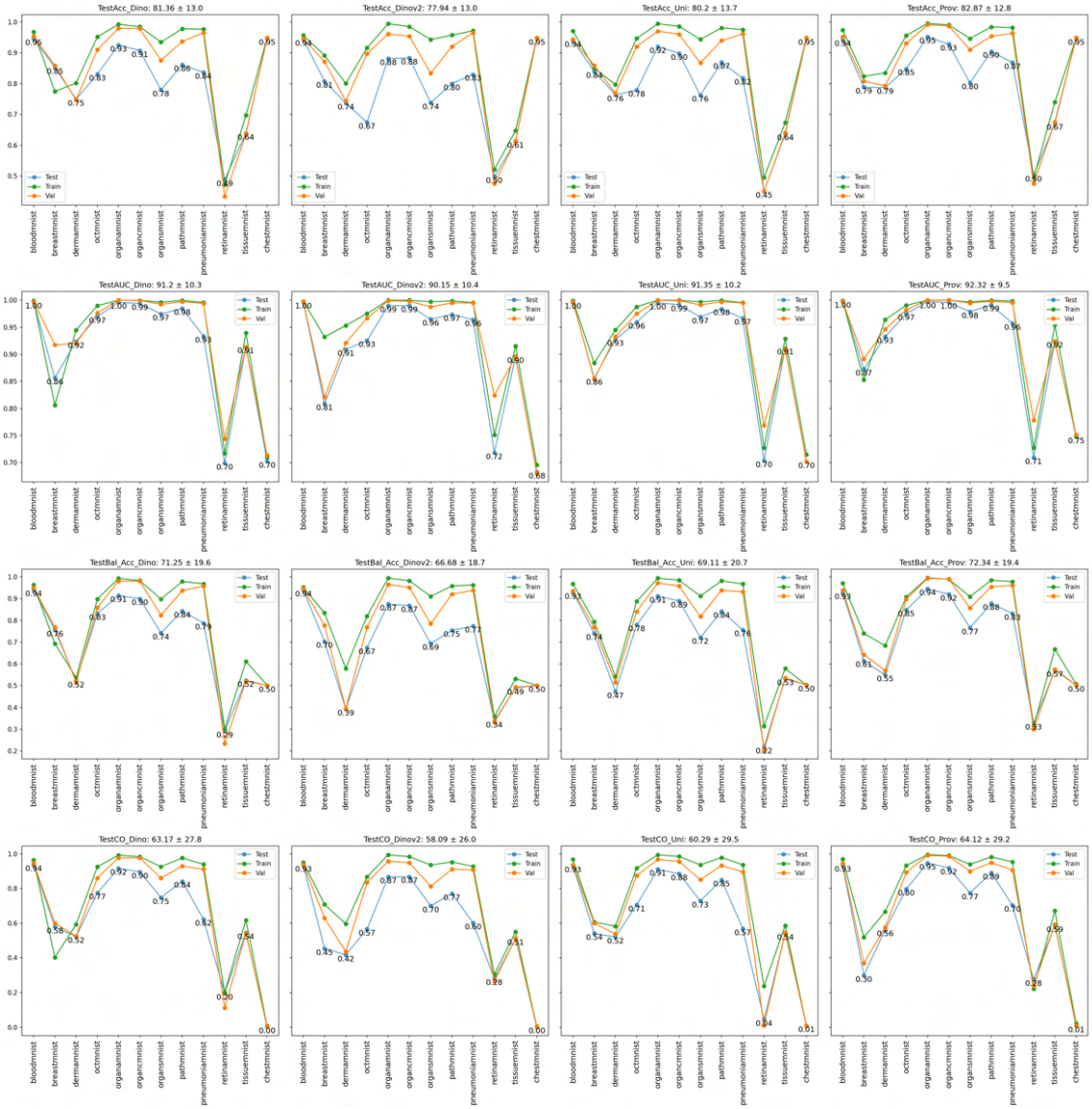


Figure 48: Every metric for 64×64 resolution with multi-domain multi-task pre-training for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

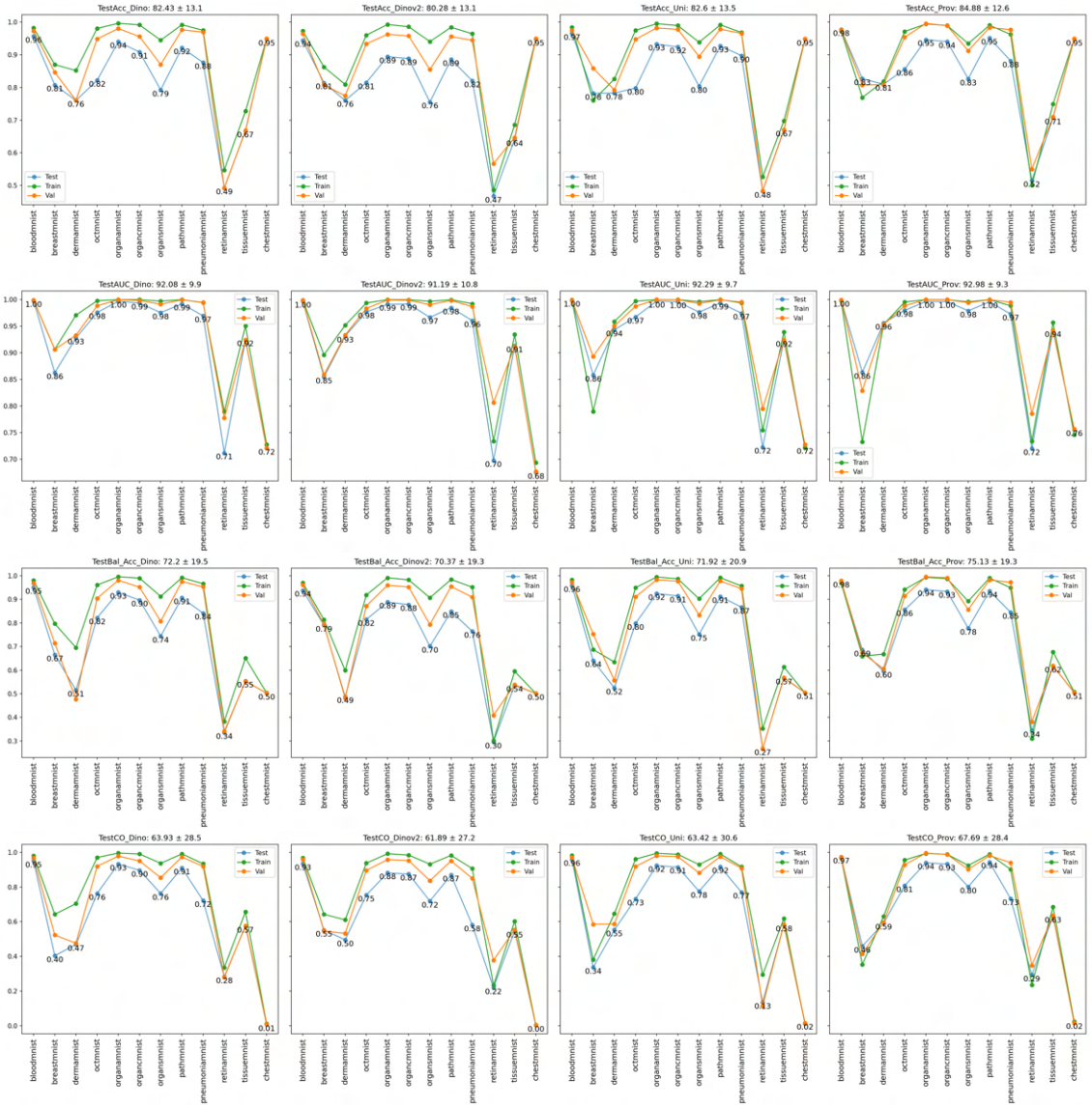


Figure 49: Every metric for 128×128 resolution with multi-domain multi-task pre-training for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

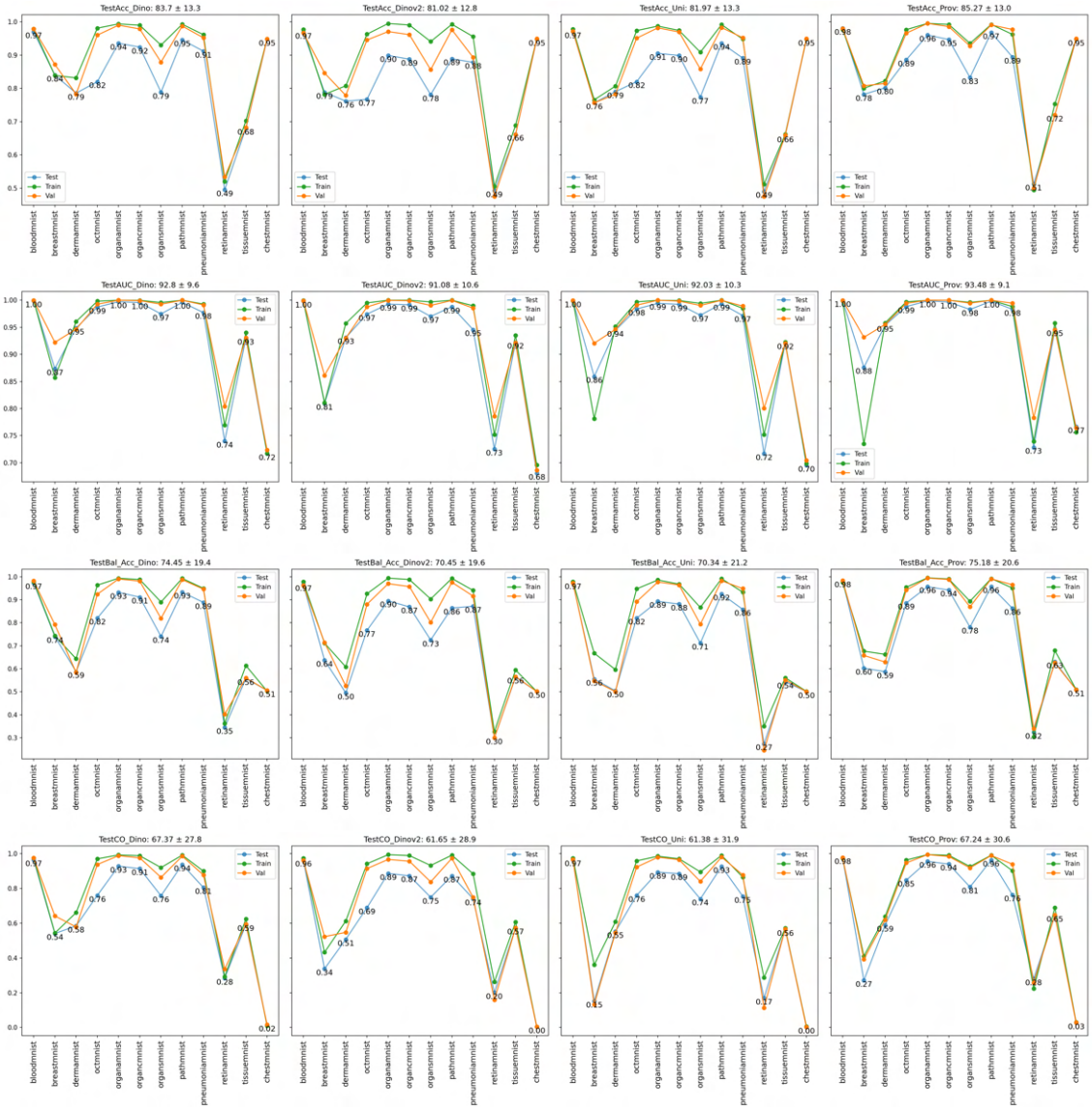


Figure 50: Every metric for 224×224 resolution with multi-domain multi-task pre-training for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

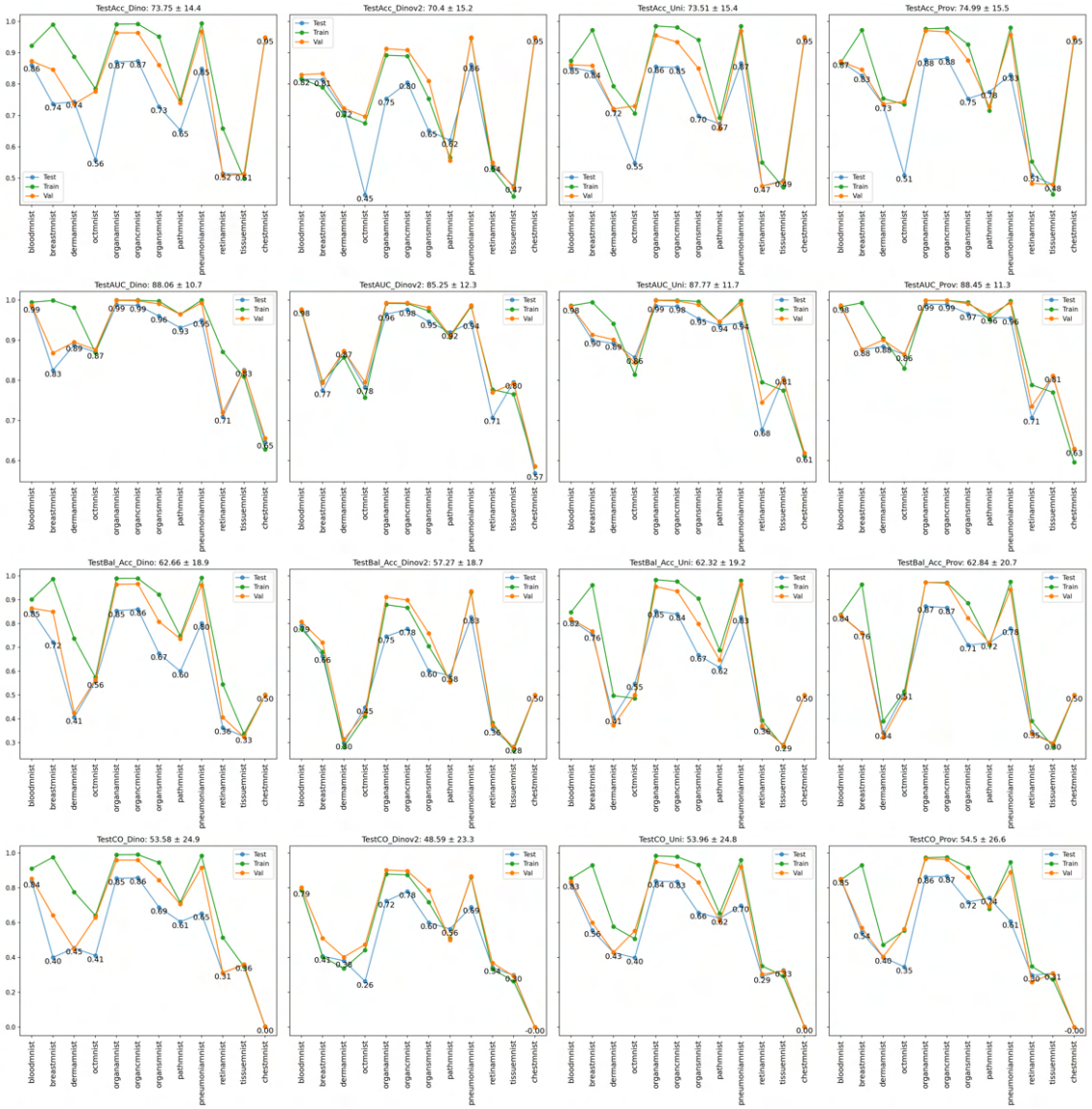


Figure 51: Every metric for 28×28 resolution with multi-domain multi-task pre-training for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

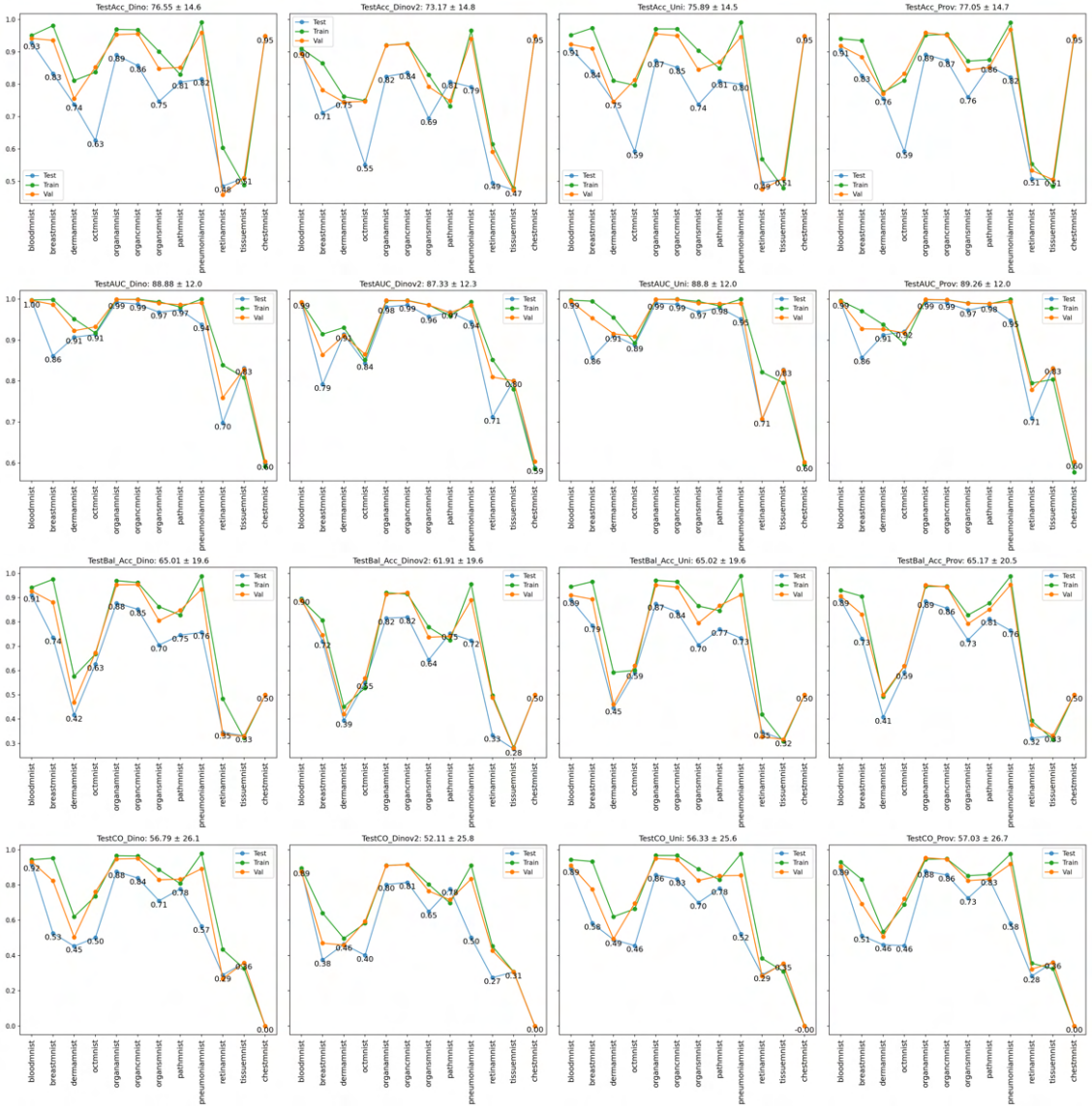


Figure 52: Every metric for 64×64 resolution with multi-domain multi-task pre-training with data-augmentation for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

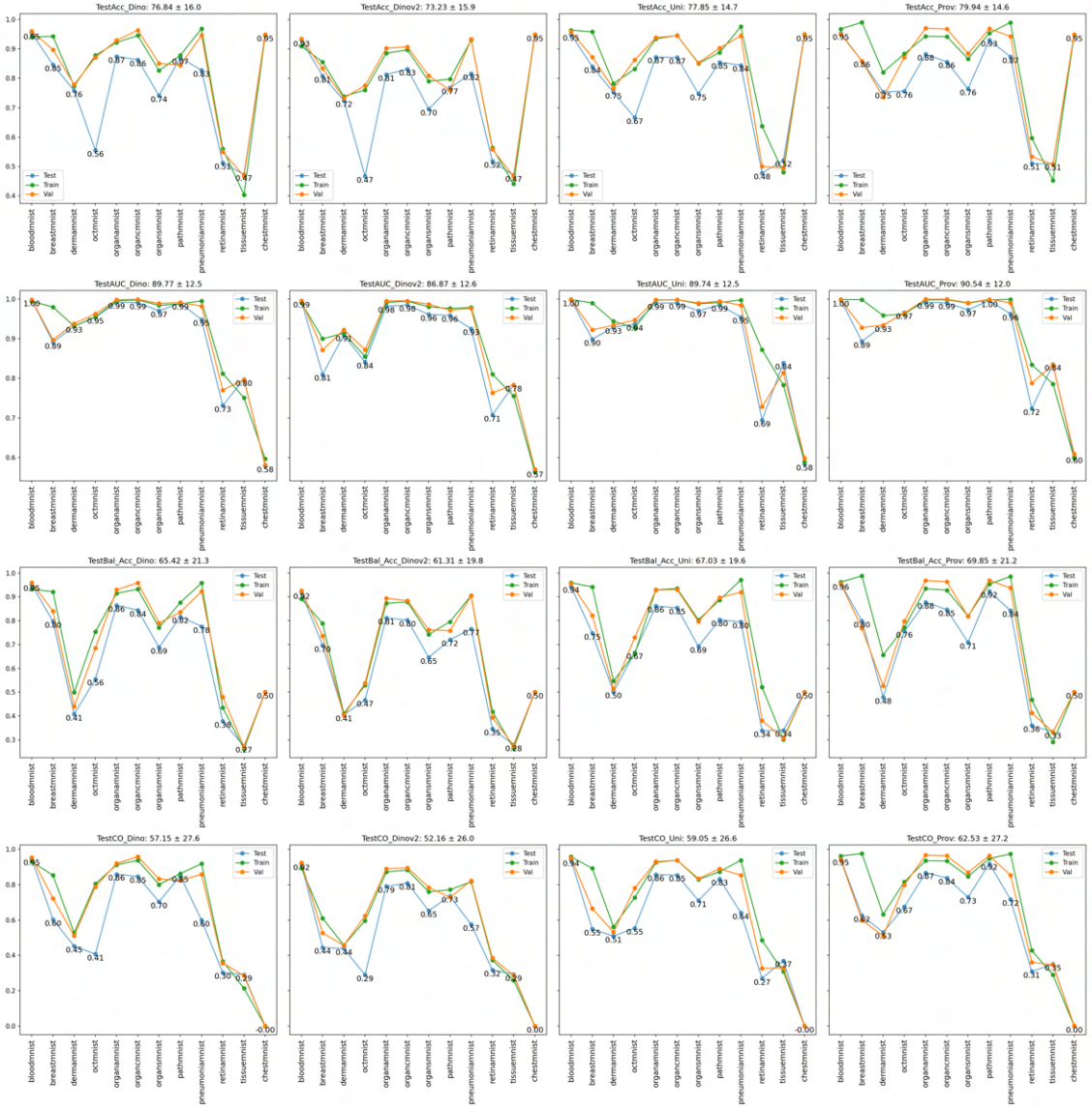


Figure 53: Every metric for 128×128 resolution with multi-domain multi-task pre-training with data-augmentation for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

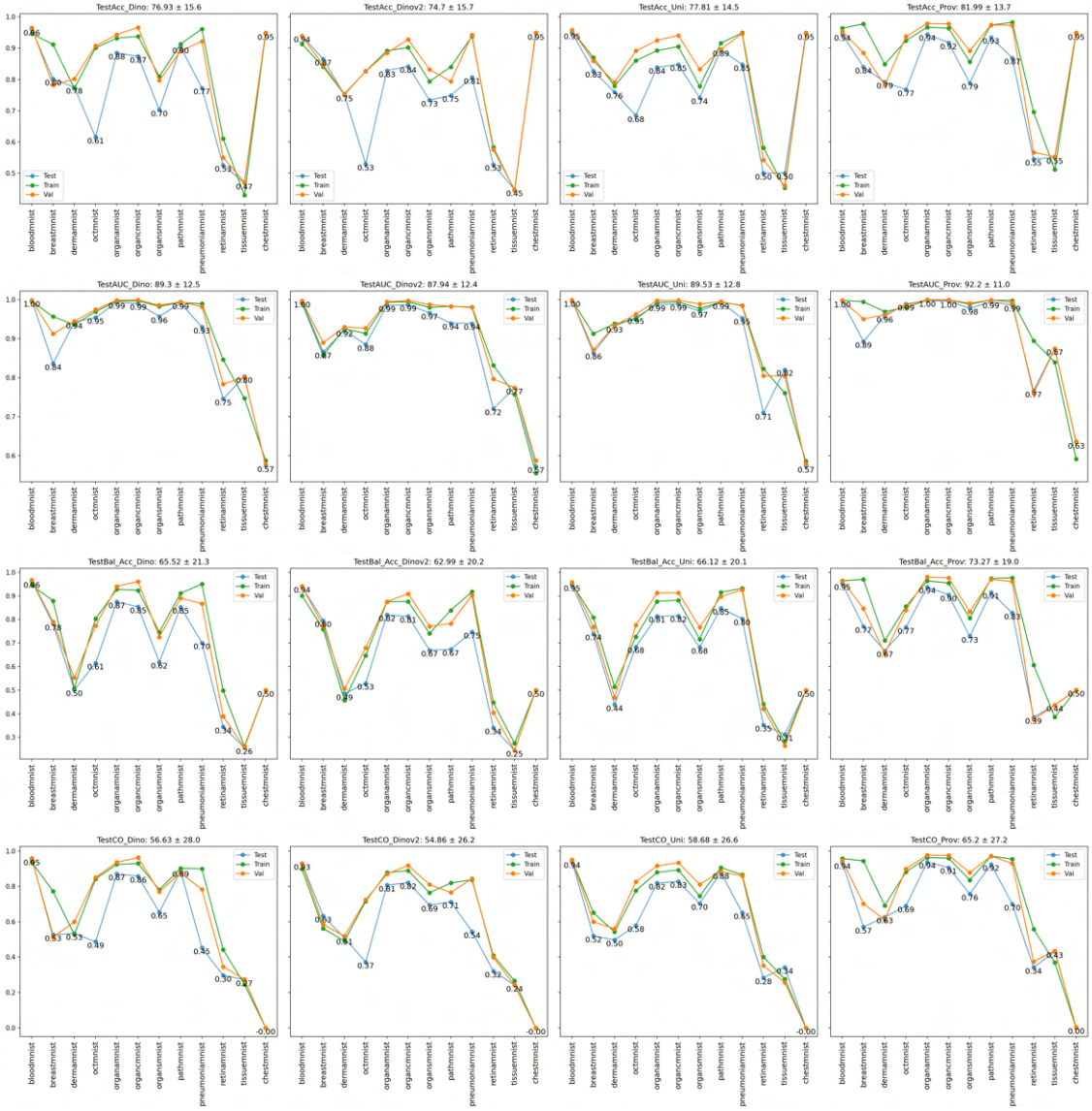


Figure 54: Every metric for 224×224 resolution with multi-domain multi-task pre-training with data-augmentation for every backbone. Training, validation, and test performance indicated in green, orange and blue color. The average and standard deviation over all 12 datasets is shown by the title of each plot. The test data is annotated.

## A.2 Performance Metrics for every Dataset on MedMNIST+

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 |
| **Classifier** | | | | | | | | |
| Dino SVM | 91.87 | 97.28 | 98.25 | 99.01 | 99.2 | 99.84 | 99.9 | 99.94 |
| Dino LightGBM | 89.89 | 95.88 | 97.52 | 97.9 | 99.12 | 99.81 | 99.9 | 99.92 |
| Dino RF | 80.44 | 85.79 | 91.06 | 90.79 | 97.31 | 98.56 | 99.33 | 99.4 |
| Dino KNN | 85.24 | 93.04 | 96.67 | 96.81 | - | - | - | - |
| Dinov2 SVM | 91.0 | 96.35 | 98.04 | 98.48 | 99.16 | 99.73 | 99.89 | 99.92 |
| Dinov2 LightGBM | 87.43 | 93.1 | 95.44 | 96.58 | 98.68 | 99.56 | 99.85 | 99.89 |
| Dinov2 RF | 69.95 | 76.38 | 84.33 | 88.45 | 94.76 | 96.42 | 98.48 | 99.15 |
| Dinov2 KNN | 77.87 | 84.36 | 92.11 | 93.92 | - | - | - | - |
| UNI SVM | 93.39 | 98.28 | 98.92 | 98.48 | 99.53 | 99.92 | 99.93 | 99.94 |
| UNI LightGBM | 92.63 | 97.34 | 98.39 | 97.84 | 99.47 | 99.88 | 99.94 | 99.93 |
| UNI RF | 83.37 | 91.14 | 93.72 | 93.66 | 97.99 | 99.35 | 99.67 | 99.52 |
| UNI KNN | 87.99 | 94.18 | 96.55 | 96.52 | - | - | - | - |
| Prov SVM | 94.65 | 98.01 | 98.28 | 98.71 | 99.65 | 99.9 | 99.91 | 99.93 |
| Prov LightGBM | 93.98 | 96.76 | 96.78 | 98.1 | 99.59 | 99.87 | 99.9 | 99.93 |
| Prov RF | 86.35 | 89.92 | 92.22 | 93.66 | 98.44 | 99.07 | 99.48 | 99.65 |
| Prov KNN | 90.5 | 92.58 | 94.94 | 96.96 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 89.83 | 97.19 | 98.1 | 98.51 | 99.01 | 99.87 | 99.92 | 99.94 |
| Dinov2 | 87.99 | 95.41 | 97.02 | 97.81 | 98.77 | 99.78 | 99.92 | 99.93 |
| UNI | 91.96 | 97.84 | 98.51 | 98.39 | 99.41 | 99.9 | 99.94 | 99.95 |
| Prov | 93.83 | 97.75 | 97.78 | 98.86 | 99.62 | 99.92 | 99.92 | 99.94 |
| **mm-PT** | | | | | | | | |
| Dino | 84.71 | 94.65 | 95.7 | 97.02 | 98.41 | 99.67 | 99.8 | 99.9 |
| Dinov2 | 86.29 | 94.39 | 94.39 | 96.84 | 98.44 | 99.66 | 99.72 | 99.86 |
| UNI | 85.56 | 94.01 | 96.52 | 97.08 | 98.58 | 99.59 | 99.87 | 99.88 |
| Prov | 88.54 | 94.36 | 97.75 | 98.07 | 98.89 | 99.77 | 99.91 | 99.93 |
| **mm-PT aug** | | | | | | | | |
| Dino | 86.06 | 92.84 | 95.38 | 96.14 | 98.53 | 99.54 | 99.77 | 99.83 |
| Dinov2 | 81.64 | 90.21 | 92.98 | 93.95 | 97.62 | 99.26 | 99.47 | 99.68 |
| UNI | 85.33 | 90.85 | 94.94 | 94.94 | 98.34 | 99.3 | 99.8 | 99.83 |
| Prov | 87.23 | 90.7 | 95.32 | 94.48 | 98.49 | 99.3 | 99.76 | 99.7 |

Table 10: Performance for accuracy and AUC for **BloodMNIST** across different resolutions, backbones, and training schemes.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 90.42 | 97.09 | 98.38 | 99.13 | 90.5 | 96.82 | 97.95 | 98.84 |
| Dino LightGBM | 87.59 | 95.1 | 97.41 | 97.78 | 88.16 | 95.18 | 97.1 | 97.54 |
| Dino RF | 73.85 | 80.39 | 88.49 | 87.99 | 76.89 | 83.23 | 89.49 | 89.18 |
| Dino KNN | 82.83 | 92.09 | 96.64 | 96.8 | 82.73 | 91.86 | 96.1 | 96.28 |
| Dinov2 SVM | 89.45 | 96.15 | 98.16 | 98.49 | 89.47 | 95.73 | 97.71 | 98.22 |
| Dinov2 LightGBM | 84.99 | 92.26 | 95.23 | 96.24 | 85.27 | 91.93 | 94.67 | 96.0 |
| Dinov2 RF | 61.76 | 71.03 | 81.01 | 85.44 | 64.25 | 72.05 | 81.54 | 86.42 |
| Dinov2 KNN | 74.57 | 82.41 | 91.47 | 93.32 | 74.03 | 81.64 | 90.76 | 92.88 |
| UNI SVM | 92.34 | 98.27 | 99.05 | 98.65 | 92.28 | 97.98 | 98.74 | 98.22 |
| UNI LightGBM | 91.17 | 97.16 | 98.41 | 97.95 | 91.38 | 96.89 | 98.12 | 97.47 |
| UNI RF | 77.17 | 88.18 | 91.68 | 92.3 | 80.37 | 89.6 | 92.63 | 92.57 |
| UNI KNN | 85.31 | 92.99 | 96.19 | 96.21 | 85.94 | 93.19 | 95.97 | 95.93 |
| Prov SVM | 94.1 | 98.07 | 98.43 | 98.87 | 93.75 | 97.68 | 97.98 | 98.5 |
| Prov LightGBM | 92.97 | 96.34 | 96.64 | 98.3 | 92.96 | 96.21 | 96.24 | 97.78 |
| Prov RF | 81.5 | 86.45 | 90.15 | 92.54 | 83.94 | 88.13 | 90.88 | 92.57 |
| Prov KNN | 88.92 | 91.68 | 94.48 | 96.85 | 88.89 | 91.31 | 94.09 | 96.45 |
| **Linear Probing** | | | | | | | | |
| Dino | 88.31 | 96.89 | 98.15 | 98.58 | 88.11 | 96.72 | 97.78 | 98.26 |
| Dinov2 | 85.48 | 94.66 | 96.87 | 97.6 | 85.92 | 94.63 | 96.52 | 97.44 |
| UNI | 90.4 | 97.65 | 98.62 | 98.46 | 90.6 | 97.47 | 98.26 | 98.12 |
| Prov | 93.1 | 97.48 | 97.68 | 98.95 | 92.79 | 97.37 | 97.4 | 98.67 |
| **mm-PT** | | | | | | | | |
| Dino | 80.69 | 94.12 | 95.04 | 97.35 | 82.09 | 93.75 | 94.97 | 96.52 |
| Dinov2 | 82.16 | 93.9 | 93.85 | 96.59 | 83.89 | 93.44 | 93.44 | 96.31 |
| UNI | 80.45 | 92.88 | 95.9 | 97.39 | 83.04 | 92.99 | 95.93 | 96.59 |
| Prov | 87.19 | 92.74 | 97.76 | 98.27 | 86.65 | 93.4 | 97.37 | 97.75 |
| **mm-PT aug** | | | | | | | | |
| Dino | 85.43 | 90.95 | 95.1 | 96.24 | 83.78 | 91.61 | 94.61 | 95.49 |
| Dinov2 | 79.06 | 89.65 | 91.85 | 94.05 | 78.57 | 88.58 | 91.79 | 92.94 |
| UNI | 81.53 | 89.16 | 94.4 | 94.69 | 82.79 | 89.3 | 94.09 | 94.09 |
| Prov | 83.6 | 89.15 | 95.72 | 95.25 | 85.0 | 89.14 | 94.54 | 93.56 |

Table 11: Performance for balanced accuracy and Cohen´s kappa for **BloodMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 85.26 | 85.9 | 91.03 | 87.18 | 88.99 | 90.79 | 91.81 | 92.98 |
| Dino LightGBM | 87.82 | 88.46 | 89.74 | 88.46 | 92.92 | 93.92 | 92.25 | 93.11 |
| Dino RF | 81.41 | 82.05 | 82.05 | 85.9 | 89.45 | 90.73 | 90.12 | 89.68 |
| Dino KNN | 78.21 | 86.54 | 85.26 | 87.18 | - | - | - | - |
| Dinov2 SVM | 83.97 | 85.9 | 83.33 | 87.18 | 86.15 | 90.73 | 91.4 | 90.56 |
| Dinov2 LightGBM | 83.97 | 83.33 | 87.18 | 86.54 | 82.5 | 90.08 | 93.42 | 89.2 |
| Dinov2 RF | 76.92 | 80.77 | 82.69 | 79.49 | 83.27 | 86.15 | 90.18 | 85.44 |
| Dinov2 KNN | 74.36 | 79.49 | 78.85 | 82.69 | - | - | - | - |
| UNI SVM | 84.62 | 82.69 | 84.62 | 87.18 | 88.53 | 90.43 | 92.29 | 91.08 |
| UNI LightGBM | 84.62 | 83.33 | 83.33 | 83.97 | 91.77 | 90.6 | 91.69 | 90.6 |
| UNI RF | 80.77 | 79.49 | 80.77 | 81.41 | 85.88 | 83.63 | 89.77 | 87.36 |
| UNI KNN | 81.41 | 81.41 | 82.69 | 82.05 | - | - | - | - |
| Prov SVM | 81.41 | 84.62 | 87.18 | 85.26 | 86.05 | 92.31 | 91.98 | 90.12 |
| Prov LightGBM | 83.97 | 87.18 | 85.9 | 85.9 | 88.45 | 93.21 | 93.69 | 90.71 |
| Prov RF | 78.85 | 78.21 | 80.13 | 81.41 | 85.09 | 87.09 | 89.39 | 85.32 |
| Prov KNN | 79.49 | 83.33 | 80.77 | 82.05 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 83.97 | 83.97 | 88.46 | 87.82 | 88.49 | 91.73 | 93.15 | 94.32 |
| Dinov2 | 82.69 | 83.33 | 88.46 | 88.46 | 84.09 | 88.62 | 91.33 | 90.94 |
| UNI | 82.69 | 80.77 | 84.62 | 84.62 | 82.92 | 86.78 | 90.18 | 87.89 |
| Prov | 79.49 | 84.62 | 85.26 | 86.54 | 82.04 | 91.0 | 92.15 | 89.08 |
| **mm-PT** | | | | | | | | |
| Dino | 80.13 | 85.26 | 80.77 | 83.97 | 81.24 | 85.61 | 86.28 | 87.36 |
| Dinov2 | 76.92 | 80.77 | 81.41 | 78.85 | 86.72 | 80.87 | 85.46 | 80.95 |
| UNI | 79.49 | 83.97 | 78.21 | 75.64 | 75.38 | 85.59 | 85.84 | 85.9 |
| Prov | 82.69 | 78.85 | 82.69 | 78.21 | 86.88 | 87.26 | 86.34 | 87.59 |
| **mm-PT aug** | | | | | | | | |
| Dino | 73.72 | 83.33 | 84.62 | 80.13 | 82.54 | 86.11 | 89.1 | 83.63 |
| Dinov2 | 81.41 | 71.15 | 80.77 | 86.54 | 77.44 | 79.2 | 80.99 | 86.61 |
| UNI | 83.97 | 83.97 | 83.97 | 82.69 | 90.06 | 85.74 | 89.87 | 86.34 |
| Prov | 82.69 | 82.69 | 85.9 | 83.97 | 87.61 | 85.71 | 89.33 | 89.26 |

Table 12: Performance for accuracy and AUC for **BreastMNIST** across different resolutions, backbones, and training schemes.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 76.38 | 79.82 | 85.59 | 81.45 | 58.47 | 62.47 | 75.73 | 65.88 |
| Dino LightGBM | 79.64 | 80.08 | 83.96 | 81.58 | 65.69 | 67.23 | 72.27 | 68.29 |
| Dino RF | 67.73 | 68.17 | 69.67 | 74.56 | 42.88 | 44.34 | 46.31 | 57.82 |
| Dino KNN | 64.04 | 78.01 | 77.88 | 80.7 | 33.63 | 62.08 | 59.81 | 65.33 |
| Dinov2 SVM | 74.0 | 78.32 | 72.06 | 78.45 | 54.1 | 61.25 | 51.01 | 63.59 |
| Dinov2 LightGBM | 74.0 | 72.81 | 78.45 | 78.01 | 54.1 | 51.85 | 63.59 | 62.08 |
| Dinov2 RF | 58.65 | 65.04 | 70.11 | 64.91 | 22.77 | 38.1 | 47.77 | 36.39 |
| Dinov2 KNN | 59.15 | 64.16 | 65.23 | 72.37 | 21.92 | 35.2 | 36.16 | 50.42 |
| UNI SVM | 72.93 | 72.37 | 77.44 | 81.45 | 53.98 | 50.42 | 58.4 | 65.88 |
| UNI LightGBM | 75.94 | 73.56 | 73.56 | 74.0 | 57.02 | 52.66 | 52.66 | 54.1 |
| UNI RF | 65.79 | 63.41 | 66.54 | 67.73 | 39.25 | 33.97 | 40.37 | 42.88 |
| UNI KNN | 69.99 | 69.99 | 76.88 | 73.43 | 45.83 | 45.83 | 55.0 | 50.68 |
| Prov SVM | 70.74 | 75.19 | 79.95 | 80.14 | 46.75 | 56.3 | 64.77 | 61.67 |
| Prov LightGBM | 76.25 | 78.45 | 76.82 | 77.57 | 56.32 | 63.59 | 59.94 | 60.61 |
| Prov RF | 63.72 | 62.53 | 66.1 | 67.73 | 33.8 | 31.15 | 38.94 | 42.88 |
| Prov KNN | 64.91 | 70.55 | 71.05 | 75.69 | 36.39 | 49.25 | 46.28 | 52.97 |
| **Linear Probing** | | | | | | | | |
| Dino | 74.0 | 76.25 | 80.83 | 82.64 | 54.1 | 56.32 | 67.77 | 67.84 |
| Dinov2 | 70.11 | 75.06 | 81.58 | 80.08 | 47.77 | 54.2 | 68.29 | 67.23 |
| UNI | 69.36 | 70.3 | 76.69 | 76.69 | 46.82 | 45.38 | 57.72 | 57.72 |
| Prov | 67.92 | 74.44 | 77.88 | 80.26 | 40.74 | 55.56 | 59.81 | 63.89 |
| **mm-PT** | | | | | | | | |
| Dino | 66.1 | 75.63 | 66.54 | 74.0 | 38.94 | 57.77 | 40.37 | 54.1 |
| Dinov2 | 78.95 | 70.3 | 79.01 | 63.72 | 49.68 | 45.38 | 55.12 | 33.8 |
| UNI | 64.91 | 74.0 | 64.04 | 55.51 | 36.39 | 54.1 | 33.63 | 15.12 |
| Prov | 69.36 | 61.47 | 68.61 | 60.28 | 46.82 | 29.9 | 45.83 | 27.06 |
| **mm-PT aug** | | | | | | | | |
| Dino | 72.24 | 73.56 | 79.7 | 78.13 | 39.98 | 52.66 | 60.31 | 52.7 |
| Dinov2 | 66.23 | 71.99 | 69.55 | 79.51 | 40.72 | 37.5 | 44.44 | 63.31 |
| UNI | 75.5 | 78.51 | 74.75 | 73.87 | 55.6 | 58.33 | 54.86 | 52.05 |
| Prov | 76.13 | 73.12 | 79.82 | 77.01 | 54.3 | 51.25 | 62.47 | 57.01 |

Table 13: Performance for balanced accuracy and Cohen´s kappa for **BreastMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 |
| **Classifier** | | | | | | | | |
| Dino SVM | 94.74 | 94.77 | 94.76 | 94.77 | 71.34 | 72.59 | 76.2 | 77.62 |
| Dino LightGBM | 94.75 | 94.76 | 94.76 | 94.77 | 69.41 | 73.24 | 76.07 | 77.33 |
| Dino RF | 94.74 | 94.74 | 94.74 | 94.74 | 66.3 | 68.77 | 70.84 | 72.28 |
| Dino KNN | 94.63 | 94.66 | 94.67 | 94.64 | - | - | - | - |
| Dinov2 SVM | 94.73 | 94.74 | 94.73 | 94.76 | 71.91 | 73.98 | 75.46 | 77.4 |
| Dinov2 LightGBM | 94.74 | 94.75 | 94.76 | 94.74 | 70.33 | 73.11 | 74.12 | 75.85 |
| Dinov2 RF | 94.74 | 94.74 | 94.74 | 94.74 | 65.96 | 69.0 | 69.34 | 70.82 |
| Dinov2 KNN | 94.63 | 94.67 | 94.68 | 94.67 | - | - | - | - |
| UNI SVM | 94.73 | 94.74 | 94.75 | 94.77 | 71.0 | 72.4 | 75.17 | 77.04 |
| UNI LightGBM | 94.73 | 94.75 | 94.76 | 94.75 | 68.68 | 71.73 | 74.31 | 75.97 |
| UNI RF | 94.74 | 94.74 | 94.74 | 94.74 | 65.05 | 67.81 | 70.65 | 71.38 |
| UNI KNN | 94.67 | 94.66 | 94.66 | 94.67 | - | - | - | - |
| Prov SVM | 94.73 | 94.76 | 94.77 | 94.76 | 70.83 | 71.69 | 75.56 | 76.67 |
| Prov LightGBM | 94.73 | 94.74 | 94.76 | 94.76 | 68.35 | 71.9 | 75.62 | 76.28 |
| Prov RF | 94.74 | 94.74 | 94.74 | 94.74 | 65.87 | 68.92 | 71.08 | 71.23 |
| Prov KNN | 94.66 | 94.67 | 94.65 | 94.67 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 94.74 | 94.75 | 94.76 | 94.77 | 70.98 | 74.59 | 77.89 | 78.92 |
| Dinov2 | 94.76 | 94.73 | 94.73 | 94.73 | 72.03 | 74.49 | 75.61 | 77.5 |
| UNI | 94.74 | 94.75 | 94.77 | 94.76 | 69.46 | 73.92 | 76.43 | 77.94 |
| Prov | 94.74 | 94.75 | 94.78 | 94.77 | 70.22 | 74.09 | 77.24 | 78.12 |
| **mm-PT** | | | | | | | | |
| Dino | 94.74 | 94.74 | 94.73 | 94.74 | 67.53 | 70.25 | 72.07 | 71.7 |
| Dinov2 | 94.74 | 94.74 | 94.75 | 94.75 | 69.65 | 68.25 | 67.57 | 68.08 |
| UNI | 94.74 | 94.74 | 94.75 | 94.75 | 69.92 | 70.19 | 72.28 | 69.55 |
| Prov | 94.74 | 94.76 | 94.78 | 94.78 | 71.24 | 74.75 | 75.6 | 76.65 |
| **mm-PT aug** | | | | | | | | |
| Dino | 94.74 | 94.74 | 94.74 | 94.74 | 64.64 | 59.91 | 57.96 | 57.43 |
| Dinov2 | 94.74 | 94.74 | 94.74 | 94.74 | 56.88 | 58.93 | 56.71 | 57.24 |
| UNI | 94.74 | 94.74 | 94.74 | 94.74 | 61.16 | 59.56 | 58.42 | 57.27 |
| Prov | 94.74 | 94.74 | 94.73 | 94.74 | 62.78 | 60.14 | 60.26 | 63.41 |

Table 14: Performance for accuracy and AUC for **ChestMNIST** for every resolution, every backbone, and every training scheme.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 50.38 | 50.67 | 51.26 | 51.84 | 1.24 | 2.14 | 4.09 | 5.87 |
| Dino LightGBM | 50.23 | 50.48 | 50.79 | 51.09 | 0.78 | 1.53 | 2.53 | 3.52 |
| Dino RF | 50.0 | 50.0 | 50.0 | 50.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Dino KNN | 50.24 | 50.26 | 50.36 | 50.51 | 0.76 | 0.81 | 1.18 | 1.67 |
| Dinov2 SVM | 50.42 | 50.51 | 51.09 | 51.52 | 1.31 | 1.62 | 3.51 | 4.88 |
| Dinov2 LightGBM | 50.23 | 50.35 | 50.65 | 50.75 | 0.75 | 1.13 | 2.14 | 2.48 |
| Dinov2 RF | 50.0 | 50.0 | 50.0 | 50.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Dinov2 KNN | 50.19 | 50.19 | 50.24 | 50.2 | 0.58 | 0.6 | 0.78 | 0.66 |
| UNI SVM | 50.36 | 50.63 | 51.24 | 51.81 | 1.14 | 2.03 | 4.03 | 5.88 |
| UNI LightGBM | 50.14 | 50.37 | 50.7 | 50.99 | 0.48 | 1.21 | 2.35 | 3.32 |
| UNI RF | 50.0 | 50.0 | 50.0 | 50.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| UNI KNN | 50.16 | 50.21 | 50.24 | 50.35 | 0.52 | 0.67 | 0.77 | 1.14 |
| Prov SVM | 50.37 | 50.66 | 51.16 | 51.69 | 1.18 | 2.08 | 3.71 | 5.49 |
| Prov LightGBM | 50.19 | 50.41 | 50.7 | 50.98 | 0.66 | 1.32 | 2.23 | 3.21 |
| Prov RF | 50.0 | 50.0 | 50.0 | 50.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Prov KNN | 50.16 | 50.24 | 50.33 | 50.44 | 0.52 | 0.77 | 1.06 | 1.41 |
| **Linear Probing** | | | | | | | | |
| Dino | 50.14 | 50.74 | 51.22 | 51.52 | 0.46 | 2.23 | 3.91 | 4.77 |
| Dinov2 | 50.14 | 50.31 | 50.37 | 50.67 | 0.49 | 1.01 | 1.19 | 2.22 |
| UNI | 50.01 | 50.31 | 50.81 | 50.98 | 0.05 | 1.04 | 2.73 | 3.29 |
| Prov | 50.14 | 50.41 | 51.01 | 51.27 | 0.45 | 1.32 | 3.24 | 4.15 |
| **mm-PT** | | | | | | | | |
| Dino | 50.07 | 50.09 | 50.21 | 50.55 | 0.25 | 0.29 | 0.69 | 1.54 |
| Dinov2 | 50.19 | 50.06 | 50.1 | 50.12 | 0.63 | 0.19 | 0.33 | 0.41 |
| UNI | 50.15 | 50.3 | 50.5 | 50.09 | 0.49 | 0.97 | 1.5 | 0.3 |
| Prov | 50.07 | 50.23 | 50.54 | 50.87 | 0.23 | 0.79 | 1.78 | 2.74 |
| **mm-PT aug** | | | | | | | | |
| Dino | 50.02 | 50.0 | 50.0 | 50.0 | 0.08 | 0.0 | -0.0 | -0.0 |
| Dinov2 | 50.0 | 50.01 | 50.0 | 50.0 | -0.0 | 0.03 | 0.0 | -0.0 |
| UNI | 50.0 | 50.0 | 50.0 | 50.0 | 0.0 | -0.01 | -0.0 | -0.0 |
| Prov | 50.0 | 50.0 | 50.0 | 50.08 | -0.0 | 0.0 | 0.01 | 0.28 |

Table 15: Performance for balanced accuracy and Cohen´s kappa for **ChestMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 76.26 | 78.45 | 80.75 | 84.99 | 92.67 | 94.67 | 95.72 | 96.56 |
| Dino LightGBM | 77.41 | 79.4 | 81.35 | 84.04 | 93.45 | 94.79 | 95.87 | 96.52 |
| Dino RF | 71.17 | 71.02 | 70.77 | 71.17 | 89.2 | 89.29 | 90.44 | 89.59 |
| Dino KNN | 73.97 | 75.91 | 76.51 | 78.35 | - | - | - | - |
| Dinov2 SVM | 77.16 | 79.55 | 81.5 | 83.19 | 93.36 | 94.06 | 95.13 | 95.66 |
| Dinov2 LightGBM | 76.96 | 78.25 | 80.4 | 81.25 | 92.94 | 93.55 | 94.72 | 95.39 |
| Dinov2 RF | 71.67 | 71.52 | 69.98 | 71.47 | 87.52 | 87.18 | 88.06 | 88.19 |
| Dinov2 KNN | 73.02 | 73.07 | 73.27 | 74.76 | - | - | - | - |
| UNI SVM | 77.76 | 80.35 | 82.74 | 84.09 | 92.76 | 94.59 | 95.85 | 96.18 |
| UNI LightGBM | 77.46 | 80.2 | 81.95 | 82.74 | 93.16 | 95.1 | 95.4 | 96.13 |
| UNI RF | 71.27 | 70.12 | 70.42 | 69.23 | 89.35 | 88.75 | 89.89 | 90.5 |
| UNI KNN | 73.32 | 74.86 | 76.66 | 77.21 | - | - | - | - |
| Prov SVM | 78.0 | 80.3 | 82.94 | 84.24 | 93.77 | 95.13 | 96.2 | 96.58 |
| Prov LightGBM | 78.8 | 80.9 | 81.1 | 83.54 | 93.66 | 95.49 | 95.8 | 96.75 |
| Prov RF | 71.17 | 70.42 | 70.62 | 71.32 | 89.64 | 90.14 | 89.56 | 88.44 |
| Prov KNN | 72.87 | 74.76 | 74.66 | 76.46 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 75.71 | 79.95 | 81.65 | 84.09 | 91.9 | 95.47 | 96.18 | 96.79 |
| Dinov2 | 76.11 | 78.65 | 80.8 | 82.54 | 91.97 | 93.92 | 95.13 | 96.01 |
| UNI | 76.56 | 79.9 | 82.34 | 84.79 | 91.98 | 94.71 | 96.01 | 96.3 |
| Prov | 77.51 | 80.4 | 82.44 | 83.64 | 93.26 | 95.36 | 96.4 | 96.51 |
| **mm-PT** | | | | | | | | |
| Dino | 72.22 | 74.91 | 76.01 | 78.55 | 87.02 | 92.37 | 92.8 | 94.85 |
| Dinov2 | 69.33 | 73.72 | 76.01 | 76.16 | 88.68 | 90.9 | 93.16 | 93.09 |
| UNI | 72.97 | 76.36 | 78.3 | 79.0 | 89.67 | 92.75 | 94.33 | 94.45 |
| Prov | 74.26 | 78.75 | 80.95 | 80.15 | 89.43 | 93.26 | 95.58 | 95.37 |
| **mm-PT aug** | | | | | | | | |
| Dino | 74.41 | 73.82 | 75.81 | 77.51 | 88.82 | 90.71 | 93.14 | 94.27 |
| Dinov2 | 71.52 | 74.66 | 72.47 | 75.06 | 87.29 | 91.35 | 90.96 | 92.16 |
| UNI | 71.82 | 74.61 | 75.21 | 75.91 | 89.21 | 91.19 | 92.87 | 93.23 |
| Prov | 73.47 | 75.61 | 75.36 | 79.15 | 88.45 | 91.22 | 93.32 | 95.78 |

Table 16: Performance for accuracy and AUC for **DermaMNIST** for every resolution, every backbone and every training scheme.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 |
| **Classifier** | | | | | | | | |
| Dino SVM | 43.38 | 52.19 | 59.08 | 66.03 | 48.13 | 54.06 | 59.9 | 69.21 |
| Dino LightGBM | 45.19 | 46.81 | 52.68 | 57.17 | 51.54 | 56.34 | 60.98 | 67.2 |
| Dino RF | 24.07 | 21.38 | 21.24 | 21.76 | 28.11 | 25.91 | 24.4 | 24.18 |
| Dino KNN | 36.61 | 39.81 | 40.43 | 45.67 | 43.89 | 47.35 | 49.08 | 53.57 |
| Dinov2 SVM | 45.72 | 51.81 | 58.14 | 62.59 | 51.09 | 55.89 | 61.33 | 65.29 |
| Dinov2 LightGBM | 43.62 | 42.27 | 50.33 | 51.12 | 50.3 | 53.12 | 57.94 | 60.06 |
| Dinov2 RF | 23.26 | 22.63 | 20.24 | 23.27 | 29.57 | 28.35 | 21.44 | 27.47 |
| Dinov2 KNN | 33.54 | 32.13 | 31.42 | 35.23 | 41.93 | 40.9 | 39.2 | 43.06 |
| UNI SVM | 51.16 | 57.24 | 63.01 | 65.14 | 52.37 | 58.61 | 64.58 | 67.4 |
| UNI LightGBM | 48.19 | 48.55 | 52.82 | 54.26 | 52.35 | 58.05 | 62.36 | 64.26 |
| UNI RF | 23.04 | 20.3 | 20.06 | 18.09 | 27.19 | 20.56 | 20.42 | 14.3 |
| UNI KNN | 35.43 | 38.46 | 40.43 | 42.66 | 41.79 | 45.49 | 49.91 | 51.34 |
| Prov SVM | 50.58 | 56.3 | 60.6 | 65.29 | 53.42 | 58.6 | 65.0 | 67.65 |
| Prov LightGBM | 50.83 | 49.9 | 51.92 | 57.54 | 54.97 | 59.45 | 60.53 | 65.93 |
| Prov RF | 22.96 | 19.87 | 19.98 | 22.78 | 26.67 | 21.0 | 22.8 | 26.03 |
| Prov KNN | 35.78 | 34.06 | 35.41 | 37.94 | 41.71 | 44.06 | 43.99 | 48.23 |
| **Linear Probing** | | | | | | | | |
| Dino | 42.38 | 57.11 | 61.29 | 66.96 | 47.66 | 58.78 | 62.68 | 68.12 |
| Dinov2 | 43.28 | 49.53 | 58.55 | 63.26 | 48.83 | 55.1 | 60.45 | 64.67 |
| UNI | 49.84 | 58.3 | 63.01 | 66.33 | 50.69 | 58.69 | 64.42 | 69.51 |
| Prov | 51.09 | 59.75 | 63.9 | 64.59 | 52.27 | 60.47 | 65.03 | 67.34 |
| **mm-PT** | | | | | | | | |
| Dino | 31.22 | 52.1 | 51.5 | 58.73 | 39.32 | 52.31 | 46.85 | 58.45 |
| Dinov2 | 38.33 | 39.28 | 48.76 | 49.5 | 43.19 | 41.71 | 49.55 | 50.73 |
| UNI | 38.85 | 47.23 | 52.49 | 50.39 | 46.54 | 52.07 | 55.23 | 55.21 |
| Prov | 33.68 | 54.95 | 59.5 | 58.72 | 40.48 | 56.18 | 59.1 | 58.89 |
| **mm-PT aug** | | | | | | | | |
| Dino | 40.61 | 41.7 | 40.78 | 50.19 | 45.32 | 45.29 | 45.19 | 53.34 |
| Dinov2 | 29.86 | 39.44 | 40.52 | 48.52 | 38.08 | 46.35 | 43.79 | 50.77 |
| UNI | 40.54 | 44.61 | 50.2 | 43.82 | 42.8 | 49.03 | 50.88 | 49.73 |
| Prov | 34.28 | 40.78 | 47.89 | 66.71 | 40.03 | 46.08 | 53.0 | 62.62 |

Table 17: Performance for balanced accuracy and Cohen´s kappa for **DermaMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 64.9 | 70.8 | 75.4 | 73.4 | 91.51 | 95.8 | 97.79 | 98.66 |
| Dino LightGBM | 61.3 | 71.5 | 75.8 | 73.2 | 90.15 | 96.1 | 98.3 | 98.98 |
| Dino RF | 44.4 | 48.1 | 52.0 | 50.5 | 83.23 | 88.45 | 94.96 | 97.42 |
| Dino KNN | 46.5 | 61.1 | 72.2 | 74.1 | - | - | - | - |
| Dinov2 SVM | 62.0 | 72.9 | 78.1 | 76.9 | 90.62 | 94.93 | 97.23 | 97.84 |
| Dinov2 LightGBM | 59.3 | 68.9 | 74.9 | 73.0 | 89.17 | 94.54 | 97.7 | 98.24 |
| Dinov2 RF | 46.2 | 45.7 | 47.8 | 52.5 | 81.18 | 83.98 | 90.04 | 93.75 |
| Dinov2 KNN | 46.8 | 54.8 | 63.5 | 66.8 | - | - | - | - |
| UNI SVM | 61.9 | 69.6 | 74.2 | 66.3 | 91.22 | 95.3 | 97.2 | 97.77 |
| UNI LightGBM | 59.6 | 69.7 | 73.3 | 76.3 | 89.21 | 95.02 | 96.91 | 98.16 |
| UNI RF | 43.6 | 46.6 | 49.2 | 50.1 | 82.06 | 86.49 | 92.52 | 94.07 |
| UNI KNN | 44.0 | 52.0 | 61.4 | 67.8 | - | - | - | - |
| Prov SVM | 62.9 | 74.4 | 75.2 | 77.5 | 91.16 | 96.14 | 97.59 | 98.35 |
| Prov LightGBM | 61.2 | 70.1 | 73.1 | 77.3 | 90.21 | 95.26 | 97.24 | 98.56 |
| Prov RF | 43.4 | 44.8 | 47.7 | 53.1 | 82.57 | 82.14 | 91.76 | 93.94 |
| Prov KNN | 44.0 | 52.0 | 61.9 | 64.6 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 62.1 | 72.3 | 75.9 | 74.8 | 92.16 | 96.2 | 97.72 | 98.3 |
| Dinov2 | 61.7 | 72.9 | 78.8 | 75.3 | 90.76 | 95.61 | 98.03 | 98.15 |
| UNI | 58.1 | 69.5 | 74.9 | 66.4 | 89.72 | 95.25 | 97.22 | 96.89 |
| Prov | 61.8 | 72.9 | 75.5 | 77.5 | 91.15 | 96.02 | 97.27 | 98.15 |
| **mm-PT** | | | | | | | | |
| Dino | 64.7 | 83.1 | 82.2 | 82.0 | 90.45 | 96.94 | 97.59 | 98.72 |
| Dinov2 | 63.0 | 67.4 | 81.5 | 76.7 | 89.1 | 92.51 | 97.7 | 97.43 |
| UNI | 64.5 | 77.9 | 79.8 | 82.0 | 89.43 | 95.95 | 96.74 | 98.37 |
| Prov | 65.8 | 84.8 | 85.6 | 88.6 | 90.72 | 97.5 | 97.98 | 98.82 |
| **mm-PT aug** | | | | | | | | |
| Dino | 55.8 | 62.6 | 55.5 | 61.5 | 87.03 | 91.36 | 95.38 | 95.4 |
| Dinov2 | 44.7 | 55.1 | 46.7 | 52.9 | 78.26 | 84.37 | 84.09 | 88.49 |
| UNI | 54.8 | 59.2 | 66.6 | 68.4 | 85.74 | 88.76 | 93.64 | 95.44 |
| Prov | 50.9 | 59.2 | 75.6 | 76.7 | 86.43 | 92.09 | 96.58 | 98.89 |

Table 18: Performance for accuracy and AUC for **OctMNIST** for every resolution, backbone, and training scheme.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 64.9 | 70.8 | 75.4 | 73.4 | 53.2 | 61.07 | 67.2 | 64.53 |
| Dino LightGBM | 61.3 | 71.5 | 75.8 | 73.2 | 48.4 | 62.0 | 67.73 | 64.27 |
| Dino RF | 44.4 | 48.1 | 52.0 | 50.5 | 25.87 | 30.8 | 36.0 | 34.0 |
| Dino KNN | 46.5 | 61.1 | 72.2 | 74.1 | 28.67 | 48.13 | 62.93 | 65.47 |
| Dinov2 SVM | 62.0 | 72.9 | 78.1 | 76.9 | 49.33 | 63.87 | 70.8 | 69.2 |
| Dinov2 LightGBM | 59.3 | 68.9 | 74.9 | 73.0 | 45.73 | 58.53 | 66.53 | 64.0 |
| Dinov2 RF | 46.2 | 45.7 | 47.8 | 52.5 | 28.27 | 27.6 | 30.4 | 36.67 |
| Dinov2 KNN | 46.8 | 54.8 | 63.5 | 66.8 | 29.07 | 39.73 | 51.33 | 55.73 |
| UNI SVM | 61.9 | 69.6 | 74.2 | 66.3 | 49.2 | 59.47 | 65.6 | 55.07 |
| UNI LightGBM | 59.6 | 69.7 | 73.3 | 76.3 | 46.13 | 59.6 | 64.4 | 68.4 |
| UNI RF | 43.6 | 46.6 | 49.2 | 50.1 | 24.8 | 28.8 | 32.27 | 33.47 |
| UNI KNN | 44.0 | 52.0 | 61.4 | 67.8 | 25.33 | 36.0 | 48.53 | 57.07 |
| Prov SVM | 62.9 | 74.4 | 75.2 | 77.5 | 50.53 | 65.87 | 66.93 | 70.0 |
| Prov LightGBM | 61.2 | 70.1 | 73.1 | 77.3 | 48.27 | 60.13 | 64.13 | 69.73 |
| Prov RF | 43.4 | 44.8 | 47.7 | 53.1 | 24.53 | 26.4 | 30.27 | 37.47 |
| Prov KNN | 44.0 | 52.0 | 61.9 | 64.6 | 25.33 | 36.0 | 49.2 | 52.8 |
| **Linear Probing** | | | | | | | | |
| Dino | 62.1 | 72.3 | 75.9 | 74.8 | 49.47 | 63.07 | 67.87 | 66.4 |
| Dinov2 | 61.7 | 72.9 | 78.8 | 75.3 | 48.93 | 63.87 | 71.73 | 67.07 |
| UNI | 58.1 | 69.5 | 74.9 | 66.4 | 44.13 | 59.33 | 66.53 | 55.2 |
| Prov | 61.8 | 72.9 | 75.5 | 77.5 | 49.07 | 63.87 | 67.33 | 70.0 |
| **mm-PT** | | | | | | | | |
| Dino | 64.7 | 83.1 | 82.2 | 82.0 | 52.93 | 77.47 | 76.27 | 76.0 |
| Dinov2 | 63.0 | 67.4 | 81.5 | 76.7 | 50.67 | 56.53 | 75.33 | 68.93 |
| UNI | 64.5 | 77.9 | 79.8 | 82.0 | 52.67 | 70.53 | 73.07 | 76.0 |
| Prov | 65.8 | 84.8 | 85.6 | 88.6 | 54.4 | 79.73 | 80.8 | 84.8 |
| **mm-PT aug** | | | | | | | | |
| Dino | 55.8 | 62.6 | 55.5 | 61.5 | 41.07 | 50.13 | 40.67 | 48.67 |
| Dinov2 | 44.7 | 55.1 | 46.7 | 52.9 | 26.27 | 40.13 | 28.93 | 37.2 |
| UNI | 54.8 | 59.2 | 66.6 | 68.4 | 39.73 | 45.6 | 55.47 | 57.87 |
| Prov | 50.9 | 59.2 | 75.6 | 76.7 | 34.53 | 45.6 | 67.47 | 68.93 |

Table 19: Performance for balanced accuracy and Cohen´s kappa for **OctMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 90.85 | 94.06 | 95.13 | 94.9 | 99.05 | 99.68 | 99.72 | 99.73 |
| Dino LightGBM | 89.0 | 93.31 | 93.69 | 93.31 | 99.22 | 99.71 | 99.72 | 99.69 |
| Dino RF | 74.45 | 81.83 | 83.69 | 85.89 | 96.34 | 97.96 | 98.19 | 98.42 |
| Dino KNN | 84.59 | 90.75 | 91.25 | 90.69 | - | - | - | - |
| Dinov2 SVM | 89.16 | 91.26 | 92.23 | 92.88 | 98.79 | 99.26 | 99.44 | 99.56 |
| Dinov2 LightGBM | 85.14 | 88.06 | 88.9 | 89.89 | 98.66 | 99.21 | 99.29 | 99.41 |
| Dinov2 RF | 66.18 | 71.93 | 73.34 | 75.81 | 94.45 | 96.19 | 96.43 | 96.99 |
| Dinov2 KNN | 75.13 | 82.39 | 83.28 | 84.42 | - | - | - | - |
| UNI SVM | 88.05 | 89.94 | 90.04 | 91.74 | 98.6 | 99.11 | 99.17 | 99.42 |
| UNI LightGBM | 83.23 | 85.74 | 84.77 | 87.29 | 98.39 | 98.88 | 98.74 | 99.08 |
| UNI RF | 67.98 | 73.16 | 73.7 | 75.55 | 95.28 | 97.27 | 97.18 | 97.47 |
| UNI KNN | 73.46 | 77.24 | 77.04 | 78.32 | - | - | - | - |
| Prov SVM | 87.9 | 91.34 | 91.62 | 91.09 | 98.59 | 99.35 | 99.35 | 99.32 |
| Prov LightGBM | 83.88 | 86.91 | 87.3 | 86.95 | 98.35 | 99.11 | 99.1 | 99.09 |
| Prov RF | 67.77 | 75.2 | 76.27 | 74.27 | 95.12 | 97.32 | 97.66 | 97.21 |
| Prov KNN | 75.11 | 80.91 | 81.26 | 77.07 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 89.74 | 94.12 | 95.09 | 94.83 | 99.27 | 99.75 | 99.8 | 99.79 |
| Dinov2 | 87.28 | 90.22 | 92.08 | 92.32 | 98.88 | 99.4 | 99.58 | 99.63 |
| UNI | 84.62 | 88.05 | 87.59 | 89.98 | 98.64 | 99.08 | 99.06 | 99.3 |
| Prov | 85.53 | 90.09 | 90.12 | 89.82 | 98.61 | 99.37 | 99.33 | 99.31 |
| **mm-PT** | | | | | | | | |
| Dino | 87.01 | 92.53 | 93.98 | 93.58 | 98.7 | 99.61 | 99.69 | 99.72 |
| Dinov2 | 84.25 | 88.18 | 89.44 | 89.9 | 98.29 | 98.98 | 99.12 | 99.28 |
| UNI | 89.22 | 92.03 | 93.23 | 90.54 | 99.11 | 99.51 | 99.72 | 99.42 |
| Prov | 89.7 | 95.21 | 94.63 | 96.06 | 99.14 | 99.74 | 99.65 | 99.86 |
| **mm-PT aug** | | | | | | | | |
| Dino | 87.06 | 89.03 | 87.43 | 88.47 | 98.83 | 99.25 | 99.19 | 99.34 |
| Dinov2 | 75.36 | 82.41 | 81.29 | 82.81 | 96.5 | 98.07 | 98.17 | 98.54 |
| UNI | 85.57 | 87.3 | 87.25 | 83.88 | 98.59 | 99.03 | 98.96 | 98.74 |
| Prov | 87.8 | 89.19 | 88.2 | 94.45 | 99.04 | 99.17 | 99.18 | 99.79 |

Table 20: Performance for accuracy and AUC for **OrganaMNIST** for every resolution, every backbone and every training scheme.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 90.01 | 92.99 | 94.47 | 94.45 | 89.73 | 93.33 | 94.54 | 94.28 |
| Dino LightGBM | 87.74 | 92.04 | 92.57 | 92.13 | 87.63 | 92.48 | 92.91 | 92.49 |
| Dino RF | 69.67 | 76.58 | 78.61 | 82.46 | 70.98 | 79.48 | 81.61 | 84.11 |
| Dino KNN | 84.05 | 89.96 | 90.52 | 89.87 | 82.65 | 89.62 | 90.18 | 89.55 |
| Dinov2 SVM | 88.31 | 90.15 | 91.31 | 92.07 | 87.82 | 90.19 | 91.28 | 92.01 |
| Dinov2 LightGBM | 83.36 | 86.33 | 87.14 | 88.16 | 83.29 | 86.59 | 87.53 | 88.64 |
| Dinov2 RF | 60.42 | 65.89 | 67.87 | 71.4 | 61.47 | 68.22 | 69.89 | 72.72 |
| Dinov2 KNN | 72.58 | 80.19 | 80.96 | 82.46 | 71.94 | 80.22 | 81.22 | 82.5 |
| UNI SVM | 87.12 | 89.1 | 89.44 | 91.13 | 86.58 | 88.72 | 88.83 | 90.73 |
| UNI LightGBM | 81.23 | 83.07 | 82.8 | 85.59 | 81.15 | 83.99 | 82.91 | 85.74 |
| UNI RF | 64.33 | 67.78 | 68.66 | 71.02 | 63.69 | 69.72 | 70.38 | 72.48 |
| UNI KNN | 71.17 | 74.5 | 73.96 | 75.59 | 70.09 | 74.44 | 74.21 | 75.65 |
| Prov SVM | 87.07 | 90.48 | 90.91 | 90.37 | 86.41 | 90.28 | 90.6 | 90.0 |
| Prov LightGBM | 82.52 | 85.1 | 85.57 | 85.77 | 81.88 | 85.3 | 85.74 | 85.36 |
| Prov RF | 64.78 | 71.19 | 72.05 | 70.42 | 63.48 | 72.05 | 73.28 | 71.04 |
| Prov KNN | 73.55 | 78.64 | 78.79 | 75.61 | 71.97 | 78.57 | 78.95 | 74.25 |
| **Linear Probing** | | | | | | | | |
| Dino | 88.87 | 93.05 | 94.35 | 94.22 | 88.47 | 93.39 | 94.49 | 94.19 |
| Dinov2 | 86.55 | 88.85 | 91.14 | 91.36 | 85.72 | 89.03 | 91.11 | 91.38 |
| UNI | 83.74 | 86.89 | 86.85 | 89.1 | 82.74 | 86.6 | 86.08 | 88.75 |
| Prov | 84.71 | 89.1 | 89.39 | 89.22 | 83.76 | 88.89 | 88.91 | 88.58 |
| **mm-PT** | | | | | | | | |
| Dino | 86.33 | 91.42 | 92.96 | 93.22 | 85.43 | 91.61 | 93.24 | 92.8 |
| Dinov2 | 83.4 | 87.49 | 88.82 | 89.52 | 82.31 | 86.74 | 88.15 | 88.67 |
| UNI | 88.38 | 91.13 | 92.48 | 89.3 | 87.89 | 91.05 | 92.4 | 89.38 |
| Prov | 88.97 | 94.47 | 94.14 | 95.66 | 88.43 | 94.62 | 93.97 | 95.58 |
| **mm-PT aug** | | | | | | | | |
| Dino | 85.41 | 87.77 | 86.39 | 87.45 | 85.46 | 87.68 | 85.91 | 87.05 |
| Dinov2 | 74.61 | 81.57 | 81.2 | 81.97 | 72.37 | 80.24 | 79.03 | 80.72 |
| UNI | 85.21 | 87.49 | 86.12 | 81.42 | 83.83 | 85.77 | 85.68 | 81.9 |
| Prov | 87.3 | 88.51 | 87.75 | 93.79 | 86.33 | 87.88 | 86.78 | 93.77 |

Table 21: Performance for balanced accuracy and Cohen´s kappa for **OrganAMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 89.7 | 92.2 | 92.56 | 91.36 | 98.92 | 99.34 | 99.45 | 99.26 |
| Dino LightGBM | 87.94 | 90.23 | 89.44 | 88.57 | 99.05 | 99.47 | 99.44 | 99.35 |
| Dino RF | 73.7 | 77.8 | 76.69 | 78.21 | 96.68 | 97.63 | 97.37 | 97.54 |
| Dino KNN | 83.63 | 87.52 | 87.13 | 86.0 | - | - | - | - |
| Dinov2 SVM | 88.51 | 89.34 | 88.69 | 88.11 | 98.45 | 98.92 | 98.88 | 98.9 |
| Dinov2 LightGBM | 84.06 | 83.8 | 82.87 | 83.93 | 98.36 | 98.74 | 98.61 | 98.74 |
| Dinov2 RF | 63.88 | 67.19 | 68.01 | 70.42 | 94.6 | 95.71 | 95.78 | 96.24 |
| Dinov2 KNN | 74.81 | 78.61 | 78.76 | 79.54 | - | - | - | - |
| UNI SVM | 86.42 | 86.84 | 86.76 | 86.2 | 98.33 | 98.64 | 98.74 | 98.71 |
| UNI LightGBM | 82.12 | 82.21 | 81.71 | 81.3 | 98.3 | 98.64 | 98.51 | 98.41 |
| UNI RF | 67.67 | 71.56 | 72.86 | 71.06 | 95.36 | 96.87 | 97.01 | 96.75 |
| UNI KNN | 73.38 | 76.38 | 76.3 | 75.51 | - | - | - | - |
| Prov SVM | 87.99 | 88.47 | 87.09 | 85.02 | 98.56 | 98.82 | 98.77 | 98.45 |
| Prov LightGBM | 83.43 | 84.3 | 83.23 | 81.3 | 98.45 | 98.84 | 98.64 | 98.52 |
| Prov RF | 67.33 | 73.62 | 72.65 | 72.07 | 95.35 | 97.07 | 96.88 | 96.85 |
| Prov KNN | 75.52 | 78.35 | 78.82 | 75.17 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 88.21 | 92.02 | 92.16 | 91.72 | 99.1 | 99.6 | 99.61 | 99.55 |
| Dinov2 | 86.08 | 87.49 | 87.33 | 87.35 | 98.55 | 99.08 | 99.12 | 99.12 |
| UNI | 84.25 | 85.19 | 84.51 | 83.41 | 98.5 | 98.89 | 98.78 | 98.71 |
| Prov | 85.93 | 86.83 | 85.75 | 83.67 | 98.64 | 98.99 | 98.93 | 98.71 |
| **mm-PT** | | | | | | | | |
| Dino | 85.11 | 90.76 | 90.82 | 92.39 | 98.57 | 99.37 | 99.35 | 99.56 |
| Dinov2 | 86.03 | 88.32 | 88.85 | 88.77 | 98.51 | 98.98 | 99.17 | 99.15 |
| UNI | 87.61 | 89.81 | 92.41 | 89.86 | 98.73 | 99.15 | 99.61 | 99.23 |
| Prov | 89.3 | 92.82 | 94.02 | 94.69 | 98.96 | 99.58 | 99.65 | 99.66 |
| **mm-PT aug** | | | | | | | | |
| Dino | 87.38 | 85.74 | 86.31 | 87.43 | 98.72 | 98.74 | 99.06 | 99.01 |
| Dinov2 | 80.49 | 83.58 | 83.12 | 84.18 | 97.64 | 98.55 | 98.47 | 98.69 |
| UNI | 85.32 | 85.21 | 87.06 | 84.74 | 98.45 | 98.73 | 98.99 | 98.9 |
| Prov | 88.23 | 87.33 | 85.59 | 91.74 | 98.9 | 99.02 | 99.0 | 99.6 |

Table 22: Performance for accuracy and AUC for **OrgancMNIST** for every resolution, every backbone and every training scheme.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 88.48 | 91.16 | 91.39 | 90.07 | 88.36 | 91.18 | 91.59 | 90.23 |
| Dino LightGBM | 86.4 | 88.37 | 87.03 | 86.24 | 86.35 | 88.95 | 88.04 | 87.06 |
| Dino RF | 69.32 | 73.15 | 70.77 | 73.3 | 69.79 | 74.67 | 73.41 | 75.18 |
| Dino KNN | 81.85 | 85.52 | 84.87 | 83.99 | 81.42 | 85.9 | 85.45 | 84.18 |
| Dinov2 SVM | 87.01 | 87.64 | 86.89 | 85.89 | 87.01 | 87.95 | 87.22 | 86.55 |
| Dinov2 LightGBM | 81.74 | 81.27 | 79.84 | 81.01 | 81.94 | 81.67 | 80.62 | 81.82 |
| Dinov2 RF | 57.56 | 61.34 | 61.39 | 64.7 | 58.15 | 62.28 | 63.31 | 66.15 |
| Dinov2 KNN | 72.17 | 75.78 | 75.82 | 76.42 | 71.23 | 75.8 | 75.98 | 76.83 |
| UNI SVM | 84.51 | 84.52 | 83.86 | 83.53 | 84.64 | 85.13 | 85.01 | 84.38 |
| UNI LightGBM | 79.19 | 78.9 | 77.09 | 77.25 | 79.75 | 79.88 | 79.28 | 78.81 |
| UNI RF | 62.11 | 65.01 | 66.06 | 63.42 | 62.91 | 67.54 | 69.06 | 66.98 |
| UNI KNN | 70.1 | 72.68 | 71.2 | 70.57 | 69.79 | 73.3 | 73.13 | 72.25 |
| Prov SVM | 86.23 | 86.7 | 84.95 | 82.23 | 86.42 | 86.97 | 85.4 | 83.05 |
| Prov LightGBM | 80.95 | 81.31 | 79.71 | 77.75 | 81.25 | 82.24 | 81.02 | 78.83 |
| Prov RF | 62.11 | 67.72 | 66.1 | 65.68 | 62.54 | 69.94 | 68.84 | 68.15 |
| Prov KNN | 72.78 | 74.56 | 74.88 | 71.0 | 72.22 | 75.52 | 76.04 | 71.83 |
| **Linear Probing** | | | | | | | | |
| Dino | 86.87 | 90.65 | 90.72 | 90.35 | 86.66 | 90.98 | 91.14 | 90.64 |
| Dinov2 | 84.05 | 85.57 | 85.24 | 85.02 | 84.24 | 85.86 | 85.68 | 85.7 |
| UNI | 82.05 | 82.55 | 81.0 | 79.93 | 82.19 | 83.27 | 82.47 | 81.21 |
| Prov | 83.87 | 84.83 | 83.28 | 80.82 | 84.08 | 85.12 | 83.89 | 81.52 |
| **mm-PT** | | | | | | | | |
| Dino | 84.31 | 89.95 | 89.71 | 91.11 | 83.22 | 89.57 | 89.62 | 91.4 |
| Dinov2 | 84.6 | 86.83 | 87.78 | 86.89 | 84.22 | 86.79 | 87.4 | 87.29 |
| UNI | 86.53 | 88.95 | 91.47 | 88.13 | 86.01 | 88.49 | 91.42 | 88.53 |
| Prov | 88.18 | 92.01 | 93.34 | 94.22 | 87.9 | 91.89 | 93.25 | 94.01 |
| **mm-PT aug** | | | | | | | | |
| Dino | 86.02 | 85.18 | 84.41 | 85.34 | 85.72 | 83.93 | 84.55 | 85.76 |
| Dinov2 | 77.81 | 81.94 | 80.22 | 81.31 | 77.92 | 81.45 | 80.88 | 82.08 |
| UNI | 84.0 | 84.14 | 85.37 | 81.54 | 83.44 | 83.33 | 85.38 | 82.76 |
| Prov | 86.69 | 85.63 | 84.62 | 90.42 | 86.69 | 85.68 | 83.76 | 90.65 |

Table 23: Performance for balanced accuracy and Cohen´s kappa for **OrganCMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 75.48 | 78.67 | 79.98 | 78.7 | 96.41 | 96.89 | 97.38 | 97.0 |
| Dino LightGBM | 74.34 | 79.08 | 79.66 | 78.08 | 96.94 | 97.87 | 97.92 | 97.65 |
| Dino RF | 58.42 | 65.39 | 66.67 | 65.84 | 94.17 | 95.81 | 95.93 | 95.63 |
| Dino KNN | 70.69 | 76.64 | 77.7 | 74.82 | - | - | - | - |
| Dinov2 SVM | 74.43 | 77.16 | 77.13 | 77.49 | 96.05 | 96.56 | 96.83 | 96.96 |
| Dinov2 LightGBM | 70.64 | 75.86 | 74.9 | 75.16 | 95.99 | 97.04 | 97.17 | 97.29 |
| Dinov2 RF | 53.59 | 58.5 | 59.44 | 61.36 | 91.43 | 93.78 | 94.02 | 94.51 |
| Dinov2 KNN | 61.91 | 71.21 | 70.36 | 71.2 | - | - | - | - |
| UNI SVM | 75.21 | 78.33 | 78.55 | 78.14 | 96.47 | 97.24 | 97.19 | 97.02 |
| UNI LightGBM | 72.65 | 76.13 | 76.59 | 76.72 | 96.58 | 97.54 | 97.28 | 97.33 |
| UNI RF | 60.85 | 64.72 | 67.25 | 66.34 | 93.05 | 95.43 | 95.73 | 95.3 |
| UNI KNN | 66.65 | 72.73 | 71.07 | 71.27 | - | - | - | - |
| Prov SVM | 76.2 | 79.59 | 79.65 | 78.23 | 96.57 | 97.44 | 97.34 | 97.06 |
| Prov LightGBM | 73.59 | 77.72 | 76.36 | 75.81 | 96.64 | 97.75 | 97.4 | 97.27 |
| Prov RF | 59.56 | 66.34 | 66.49 | 66.78 | 93.48 | 95.77 | 95.52 | 95.63 |
| Prov KNN | 68.2 | 74.32 | 72.8 | 71.1 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 74.83 | 78.96 | 80.58 | 79.49 | 97.09 | 97.9 | 98.07 | 97.9 |
| Dinov2 | 71.67 | 76.36 | 76.92 | 76.74 | 96.25 | 97.34 | 97.56 | 97.6 |
| UNI | 72.54 | 77.76 | 76.58 | 77.51 | 96.73 | 97.72 | 97.35 | 97.39 |
| Prov | 73.37 | 78.57 | 78.54 | 77.48 | 96.79 | 97.9 | 97.76 | 97.54 |
| **mm-PT** | | | | | | | | |
| Dino | 70.23 | 77.97 | 79.25 | 78.85 | 95.58 | 97.44 | 97.55 | 97.46 |
| Dinov2 | 70.17 | 73.81 | 75.52 | 78.11 | 95.45 | 96.48 | 96.66 | 97.0 |
| UNI | 72.32 | 76.23 | 80.37 | 77.36 | 95.7 | 96.88 | 97.64 | 97.22 |
| Prov | 75.38 | 80.23 | 82.62 | 83.27 | 96.69 | 97.86 | 97.9 | 98.29 |
| **mm-PT aug** | | | | | | | | |
| Dino | 72.79 | 74.71 | 74.1 | 70.24 | 96.01 | 96.79 | 97.02 | 95.69 |
| Dinov2 | 65.15 | 69.46 | 69.56 | 73.35 | 94.65 | 95.72 | 96.12 | 96.61 |
| UNI | 69.84 | 73.73 | 74.75 | 74.11 | 95.39 | 96.85 | 97.0 | 97.08 |
| Prov | 75.38 | 75.99 | 76.38 | 78.81 | 96.6 | 97.26 | 97.28 | 97.83 |

Table 24: Performance for accuracy and AUC for **OrganSMNIST** for every resolution, every backbone and every training scheme.

| Methods | Balanced Accuaracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 70.85 | 74.32 | 75.23 | 73.53 | 71.89 | 75.62 | 77.1 | 75.63 |
| Dino LightGBM | 68.71 | 74.39 | 73.6 | 71.98 | 70.49 | 76.03 | 76.67 | 74.86 |
| Dino RF | 48.69 | 56.57 | 56.92 | 56.12 | 50.56 | 59.37 | 61.01 | 60.07 |
| Dino KNN | 65.08 | 71.83 | 72.74 | 70.06 | 66.14 | 73.22 | 74.46 | 71.19 |
| Dinov2 SVM | 69.05 | 72.15 | 71.6 | 71.65 | 70.63 | 73.85 | 73.81 | 74.22 |
| Dinov2 LightGBM | 64.13 | 70.04 | 68.78 | 68.86 | 66.15 | 72.28 | 71.19 | 71.47 |
| Dinov2 RF | 42.97 | 48.01 | 48.42 | 50.84 | 44.48 | 50.85 | 52.06 | 54.47 |
| Dinov2 KNN | 55.21 | 65.7 | 64.53 | 64.91 | 55.62 | 66.91 | 66.01 | 66.91 |
| UNI SVM | 70.55 | 73.51 | 73.08 | 72.61 | 71.58 | 75.22 | 75.44 | 74.98 |
| UNI LightGBM | 66.86 | 70.2 | 70.03 | 70.28 | 68.53 | 72.6 | 73.1 | 73.28 |
| UNI RF | 51.87 | 55.32 | 57.41 | 56.54 | 54.0 | 58.64 | 61.85 | 60.77 |
| UNI KNN | 60.73 | 67.36 | 65.01 | 65.59 | 61.45 | 68.73 | 66.79 | 67.07 |
| Prov SVM | 71.09 | 74.45 | 74.62 | 73.26 | 72.66 | 76.65 | 76.72 | 75.09 |
| Prov LightGBM | 68.0 | 71.89 | 69.7 | 69.73 | 69.6 | 74.44 | 72.84 | 72.24 |
| Prov RF | 49.97 | 56.73 | 56.45 | 57.41 | 52.23 | 60.6 | 60.91 | 61.28 |
| Prov KNN | 62.56 | 68.75 | 67.62 | 66.47 | 63.24 | 70.51 | 68.83 | 66.88 |
| **Linear Probing** | | | | | | | | |
| Dino | 70.2 | 74.86 | 75.52 | 74.29 | 71.14 | 75.96 | 77.76 | 76.54 |
| Dinov2 | 65.84 | 71.35 | 71.61 | 70.98 | 67.4 | 72.92 | 73.58 | 73.35 |
| UNI | 67.16 | 72.81 | 70.45 | 71.29 | 68.45 | 74.54 | 73.15 | 74.21 |
| Prov | 68.14 | 73.34 | 73.12 | 72.45 | 69.43 | 75.46 | 75.41 | 74.2 |
| **mm-PT** | | | | | | | | |
| Dino | 66.35 | 74.12 | 74.4 | 74.04 | 66.04 | 74.81 | 76.26 | 75.8 |
| Dinov2 | 66.54 | 69.38 | 70.11 | 72.5 | 65.93 | 69.99 | 71.89 | 74.92 |
| UNI | 67.76 | 72.14 | 75.16 | 71.15 | 68.33 | 72.85 | 77.51 | 73.96 |
| Prov | 70.89 | 76.6 | 77.79 | 78.1 | 71.84 | 77.43 | 80.13 | 80.84 |
| **mm-PT aug** | | | | | | | | |
| Dino | 67.4 | 70.49 | 68.78 | 61.89 | 68.77 | 71.05 | 70.33 | 65.4 |
| Dinov2 | 60.23 | 64.39 | 64.57 | 66.97 | 59.97 | 64.92 | 65.24 | 69.4 |
| UNI | 66.75 | 70.46 | 69.19 | 68.16 | 65.66 | 70.01 | 71.0 | 70.29 |
| Prov | 70.99 | 72.58 | 70.87 | 72.86 | 71.8 | 72.57 | 72.85 | 75.74 |

Table 25: Performance for balanced accuracy and Cohen´s kappa for **OrganSMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 85.77 | 94.0 | 94.86 | 94.86 | 97.96 | 99.43 | 99.55 | 99.65 |
| Dino LightGBM | 85.33 | 93.84 | 95.15 | 95.81 | 98.32 | 99.62 | 99.71 | 99.72 |
| Dino RF | 78.22 | 88.16 | 90.46 | 90.18 | 96.46 | 98.61 | 99.04 | 99.02 |
| Dino KNN | 80.54 | 90.39 | 93.72 | 94.32 | - | - | - | - |
| Dinov2 SVM | 87.41 | 91.62 | 93.23 | 93.79 | 98.35 | 99.15 | 99.35 | 99.42 |
| Dinov2 LightGBM | 86.94 | 91.57 | 93.26 | 93.4 | 98.69 | 99.31 | 99.45 | 99.36 |
| Dinov2 RF | 76.82 | 85.14 | 86.74 | 86.62 | 96.42 | 98.22 | 98.65 | 98.62 |
| Dinov2 KNN | 82.16 | 87.79 | 89.5 | 90.22 | - | - | - | - |
| UNI SVM | 88.8 | 95.61 | 95.5 | 95.89 | 98.55 | 99.64 | 99.51 | 99.43 |
| UNI LightGBM | 87.56 | 95.49 | 96.13 | 96.71 | 98.77 | 99.77 | 99.48 | 99.45 |
| UNI RF | 82.53 | 92.45 | 95.35 | 96.75 | 97.38 | 99.47 | 99.48 | 99.78 |
| UNI KNN | 82.66 | 94.69 | 96.66 | 96.57 | - | - | - | - |
| Prov SVM | 88.72 | 94.83 | 96.43 | 95.31 | 98.53 | 99.39 | 99.53 | 99.41 |
| Prov LightGBM | 88.25 | 95.31 | 96.71 | 96.98 | 98.86 | 99.71 | 99.72 | 99.55 |
| Prov RF | 83.29 | 91.45 | 95.84 | 96.07 | 97.71 | 99.35 | 99.68 | 99.71 |
| Prov KNN | 83.16 | 93.26 | 95.91 | 96.35 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 84.74 | 94.01 | 94.09 | 96.07 | 97.95 | 99.6 | 99.63 | 99.75 |
| Dinov2 | 87.31 | 91.23 | 92.79 | 93.33 | 98.61 | 99.27 | 99.42 | 99.47 |
| UNI | 88.89 | 95.75 | 94.72 | 95.25 | 98.93 | 99.75 | 99.55 | 99.5 |
| Prov | 87.83 | 94.29 | 96.6 | 96.1 | 98.92 | 99.54 | 99.63 | 99.73 |
| **mm-PT** | | | | | | | | |
| Dino | 76.91 | 86.13 | 92.13 | 94.58 | 96.57 | 98.47 | 99.2 | 99.59 |
| Dinov2 | 74.16 | 80.01 | 88.61 | 88.9 | 95.89 | 97.35 | 98.48 | 98.78 |
| UNI | 81.62 | 86.94 | 92.72 | 93.57 | 97.0 | 98.39 | 99.38 | 99.34 |
| Prov | 83.45 | 90.49 | 95.13 | 96.81 | 97.86 | 99.13 | 99.58 | 99.81 |
| **mm-PT aug** | | | | | | | | |
| Dino | 65.29 | 80.65 | 87.02 | 90.42 | 93.1 | 97.39 | 98.95 | 99.18 |
| Dinov2 | 62.06 | 80.78 | 76.81 | 74.86 | 91.98 | 96.81 | 95.88 | 93.97 |
| UNI | 67.33 | 80.91 | 85.43 | 89.47 | 93.75 | 97.9 | 98.52 | 99.24 |
| Prov | 77.55 | 85.61 | 92.99 | 93.45 | 95.71 | 98.37 | 99.6 | 99.44 |

Table 26: Performance for accuracy and AUC for **PathMNIST** for every resolution, every backbone and every training scheme.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 81.5 | 91.66 | 92.88 | 93.21 | 83.66 | 93.1 | 94.09 | 94.09 |
| Dino LightGBM | 81.11 | 91.71 | 92.8 | 93.88 | 83.15 | 92.93 | 94.43 | 95.18 |
| Dino RF | 72.34 | 84.02 | 87.16 | 87.05 | 74.78 | 86.36 | 89.02 | 88.71 |
| Dino KNN | 77.14 | 88.48 | 92.08 | 93.11 | 77.7 | 88.98 | 92.79 | 93.48 |
| Dinov2 SVM | 84.09 | 88.63 | 90.76 | 91.26 | 85.54 | 90.37 | 92.23 | 92.87 |
| Dinov2 LightGBM | 84.14 | 88.65 | 91.06 | 91.07 | 85.0 | 90.33 | 92.26 | 92.42 |
| Dinov2 RF | 72.57 | 80.57 | 83.69 | 84.34 | 73.29 | 82.89 | 84.78 | 84.61 |
| Dinov2 KNN | 78.99 | 84.9 | 87.86 | 88.66 | 79.55 | 86.01 | 87.98 | 88.79 |
| UNI SVM | 85.7 | 94.12 | 94.28 | 94.11 | 87.14 | 94.96 | 94.84 | 95.28 |
| UNI LightGBM | 84.44 | 94.08 | 95.06 | 95.81 | 85.72 | 94.82 | 95.56 | 96.23 |
| UNI RF | 78.3 | 90.03 | 93.37 | 95.94 | 79.84 | 91.33 | 94.66 | 96.28 |
| UNI KNN | 79.58 | 93.33 | 95.29 | 95.1 | 80.12 | 93.91 | 96.16 | 96.06 |
| Prov SVM | 85.73 | 93.27 | 95.45 | 94.21 | 87.05 | 94.07 | 95.91 | 94.62 |
| Prov LightGBM | 85.64 | 93.95 | 95.71 | 95.88 | 86.52 | 94.61 | 96.23 | 96.53 |
| Prov RF | 79.32 | 87.27 | 94.41 | 94.63 | 80.73 | 90.16 | 95.22 | 95.49 |
| Prov KNN | 79.83 | 91.41 | 94.61 | 94.88 | 80.7 | 92.27 | 95.3 | 95.81 |
| **Linear Probing** | | | | | | | | |
| Dino | 79.65 | 91.87 | 91.46 | 94.44 | 82.46 | 93.12 | 93.21 | 95.49 |
| Dinov2 | 84.14 | 87.97 | 90.48 | 90.92 | 85.43 | 89.93 | 91.72 | 92.34 |
| UNI | 85.94 | 94.52 | 93.75 | 93.77 | 87.24 | 95.12 | 93.95 | 94.55 |
| Prov | 84.81 | 92.77 | 95.6 | 94.83 | 86.03 | 93.45 | 96.1 | 95.52 |
| **mm-PT** | | | | | | | | |
| Dino | 72.92 | 84.32 | 90.65 | 93.28 | 73.58 | 84.15 | 90.98 | 93.78 |
| Dinov2 | 70.07 | 75.41 | 84.83 | 86.4 | 70.38 | 77.07 | 86.91 | 87.24 |
| UNI | 77.6 | 84.15 | 91.22 | 92.5 | 78.91 | 85.01 | 91.64 | 92.62 |
| Prov | 80.24 | 87.75 | 93.56 | 95.73 | 81.04 | 89.07 | 94.4 | 96.34 |
| **mm-PT aug** | | | | | | | | |
| Dino | 59.96 | 74.55 | 81.69 | 85.3 | 60.53 | 77.79 | 85.05 | 88.94 |
| Dinov2 | 58.13 | 75.28 | 71.99 | 67.31 | 56.28 | 77.82 | 73.49 | 71.16 |
| UNI | 61.52 | 76.99 | 80.37 | 84.78 | 62.41 | 78.08 | 83.21 | 87.87 |
| Prov | 71.86 | 81.32 | 92.32 | 91.49 | 74.17 | 83.47 | 91.97 | 92.47 |

Table 27: Performance for balanced accuracy and Cohen´s kappa for **PathMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 |
| **Classifier** | | | | | | | | |
| Dino SVM | 86.38 | 88.14 | 89.58 | 91.19 | 97.77 | 97.87 | 98.81 | 98.63 |
| Dino LightGBM | 84.29 | 85.1 | 88.94 | 89.58 | 96.28 | 97.1 | 98.46 | 98.6 |
| Dino RF | 83.49 | 80.77 | 86.38 | 85.42 | 94.56 | 95.04 | 97.37 | 97.41 |
| Dino KNN | 85.74 | 87.82 | 90.54 | 89.74 | - | - | - | - |
| Dinov2 SVM | 84.78 | 87.98 | 90.22 | 91.83 | 95.92 | 97.69 | 98.43 | 98.81 |
| Dinov2 LightGBM | 85.26 | 86.54 | 86.38 | 88.62 | 94.97 | 97.13 | 97.14 | 97.6 |
| Dinov2 RF | 84.29 | 83.33 | 81.57 | 82.05 | 94.17 | 95.45 | 93.8 | 93.83 |
| Dinov2 KNN | 87.82 | 87.18 | 88.14 | 88.46 | - | - | - | - |
| UNI SVM | 86.38 | 85.58 | 89.1 | 89.9 | 96.62 | 97.54 | 98.44 | 99.21 |
| UNI LightGBM | 85.42 | 84.62 | 86.22 | 88.94 | 95.14 | 97.13 | 97.94 | 98.66 |
| UNI RF | 81.41 | 82.05 | 82.37 | 83.49 | 91.87 | 95.78 | 96.03 | 97.06 |
| UNI KNN | 83.17 | 87.02 | 90.22 | 91.51 | - | - | - | - |
| Prov SVM | 84.62 | 86.22 | 89.1 | 90.87 | 95.94 | 97.24 | 98.87 | 98.9 |
| Prov LightGBM | 82.69 | 85.42 | 87.34 | 90.06 | 94.9 | 97.06 | 98.26 | 98.7 |
| Prov RF | 80.77 | 83.01 | 83.65 | 88.46 | 92.75 | 95.34 | 96.26 | 96.98 |
| Prov KNN | 83.01 | 89.1 | 90.54 | 90.87 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 87.18 | 87.02 | 90.71 | 91.51 | 97.47 | 97.62 | 98.92 | 98.75 |
| Dinov2 | 84.94 | 87.34 | 87.66 | 87.34 | 94.73 | 97.47 | 98.13 | 98.28 |
| UNI | 85.26 | 86.06 | 85.26 | 90.06 | 96.25 | 97.02 | 97.85 | 98.78 |
| Prov | 83.17 | 84.62 | 87.5 | 90.22 | 95.53 | 97.19 | 98.54 | 98.71 |
| **mm-PT** | | | | | | | | |
| Dino | 72.28 | 83.65 | 87.66 | 91.19 | 94.63 | 93.36 | 96.96 | 97.71 |
| Dinov2 | 85.58 | 83.01 | 82.05 | 87.82 | 95.44 | 96.41 | 96.06 | 94.55 |
| UNI | 77.56 | 81.73 | 89.74 | 89.1 | 93.21 | 96.64 | 97.45 | 97.16 |
| Prov | 86.06 | 86.86 | 88.14 | 89.42 | 95.67 | 95.7 | 97.3 | 98.32 |
| **mm-PT aug** | | | | | | | | |
| Dino | 84.94 | 81.57 | 82.69 | 77.24 | 94.96 | 93.77 | 94.82 | 92.97 |
| Dinov2 | 86.22 | 79.17 | 81.57 | 80.61 | 94.44 | 94.38 | 92.5 | 93.89 |
| UNI | 86.7 | 79.97 | 84.46 | 84.78 | 94.28 | 95.13 | 95.44 | 95.28 |
| Prov | 83.01 | 82.21 | 87.34 | 86.86 | 95.52 | 94.72 | 96.27 | 98.63 |

Table 28: Performance for accuracy and AUC for **PneumoniaMNIST** for every resolution, every backbone, and every training scheme.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 81.92 | 84.27 | 86.2 | 88.33 | 68.72 | 73.04 | 76.51 | 80.3 |
| Dino LightGBM | 79.49 | 80.38 | 85.51 | 86.2 | 63.77 | 65.65 | 75.07 | 76.51 |
| Dino RF | 78.76 | 74.87 | 82.01 | 80.9 | 62.03 | 54.8 | 68.78 | 66.51 |
| Dino KNN | 82.69 | 84.87 | 87.91 | 87.18 | 68.24 | 72.86 | 78.98 | 77.26 |
| Dinov2 SVM | 80.21 | 84.32 | 87.05 | 89.1 | 65.04 | 72.8 | 78.04 | 81.77 |
| Dinov2 LightGBM | 81.2 | 82.56 | 82.09 | 84.91 | 66.48 | 69.4 | 68.84 | 74.21 |
| Dinov2 RF | 80.34 | 78.46 | 76.11 | 76.41 | 64.43 | 61.55 | 57.05 | 57.97 |
| Dinov2 KNN | 85.13 | 83.85 | 85.38 | 85.9 | 73.0 | 71.22 | 73.67 | 74.47 |
| UNI SVM | 82.26 | 80.85 | 85.56 | 86.71 | 68.95 | 66.73 | 75.36 | 77.32 |
| UNI LightGBM | 81.15 | 79.83 | 81.79 | 85.34 | 66.7 | 64.51 | 68.38 | 74.98 |
| UNI RF | 76.41 | 76.58 | 76.75 | 78.42 | 57.12 | 58.13 | 58.72 | 61.75 |
| UNI KNN | 79.36 | 83.89 | 88.16 | 89.7 | 62.06 | 70.99 | 78.54 | 81.42 |
| Prov SVM | 80.0 | 81.97 | 85.73 | 87.99 | 64.64 | 68.5 | 75.45 | 79.59 |
| Prov LightGBM | 77.61 | 80.81 | 83.21 | 86.84 | 59.93 | 66.45 | 71.09 | 77.66 |
| Prov RF | 75.3 | 77.69 | 78.55 | 84.79 | 55.22 | 60.45 | 62.08 | 73.87 |
| Prov KNN | 79.23 | 86.5 | 88.25 | 88.5 | 61.73 | 75.84 | 79.13 | 79.81 |
| **Linear Probing** | | | | | | | | |
| Dino | 83.08 | 82.86 | 87.78 | 88.76 | 70.75 | 70.36 | 79.21 | 81.05 |
| Dinov2 | 80.94 | 83.46 | 83.63 | 83.29 | 65.82 | 71.25 | 71.87 | 71.14 |
| UNI | 80.77 | 81.58 | 80.68 | 86.84 | 66.18 | 67.99 | 66.11 | 77.66 |
| Prov | 78.25 | 79.74 | 83.42 | 87.05 | 61.15 | 64.44 | 71.48 | 78.04 |
| **mm-PT** | | | | | | | | |
| Dino | 63.03 | 78.72 | 84.15 | 88.85 | 30.59 | 62.22 | 72.18 | 80.51 |
| Dinov2 | 81.03 | 77.35 | 76.5 | 87.18 | 66.85 | 60.15 | 58.05 | 74.11 |
| UNI | 70.09 | 75.64 | 86.67 | 85.81 | 45.63 | 56.82 | 77.02 | 75.5 |
| Prov | 82.35 | 83.16 | 84.53 | 86.24 | 68.51 | 70.29 | 73.19 | 76.26 |
| **mm-PT aug** | | | | | | | | |
| Dino | 80.17 | 75.68 | 77.61 | 70.09 | 65.25 | 56.64 | 59.93 | 45.28 |
| Dinov2 | 82.74 | 72.39 | 76.54 | 74.74 | 69.01 | 50.19 | 57.45 | 54.47 |
| UNI | 82.69 | 73.38 | 79.53 | 80.3 | 69.74 | 52.24 | 64.04 | 65.11 |
| Prov | 77.95 | 76.45 | 84.23 | 82.74 | 60.67 | 58.23 | 71.71 | 70.02 |

Table 29: Performance for balanced accuracy and Cohen´s kappa for **PneumoniaMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 |
| **Classifier** | | | | | | | | |
| Dino SVM | 50.25 | 54.0 | 54.75 | 60.25 | 72.54 | 74.51 | 77.6 | 83.03 |
| Dino LightGBM | 51.75 | 53.25 | 59.5 | 62.75 | 73.48 | 75.91 | 82.2 | 85.25 |
| Dino RF | 51.75 | 56.5 | 59.0 | 60.5 | 71.99 | 72.05 | 79.13 | 84.18 |
| Dino KNN | 48.0 | 52.5 | 51.0 | 59.0 | - | - | - | - |
| Dinov2 SVM | 50.25 | 54.75 | 56.25 | 64.5 | 72.06 | 76.02 | 82.07 | 86.1 |
| Dinov2 LightGBM | 48.5 | 52.75 | 57.75 | 64.75 | 69.6 | 77.08 | 82.28 | 85.18 |
| Dinov2 RF | 50.5 | 52.25 | 53.25 | 58.75 | 70.48 | 75.74 | 81.03 | 82.56 |
| Dinov2 KNN | 51.0 | 49.0 | 49.25 | 56.0 | - | - | - | - |
| UNI SVM | 52.75 | 54.0 | 60.0 | 60.25 | 72.65 | 75.92 | 80.13 | 81.98 |
| UNI LightGBM | 49.75 | 54.5 | 58.0 | 61.5 | 71.53 | 77.14 | 79.0 | 82.08 |
| UNI RF | 51.25 | 55.0 | 58.0 | 57.75 | 69.98 | 76.01 | 76.88 | 80.49 |
| UNI KNN | 45.75 | 48.75 | 52.25 | 53.0 | - | - | - | - |
| Prov SVM | 51.5 | 56.75 | 58.25 | 62.0 | 72.64 | 77.61 | 81.81 | 84.72 |
| Prov LightGBM | 46.75 | 55.0 | 57.75 | 62.75 | 70.41 | 77.4 | 83.12 | 86.08 |
| Prov RF | 52.75 | 53.75 | 61.0 | 62.5 | 70.26 | 76.66 | 81.11 | 84.48 |
| Prov KNN | 46.0 | 51.75 | 53.0 | 59.0 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 53.25 | 57.75 | 59.0 | 61.0 | 72.14 | 77.61 | 83.18 | 85.85 |
| Dinov2 | 51.0 | 56.25 | 57.25 | 64.25 | 70.41 | 78.08 | 83.45 | 85.85 |
| UNI | 54.25 | 56.25 | 59.25 | 63.75 | 72.56 | 78.35 | 81.88 | 84.41 |
| Prov | 52.75 | 56.75 | 62.0 | 62.0 | 72.3 | 77.83 | 84.03 | 85.55 |
| **mm-PT** | | | | | | | | |
| Dino | 48.25 | 48.75 | 49.25 | 49.5 | 69.75 | 69.9 | 71.2 | 73.98 |
| Dinov2 | 53.25 | 49.75 | 46.75 | 49.25 | 72.25 | 71.83 | 69.8 | 72.54 |
| UNI | 49.25 | 45.0 | 48.25 | 49.0 | 69.14 | 70.36 | 72.29 | 71.67 |
| Prov | 48.5 | 50.25 | 51.5 | 51.25 | 66.54 | 70.9 | 72.02 | 72.84 |
| **mm-PT aug** | | | | | | | | |
| Dino | 51.5 | 48.5 | 51.25 | 52.5 | 70.9 | 69.79 | 73.12 | 74.54 |
| Dinov2 | 54.0 | 49.5 | 51.75 | 52.75 | 70.71 | 71.22 | 70.78 | 72.0 |
| UNI | 47.5 | 49.5 | 47.75 | 50.0 | 67.68 | 70.56 | 69.45 | 70.92 |
| Prov | 51.0 | 50.75 | 51.0 | 54.5 | 70.68 | 70.92 | 72.35 | 76.6 |

Table 30: Performance for accuracy and AUC for **RetinaMNIST** for every resolution, every backbone, and every training scheme.

| Methods | Balanced Accuracy | | | | Cohen´s Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 33.14 | 36.91 | 41.47 | 44.6 | 25.47 | 28.92 | 32.56 | 41.42 |
| Dino LightGBM | 34.39 | 34.6 | 43.37 | 46.76 | 28.58 | 29.39 | 40.0 | 45.67 |
| Dino RF | 33.93 | 35.53 | 37.81 | 40.13 | 28.37 | 32.56 | 36.41 | 39.03 |
| Dino KNN | 31.2 | 33.68 | 33.79 | 40.93 | 20.83 | 27.55 | 26.7 | 37.57 |
| Dinov2 SVM | 33.96 | 37.13 | 40.14 | 47.48 | 23.87 | 31.64 | 33.42 | 47.3 |
| Dinov2 LightGBM | 33.9 | 38.06 | 42.51 | 49.85 | 24.33 | 29.29 | 36.59 | 47.37 |
| Dinov2 RF | 31.18 | 31.34 | 33.1 | 40.1 | 24.42 | 24.56 | 25.75 | 36.7 |
| Dinov2 KNN | 32.76 | 31.04 | 33.13 | 38.86 | 25.24 | 20.49 | 20.94 | 32.8 |
| UNI SVM | 34.76 | 36.04 | 44.56 | 45.22 | 30.18 | 31.42 | 40.13 | 41.69 |
| UNI LightGBM | 33.21 | 38.35 | 42.32 | 47.61 | 26.76 | 33.36 | 38.06 | 43.86 |
| UNI RF | 33.32 | 37.03 | 38.39 | 38.61 | 28.93 | 33.88 | 37.17 | 36.78 |
| UNI KNN | 27.01 | 32.42 | 34.61 | 33.2 | 19.07 | 24.41 | 27.26 | 29.45 |
| Prov SVM | 33.6 | 40.22 | 43.05 | 47.06 | 27.39 | 34.66 | 38.33 | 43.83 |
| Prov LightGBM | 29.92 | 39.27 | 42.42 | 47.55 | 22.81 | 33.98 | 38.17 | 45.39 |
| Prov RF | 33.94 | 34.44 | 41.56 | 42.98 | 30.2 | 30.23 | 40.71 | 43.88 |
| Prov KNN | 27.56 | 33.55 | 33.65 | 42.22 | 19.43 | 27.56 | 27.18 | 37.96 |
| **Linear Probing** | | | | | | | | |
| Dino | 33.53 | 41.0 | 42.98 | 46.04 | 28.98 | 36.93 | 38.97 | 42.82 |
| Dinov2 | 31.32 | 38.31 | 42.14 | 48.69 | 24.22 | 33.8 | 35.51 | 46.62 |
| UNI | 34.95 | 38.94 | 43.48 | 47.5 | 31.91 | 35.66 | 39.18 | 46.22 |
| Prov | 32.57 | 37.95 | 45.41 | 45.21 | 29.23 | 34.96 | 43.98 | 43.21 |
| **mm-PT** | | | | | | | | |
| Dino | 29.17 | 29.09 | 33.61 | 34.52 | 19.79 | 19.65 | 28.27 | 28.35 |
| Dinov2 | 38.18 | 33.79 | 29.63 | 29.95 | 34.09 | 28.04 | 21.94 | 20.09 |
| UNI | 34.07 | 21.89 | 26.69 | 27.42 | 28.43 | 4.49 | 13.49 | 16.94 |
| Prov | 27.95 | 32.83 | 34.39 | 32.43 | 20.29 | 27.6 | 29.24 | 27.98 |
| **mm-PT aug** | | | | | | | | |
| Dino | 36.29 | 34.57 | 37.79 | 34.48 | 31.08 | 28.93 | 30.26 | 29.69 |
| Dinov2 | 35.74 | 33.41 | 34.59 | 34.11 | 34.19 | 27.44 | 31.69 | 32.17 |
| UNI | 36.24 | 34.79 | 33.89 | 35.23 | 29.01 | 29.08 | 26.97 | 28.24 |
| Prov | 34.66 | 32.05 | 36.07 | 38.53 | 29.7 | 28.33 | 30.88 | 34.16 |

Table 31: Performance for balanced accuracy and Cohen´s kappa for **retinaMNIST** for every resolution, every backbone and every training scheme.

| Methods | Accuracy | | | | Area Under the ROC Curve (AUC) | | | |
|---|---|---|---|---|---|---|---|---|
| | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 | 28 × 28 | 64 × 64 | 128 × 128 | 224 × 224 |
| **Classifier** | | | | | | | | |
| Dino SVM | 57.94 | 62.84 | 63.45 | 63.65 | 87.18 | 89.94 | 90.37 | 90.34 |
| Dino LightGBM | 58.19 | 62.84 | 63.65 | 63.73 | 87.64 | 90.58 | 91.07 | 90.93 |
| Dino RF | 49.38 | 49.86 | 50.98 | 51.59 | 81.41 | 84.3 | 84.89 | 84.68 |
| Dino KNN | 51.56 | 56.1 | 57.39 | 57.12 | - | - | - | - |
| Dinov2 SVM | 59.21 | 61.36 | 62.87 | 63.4 | 87.96 | 89.23 | 89.95 | 90.31 |
| Dinov2 LightGBM | 58.07 | 60.55 | 62.11 | 62.63 | 87.83 | 89.25 | 90.07 | 90.44 |
| Dinov2 RF | 48.34 | 49.92 | 49.55 | 49.61 | 81.19 | 83.1 | 83.46 | 83.69 |
| Dinov2 KNN | 51.16 | 53.84 | 54.64 | 54.9 | - | - | - | - |
| UNI SVM | 57.22 | 63.54 | 64.41 | 63.65 | 86.83 | 90.44 | 90.7 | 90.52 |
| UNI LightGBM | 57.31 | 63.43 | 63.41 | 62.48 | 87.01 | 90.73 | 90.75 | 90.43 |
| UNI RF | 48.66 | 51.94 | 51.22 | 50.41 | 80.58 | 84.8 | 83.74 | 83.04 |
| UNI KNN | 50.17 | 56.24 | 55.09 | 54.2 | - | - | - | - |
| Prov SVM | 57.82 | 64.03 | 63.86 | 64.15 | 87.26 | 90.55 | 90.41 | 90.61 |
| Prov LightGBM | 57.71 | 63.12 | 62.71 | 62.95 | 87.42 | 90.54 | 90.32 | 90.64 |
| Prov RF | 48.69 | 50.63 | 49.68 | 49.78 | 80.32 | 84.13 | 82.16 | 82.7 |
| Prov KNN | 50.08 | 55.26 | 53.58 | 55.0 | - | - | - | - |
| **Linear Probing** | | | | | | | | |
| Dino | 57.61 | 63.2 | 63.67 | 64.03 | 87.23 | 90.57 | 90.95 | 90.96 |
| Dinov2 | 58.32 | 61.19 | 62.94 | 63.61 | 87.92 | 89.51 | 90.44 | 90.83 |
| UNI | 56.03 | 63.61 | 63.96 | 62.93 | 86.35 | 90.87 | 90.84 | 90.58 |
| Prov | 56.64 | 63.74 | 63.61 | 64.36 | 86.89 | 90.8 | 90.85 | 91.12 |
| **mm-PT** | | | | | | | | |
| Dino | 57.78 | 63.85 | 66.69 | 67.99 | 87.41 | 91.37 | 92.46 | 93.01 |
| Dinov2 | 58.37 | 61.24 | 64.05 | 66.15 | 87.42 | 89.59 | 91.32 | 92.23 |
| UNI | 58.37 | 63.71 | 66.94 | 65.72 | 87.65 | 91.14 | 92.37 | 92.14 |
| Prov | 58.49 | 67.13 | 70.73 | 71.95 | 87.98 | 92.4 | 94.22 | 94.58 |
| **mm-PT aug** | | | | | | | | |
| Dino | 51.29 | 51.05 | 47.23 | 46.85 | 82.59 | 83.14 | 79.74 | 80.29 |
| Dinov2 | 47.46 | 47.27 | 46.98 | 44.71 | 79.53 | 80.12 | 78.31 | 77.37 |
| UNI | 49.24 | 50.64 | 52.06 | 50.01 | 80.55 | 82.79 | 83.88 | 82.06 |
| Prov | 47.91 | 50.54 | 50.88 | 55.02 | 81.15 | 83.21 | 83.54 | 87.47 |

Table 32: Performance for accuracy and AUC for **TissueMNIST** across various resolutions, backbones, and training schemes.

| Methods | Balanced Accuracy | | | | Cohen's Kappa | | | |
|---|---|---|---|---|---|---|---|---|
| | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** | **28 × 28** | **64 × 64** | **128 × 128** | **224 × 224** |
| **Classifier** | | | | | | | | |
| Dino SVM | 42.85 | 50.04 | 51.61 | 51.41 | 45.59 | 52.47 | 53.32 | 53.53 |
| Dino LightGBM | 43.95 | 50.27 | 51.88 | 51.55 | 46.18 | 52.64 | 53.71 | 53.79 |
| Dino RF | 28.64 | 27.7 | 31.05 | 31.6 | 31.75 | 31.99 | 33.89 | 34.85 |
| Dino KNN | 34.27 | 41.5 | 44.02 | 42.72 | 36.46 | 43.52 | 45.37 | 44.71 |
| Dinov2 SVM | 45.53 | 48.14 | 49.93 | 50.56 | 47.45 | 50.45 | 52.5 | 53.19 |
| Dinov2 LightGBM | 44.03 | 47.31 | 49.14 | 49.68 | 46.09 | 49.56 | 51.67 | 52.38 |
| Dinov2 RF | 27.0 | 28.33 | 27.93 | 28.27 | 29.81 | 32.23 | 31.66 | 31.89 |
| Dinov2 KNN | 35.29 | 38.62 | 39.78 | 40.78 | 36.49 | 40.18 | 41.44 | 41.96 |
| UNI SVM | 41.74 | 50.89 | 52.17 | 50.9 | 44.51 | 53.39 | 54.52 | 53.5 |
| UNI LightGBM | 42.28 | 51.1 | 51.15 | 49.15 | 44.98 | 53.38 | 53.33 | 52.06 |
| UNI RF | 27.94 | 32.01 | 31.44 | 30.28 | 30.58 | 35.63 | 34.44 | 33.13 |
| UNI KNN | 32.6 | 42.63 | 40.1 | 38.39 | 34.38 | 43.93 | 41.92 | 40.5 |
| Prov SVM | 42.88 | 51.68 | 51.88 | 51.46 | 45.49 | 54.04 | 53.81 | 54.18 |
| Prov LightGBM | 43.3 | 50.78 | 49.95 | 49.71 | 45.58 | 52.93 | 52.34 | 52.71 |
| Prov RF | 27.73 | 31.81 | 29.77 | 29.58 | 30.49 | 33.7 | 32.21 | 32.29 |
| Prov KNN | 32.39 | 41.43 | 38.2 | 39.37 | 34.42 | 42.33 | 39.73 | 41.85 |
| **Linear Probing** | | | | | | | | |
| Dino | 42.48 | 51.69 | 52.6 | 52.66 | 45.12 | 53.09 | 53.65 | 54.15 |
| Dinov2 | 44.32 | 48.4 | 50.59 | 52.35 | 46.26 | 50.45 | 52.77 | 53.73 |
| UNI | 40.83 | 52.64 | 52.64 | 50.34 | 43.07 | 53.91 | 54.25 | 52.65 |
| Prov | 42.21 | 52.06 | 52.39 | 52.77 | 44.16 | 53.8 | 53.62 | 54.69 |
| **mm-PT** | | | | | | | | |
| Dino | 43.17 | 52.4 | 55.42 | 55.76 | 46.29 | 54.44 | 57.47 | 59.2 |
| Dinov2 | 43.61 | 48.92 | 53.59 | 56.34 | 46.78 | 51.08 | 54.57 | 57.35 |
| UNI | 43.65 | 53.11 | 56.64 | 54.34 | 46.36 | 54.05 | 58.16 | 56.43 |
| Prov | 44.47 | 57.08 | 61.74 | 63.08 | 47.03 | 58.59 | 63.25 | 64.59 |
| **mm-PT aug** | | | | | | | | |
| Dino | 32.57 | 33.11 | 27.35 | 25.59 | 35.96 | 35.78 | 29.06 | 27.26 |
| Dinov2 | 28.19 | 27.73 | 27.96 | 24.51 | 29.73 | 30.64 | 29.18 | 24.12 |
| UNI | 29.11 | 31.52 | 33.97 | 31.25 | 32.52 | 35.21 | 37.01 | 34.25 |
| Prov | 29.78 | 33.26 | 33.26 | 43.64 | 30.79 | 36.08 | 34.98 | 43.2 |

Table 33: Performance for balanced accuracy and Cohen´s kappa for **TissueMNIST** across various resolutions, backbones, and training schemes.

## A.3 Plots for Performance on MedMNIST-C

This section of the appendix has every plot for every metric, every backbone, every training method and every resolution on MedMNIST-C.

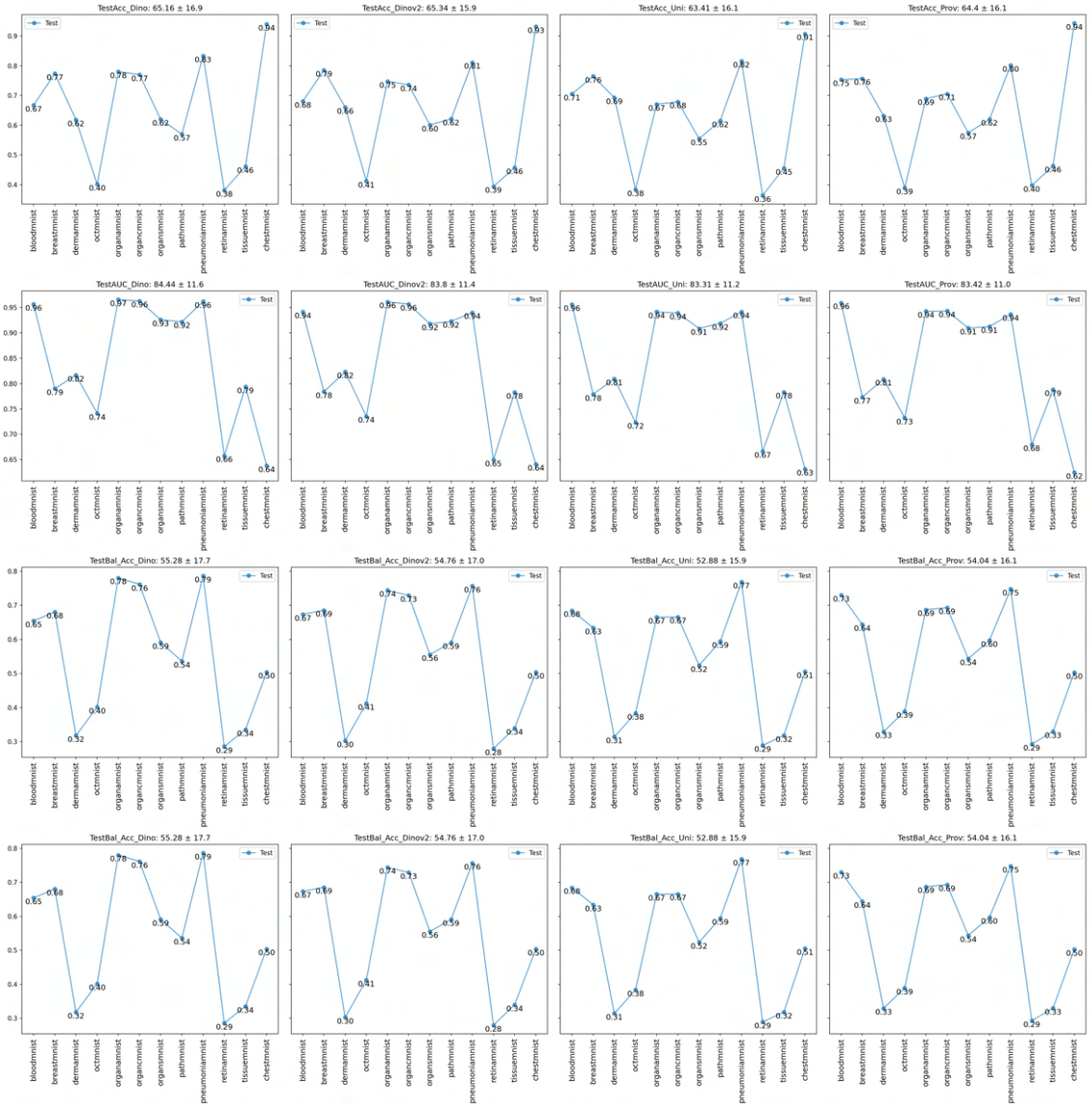


Figure 55: Every metric for 28×28 resolution with SVM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

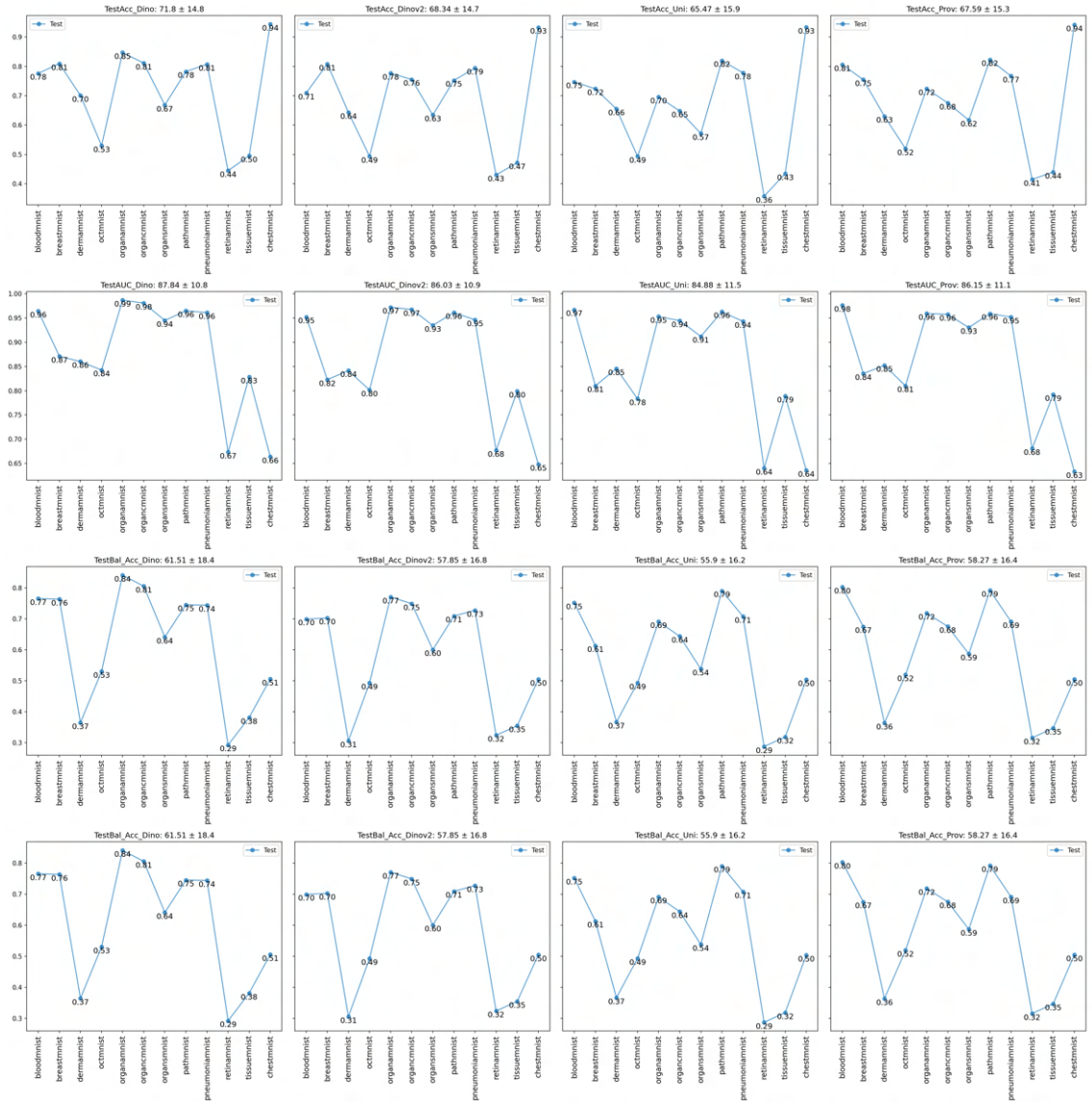


Figure 56: Every metric for 64×64 resolution with SVM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

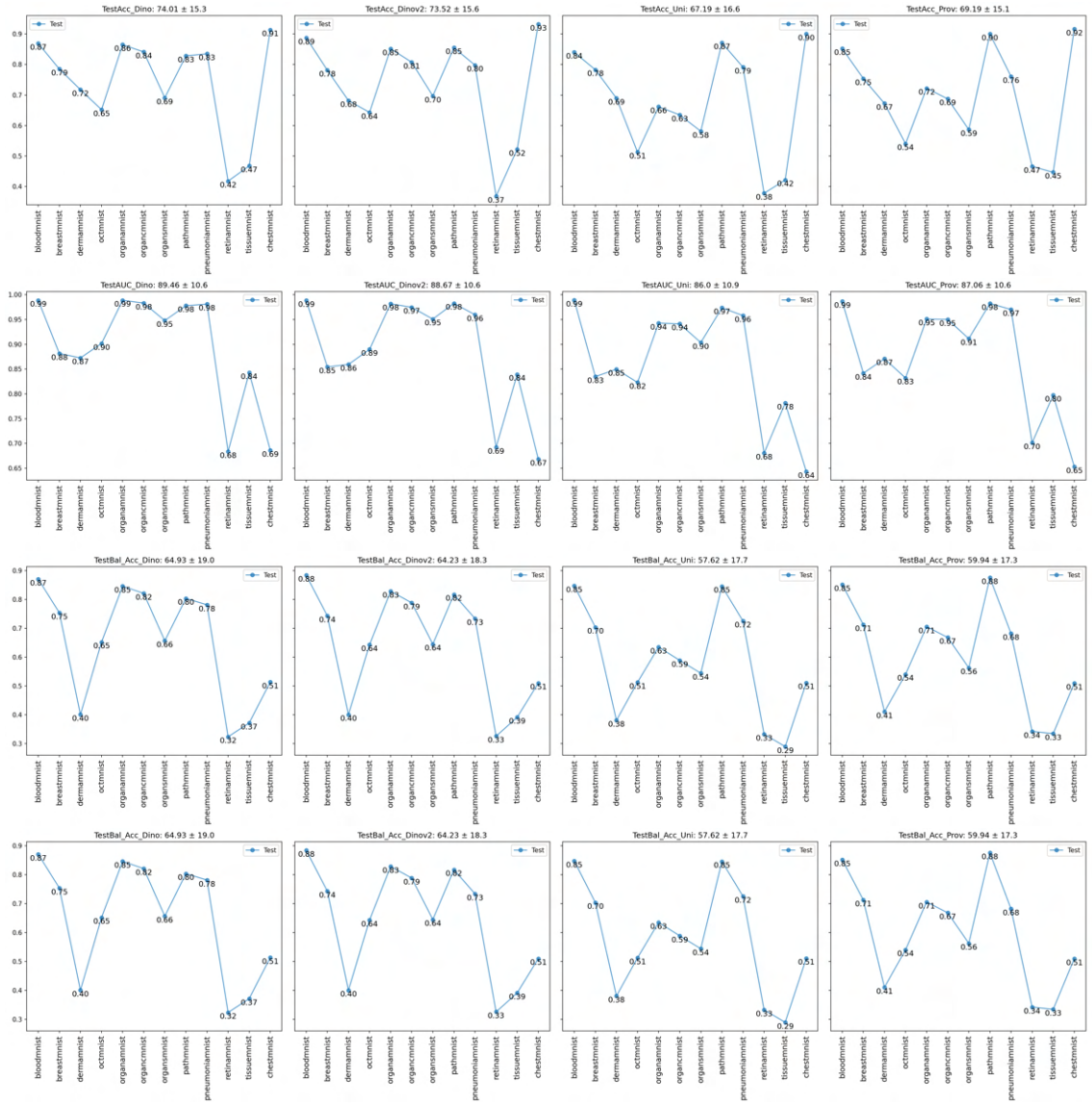


Figure 57: Every metric for 128×128 resolution with SVM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

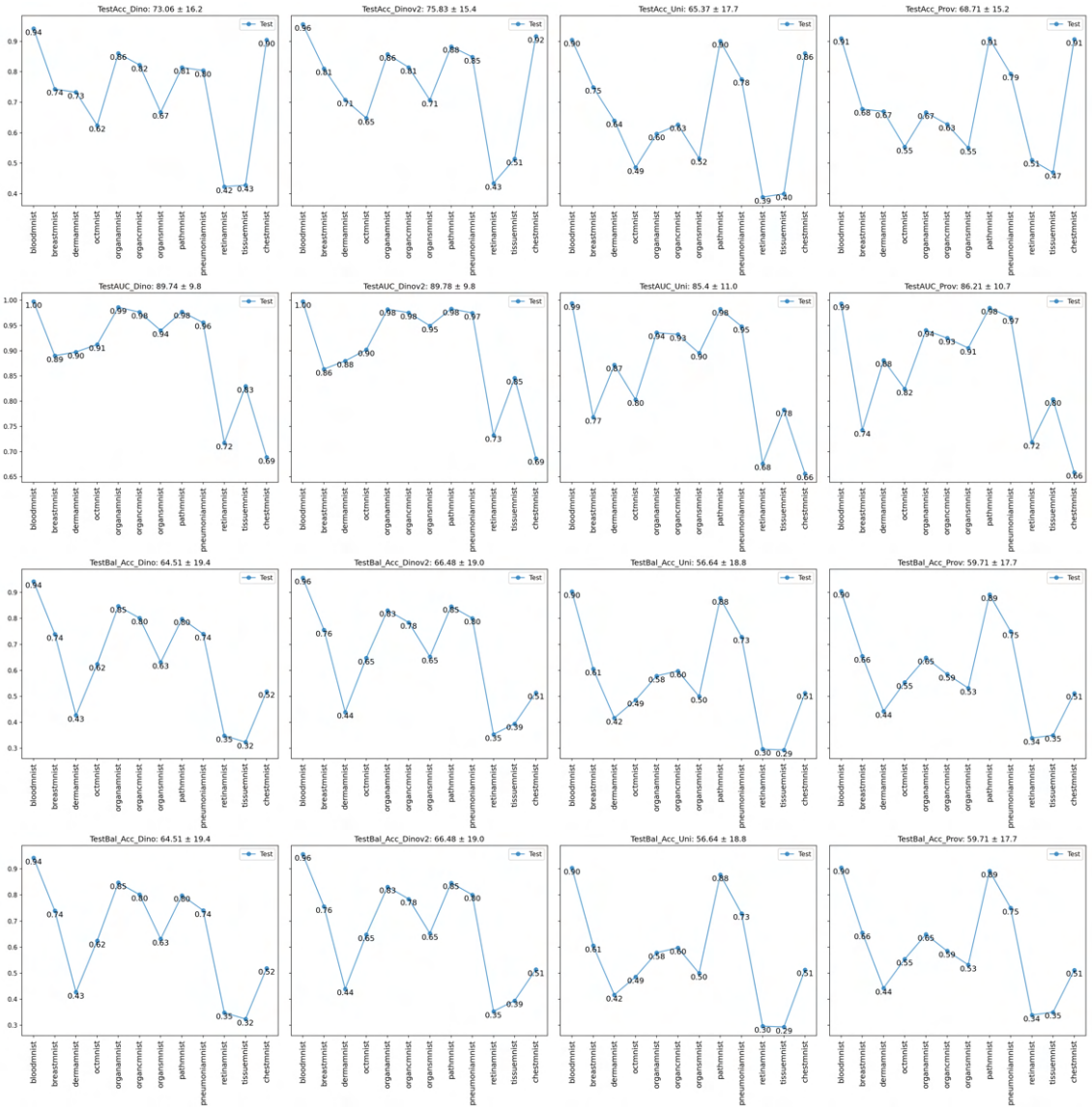


Figure 58: Every metric for 224×224 resolution with SVM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

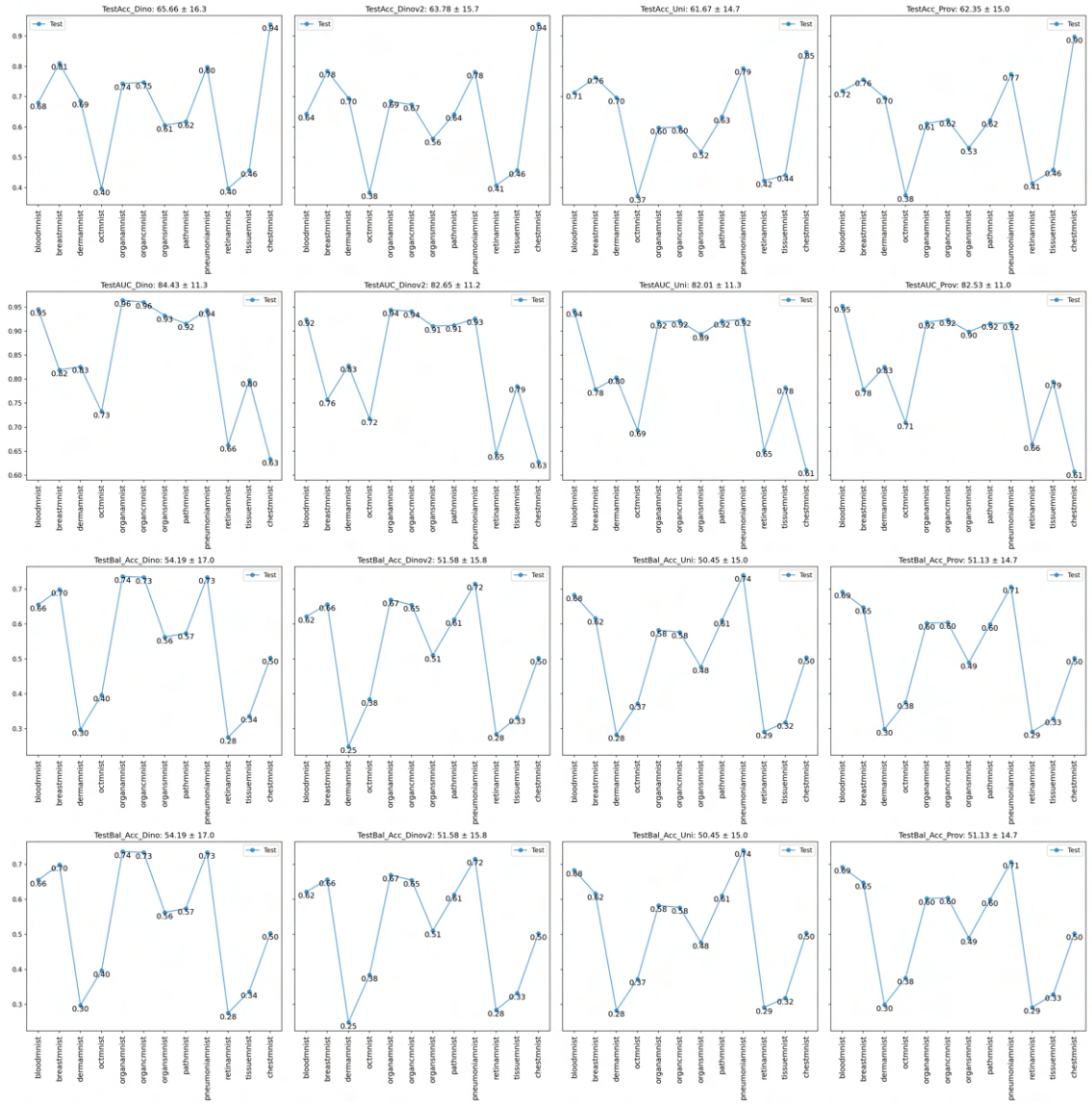


Figure 59: Every metric for 28×28 resolution with LightGBM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

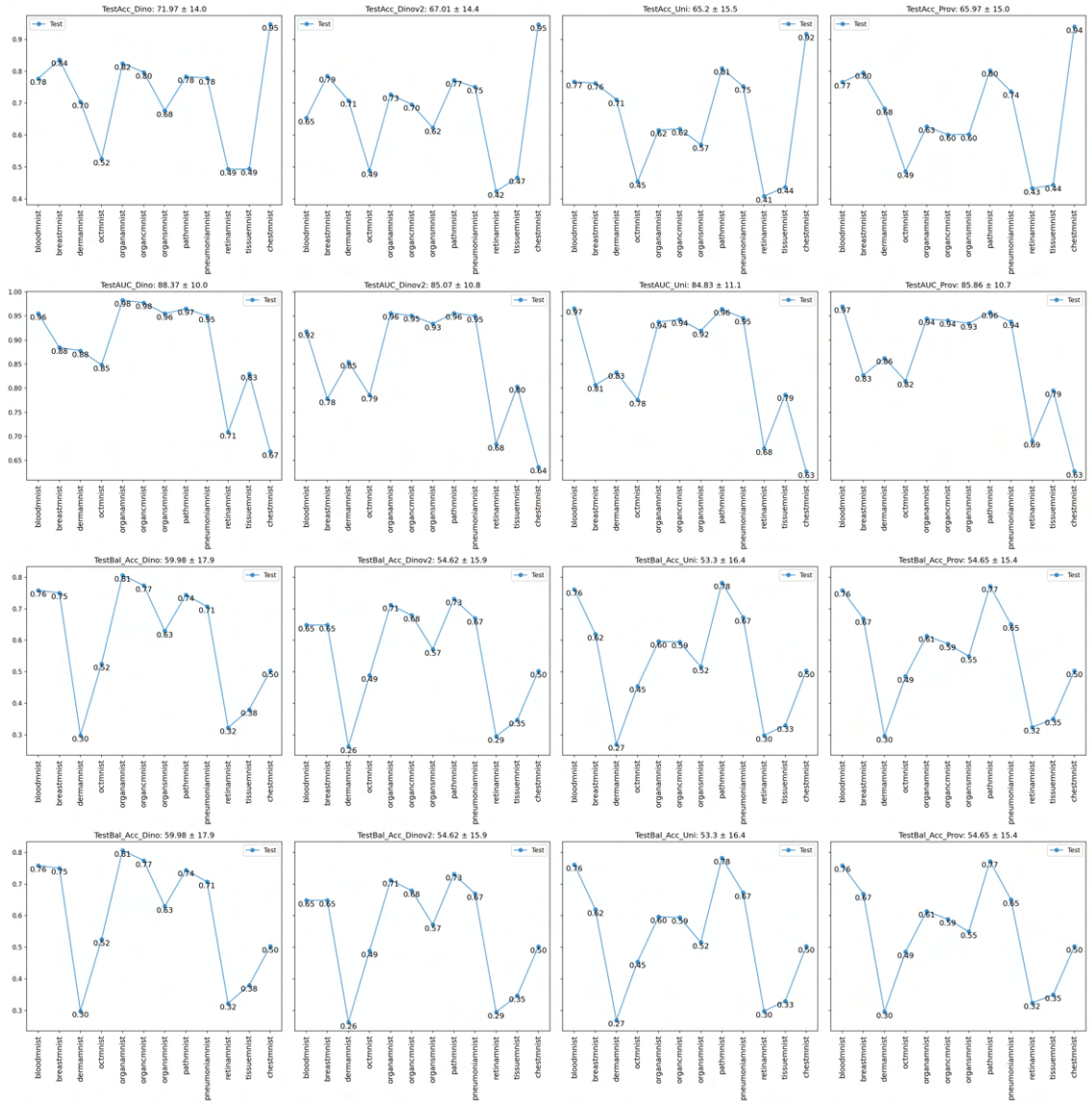


Figure 60: Every metric for 64×64 resolution with LightGBM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

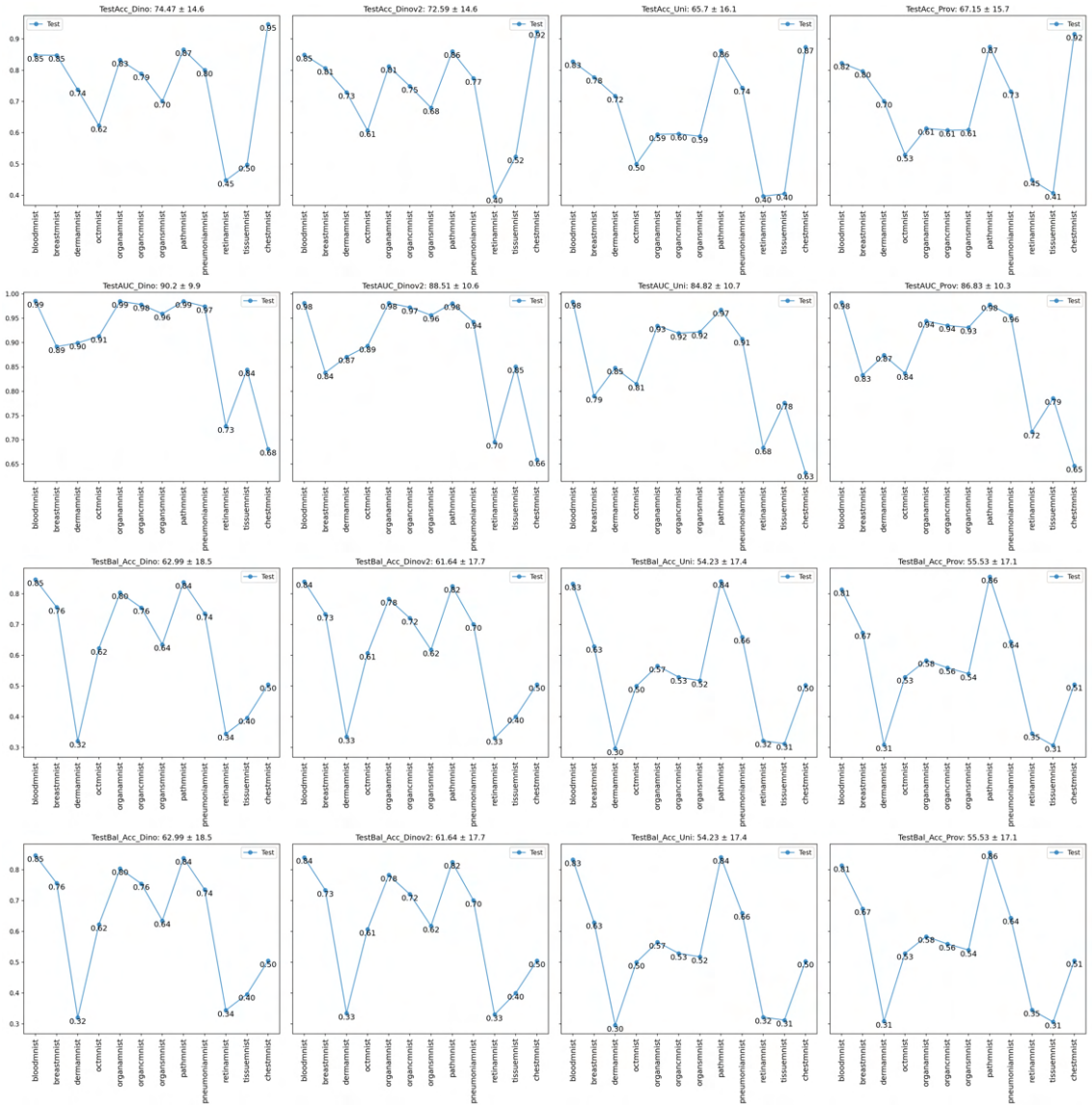


Figure 61: Every metric for 128×128 resolution with LightGBM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

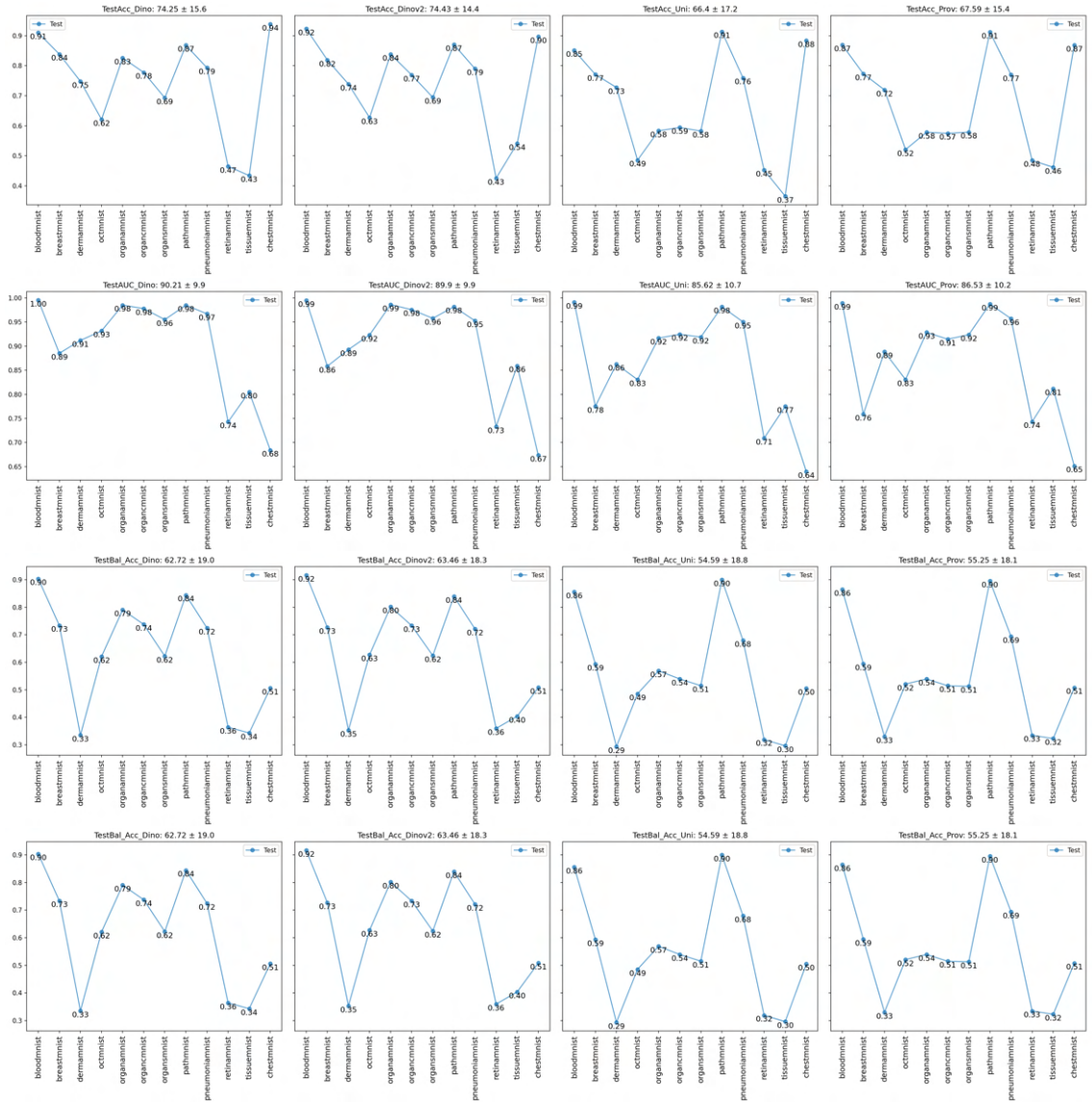


Figure 62: Every metric for 224×224 resolution with LightGBM as classification head for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

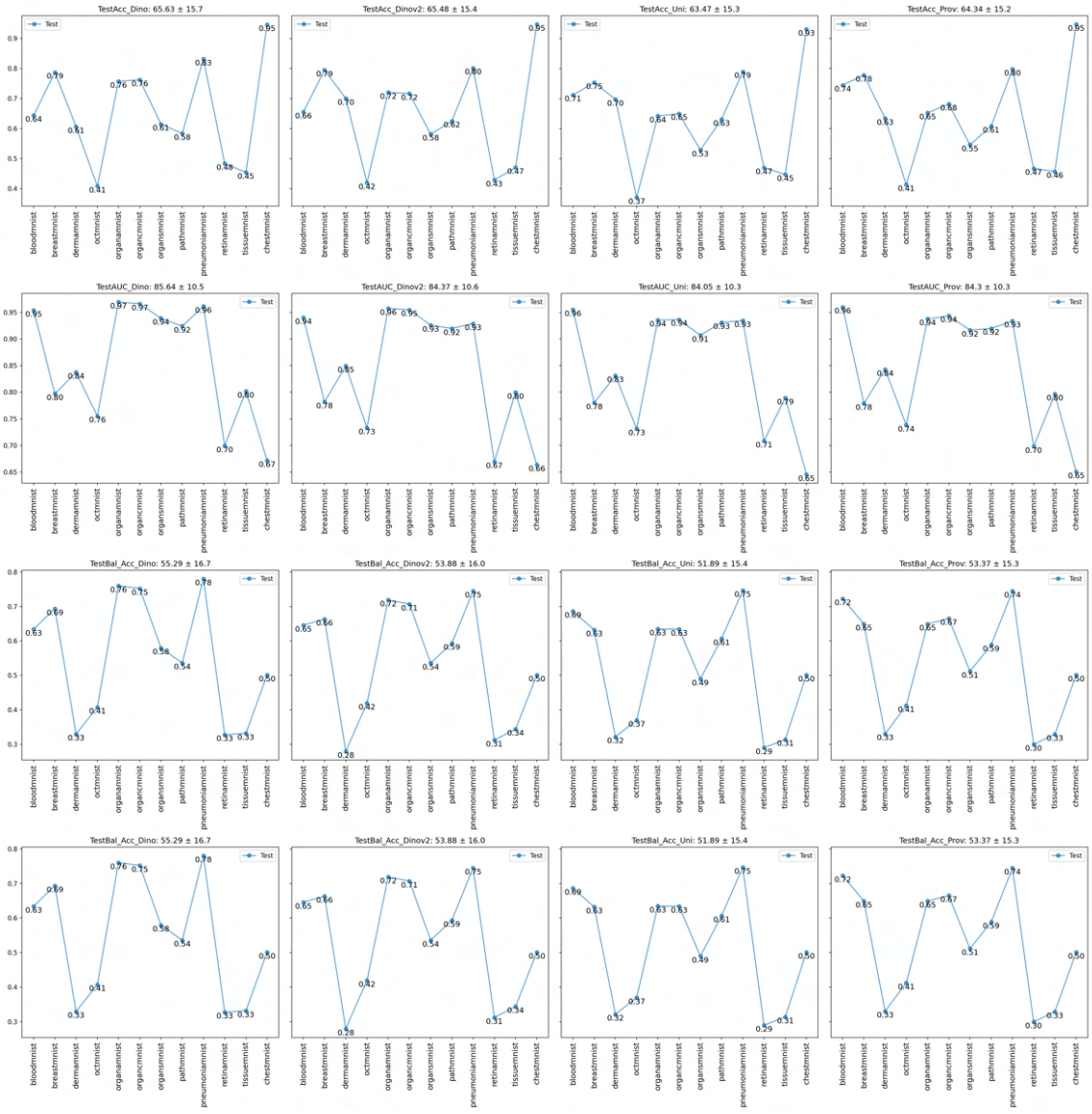


Figure 63: Every metric for 28×28 resolution with linear probing for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot

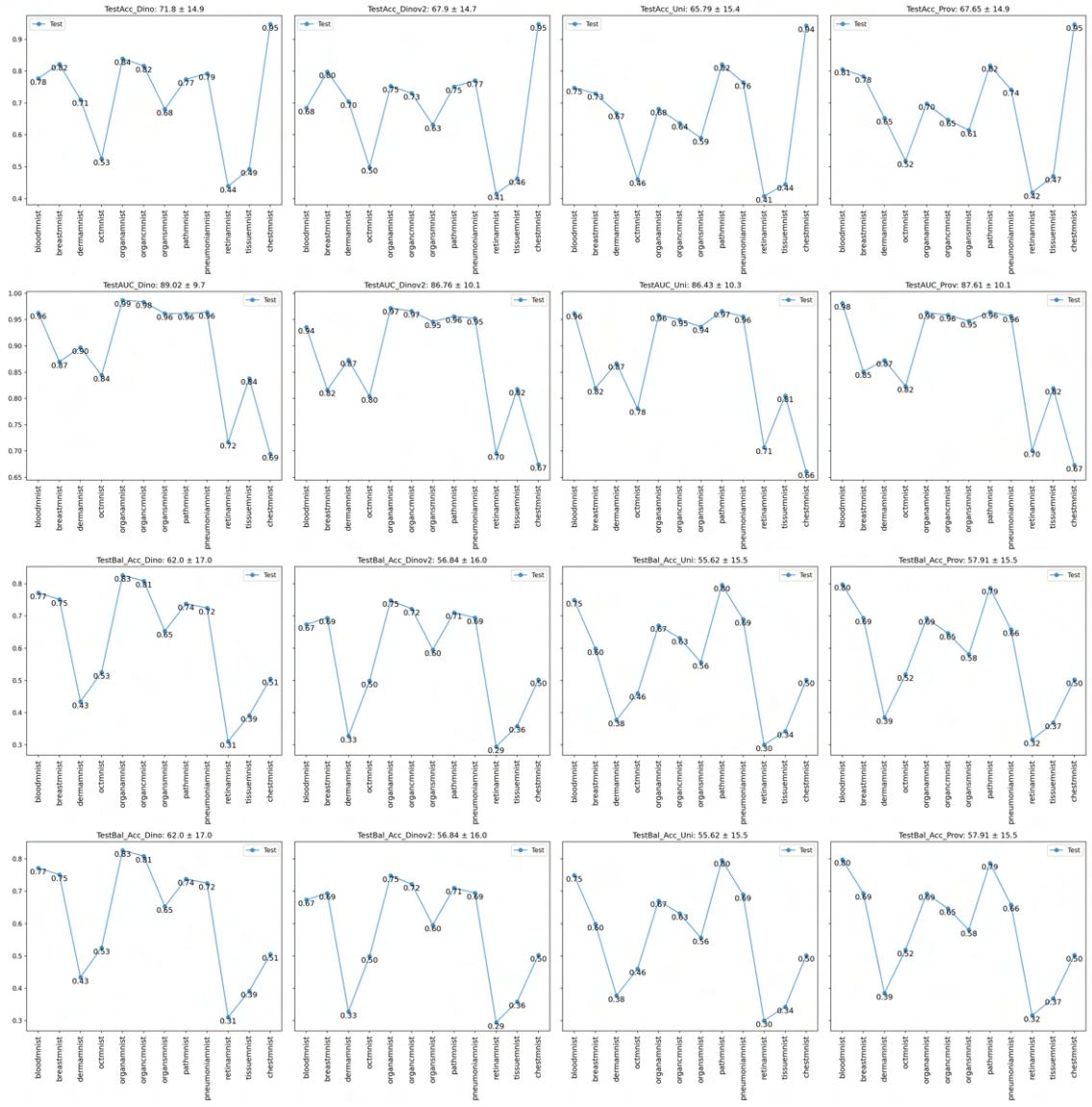


Figure 64: Every metric for 64×64 resolution with linear probing for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

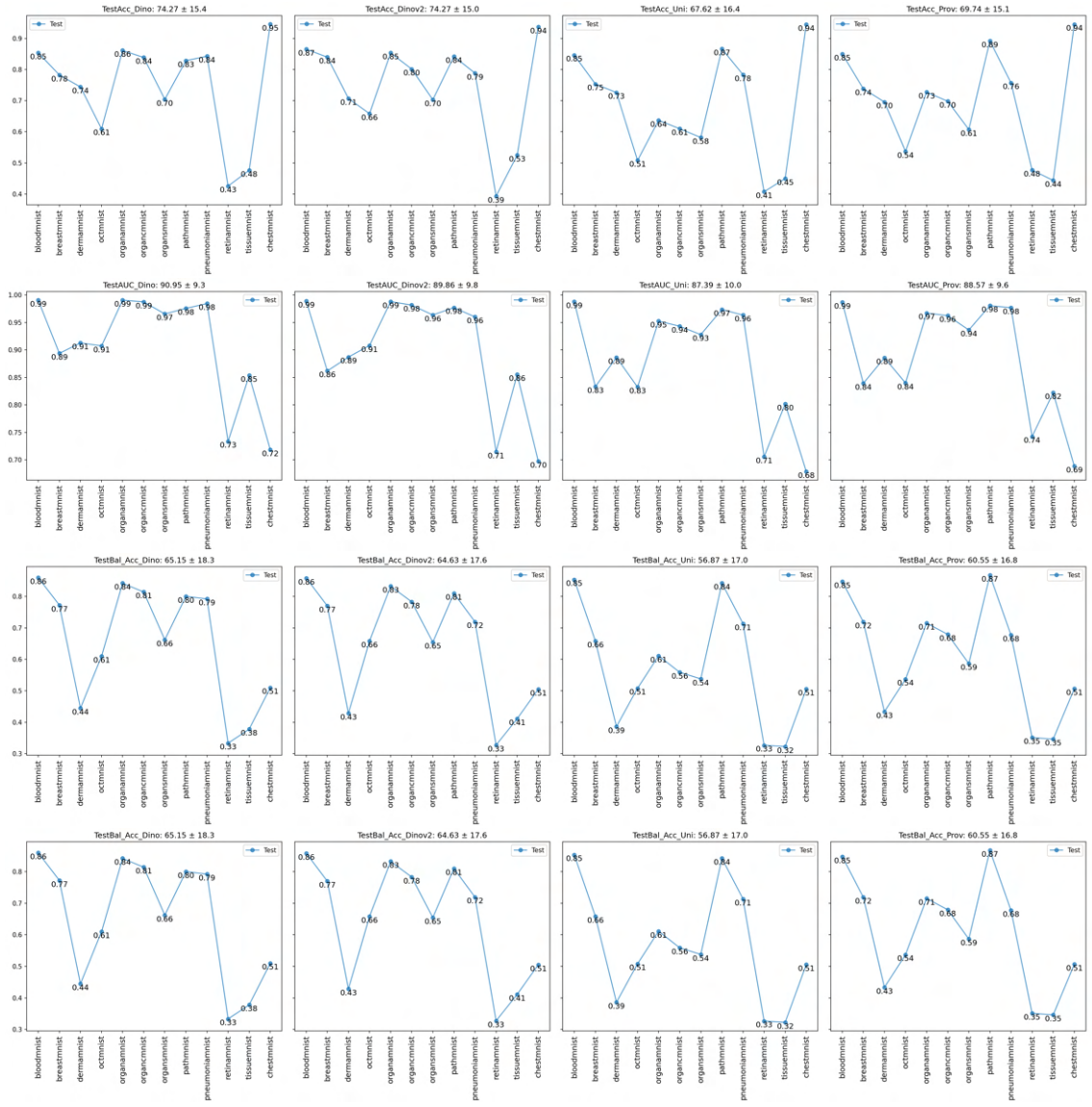


Figure 65: Every metric for 128×128 resolution with linear probing for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

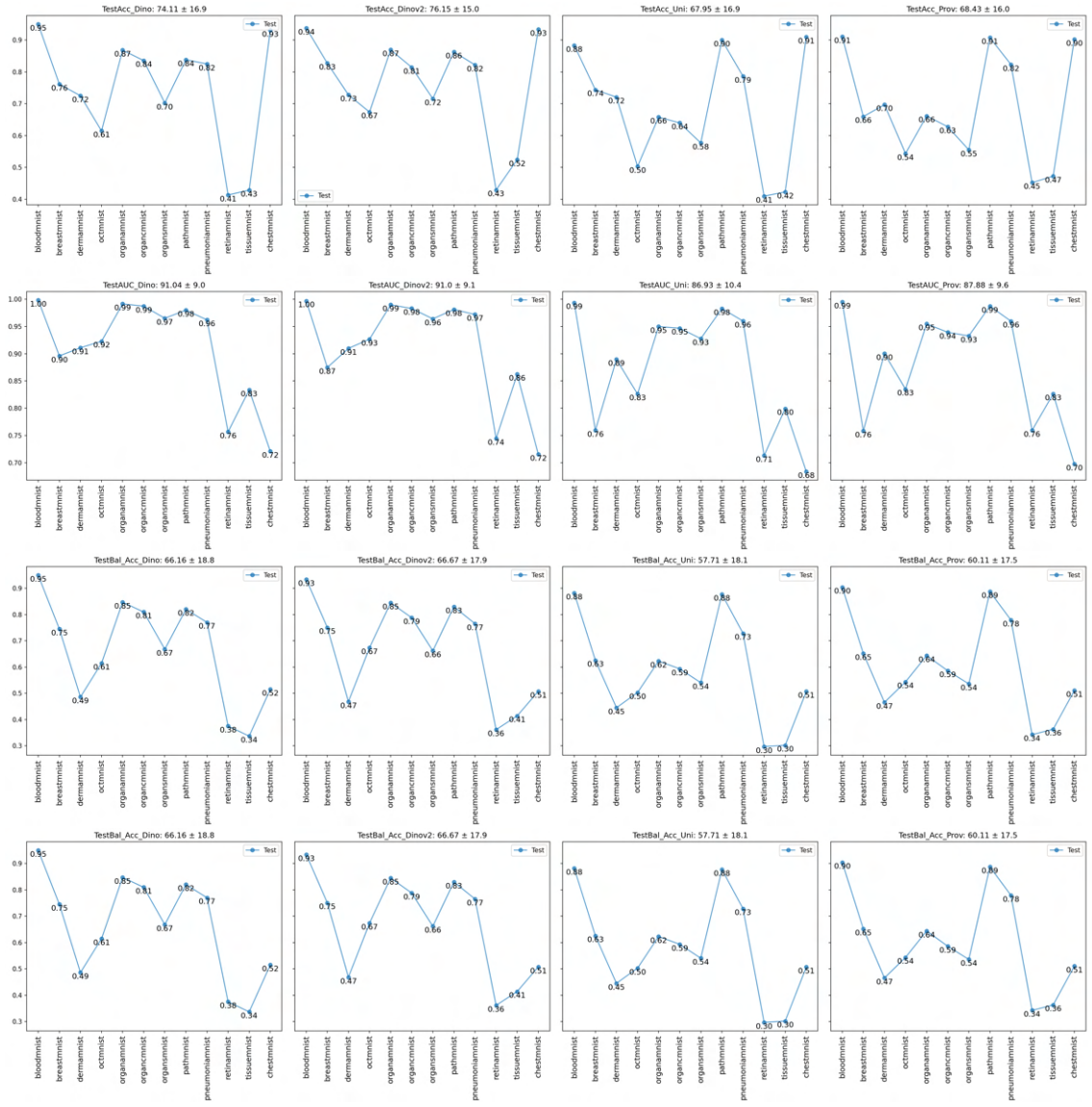


Figure 66: Every metric for 224×224 resolution with linear probing for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

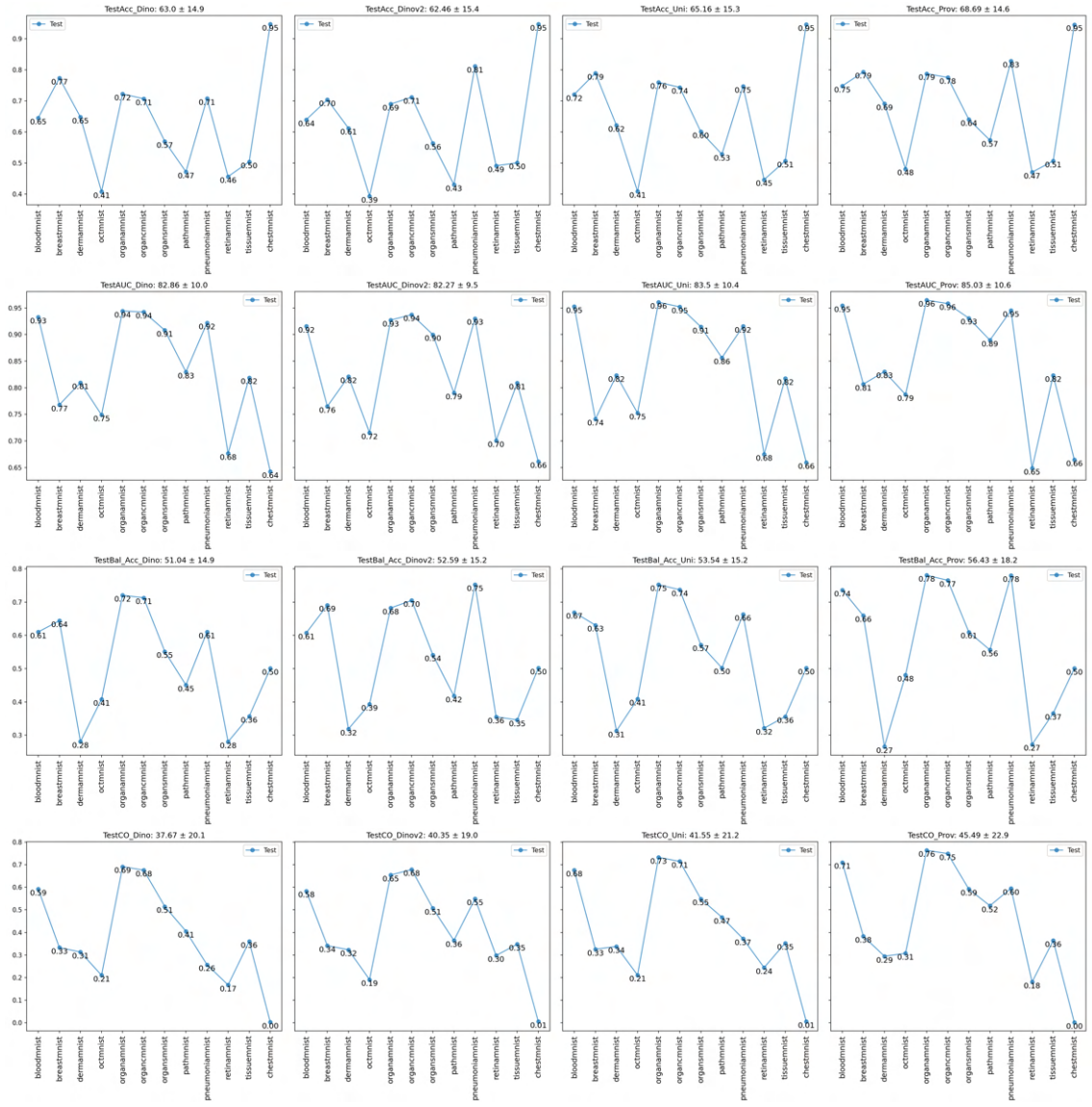


Figure 67: Every metric for 28×28 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

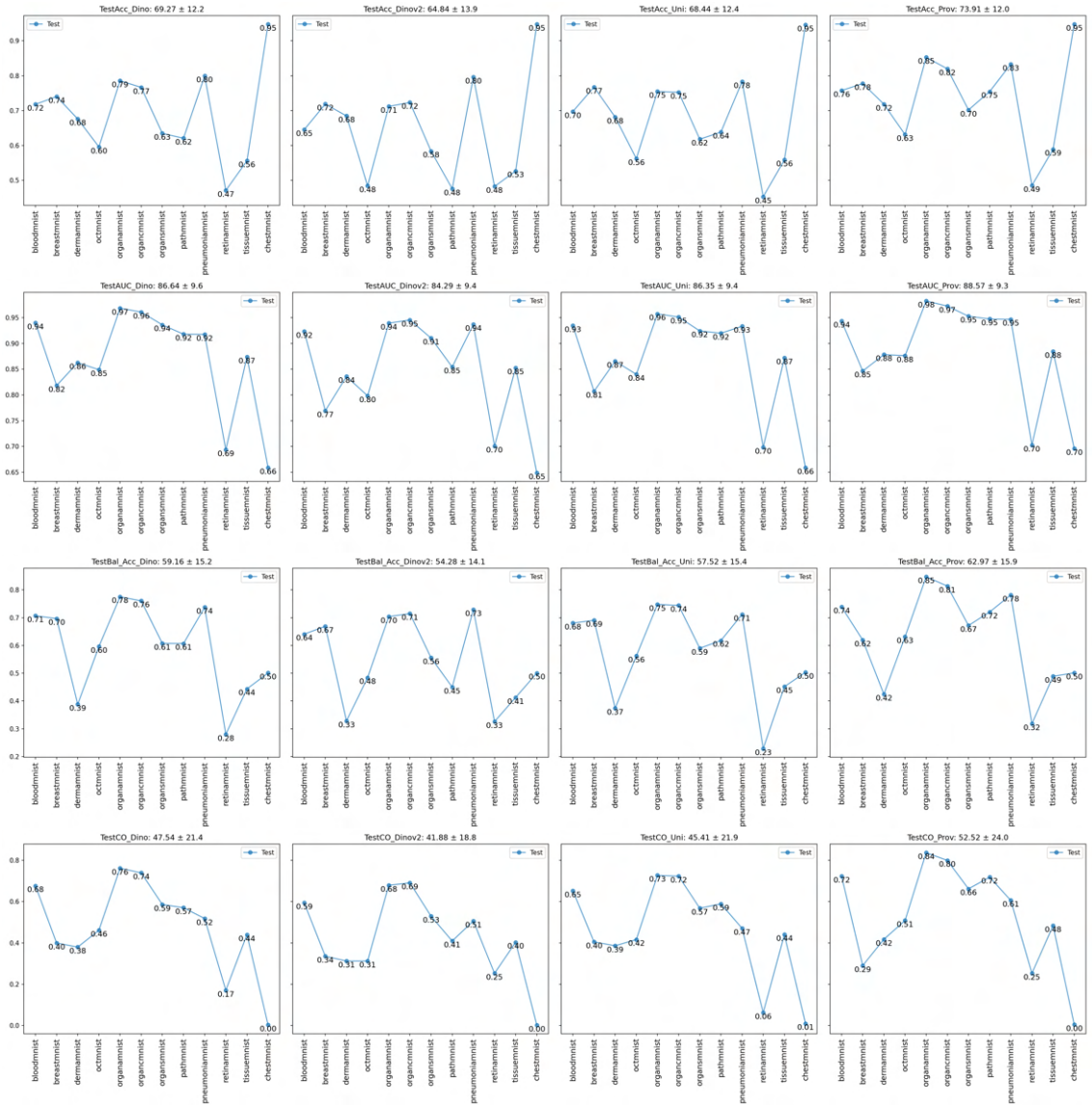


Figure 68: Every metric for 64×64 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

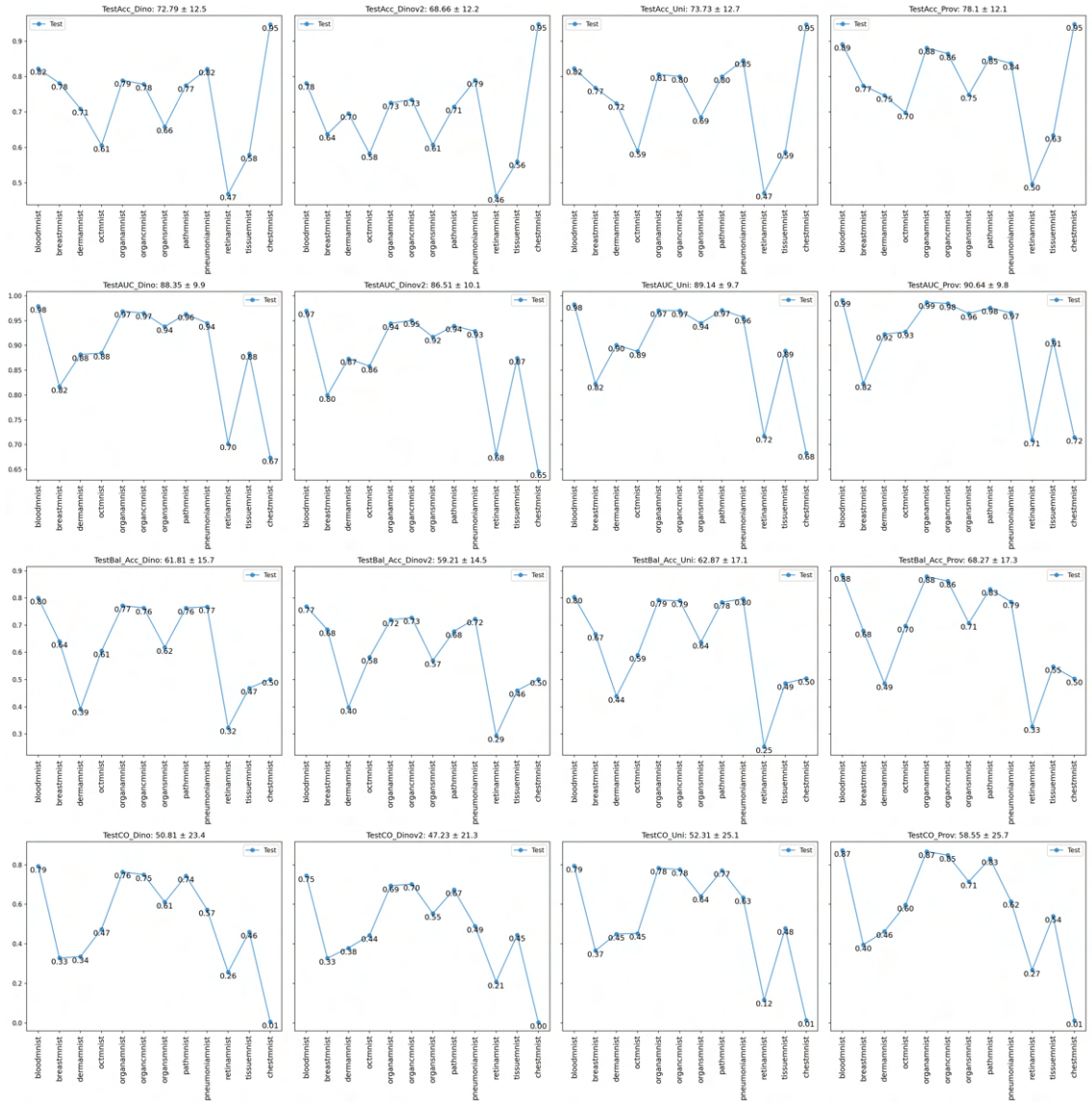


Figure 69: Every metric for 128×128 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

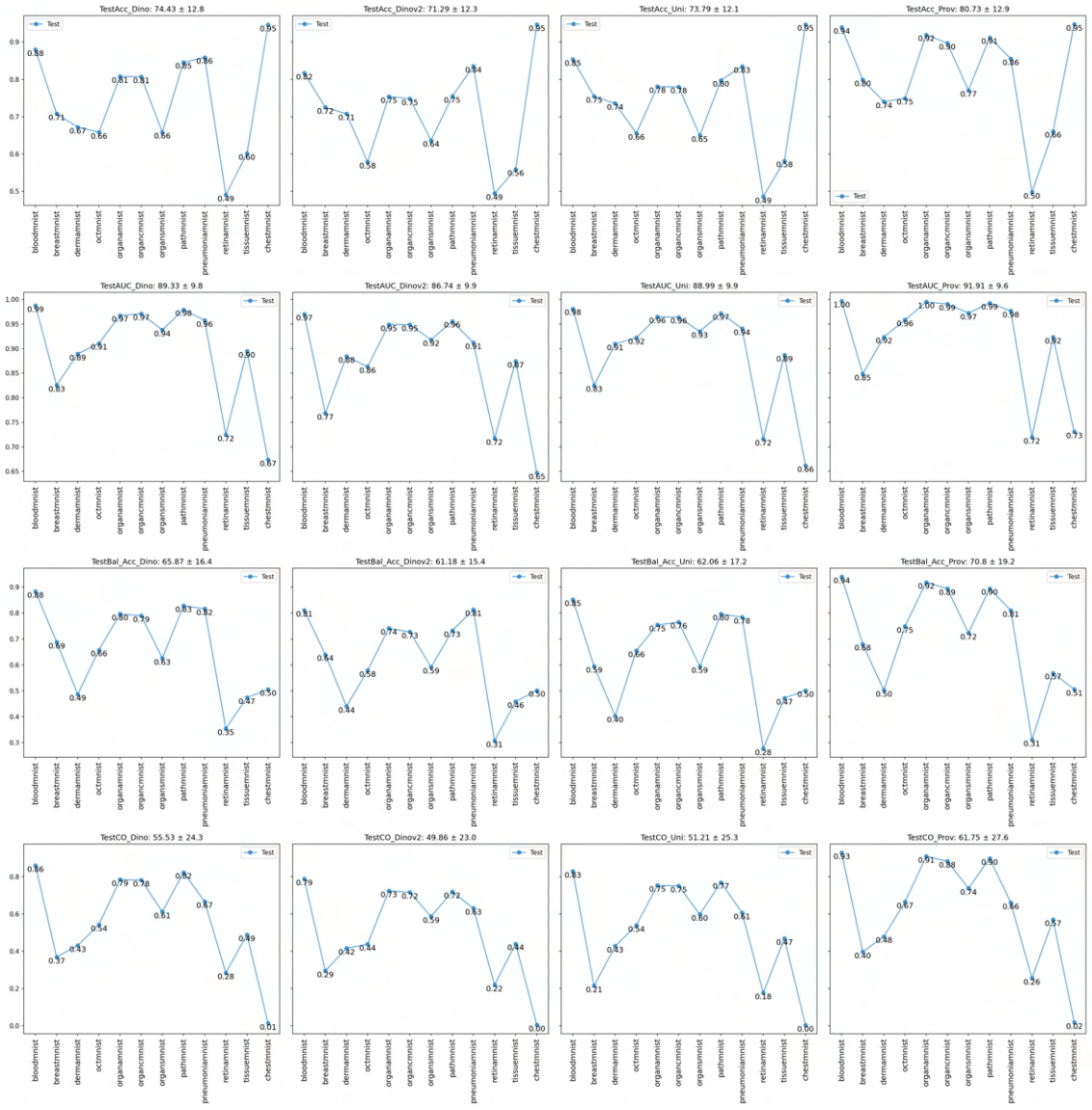


Figure 70: Every metric for 224×224 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

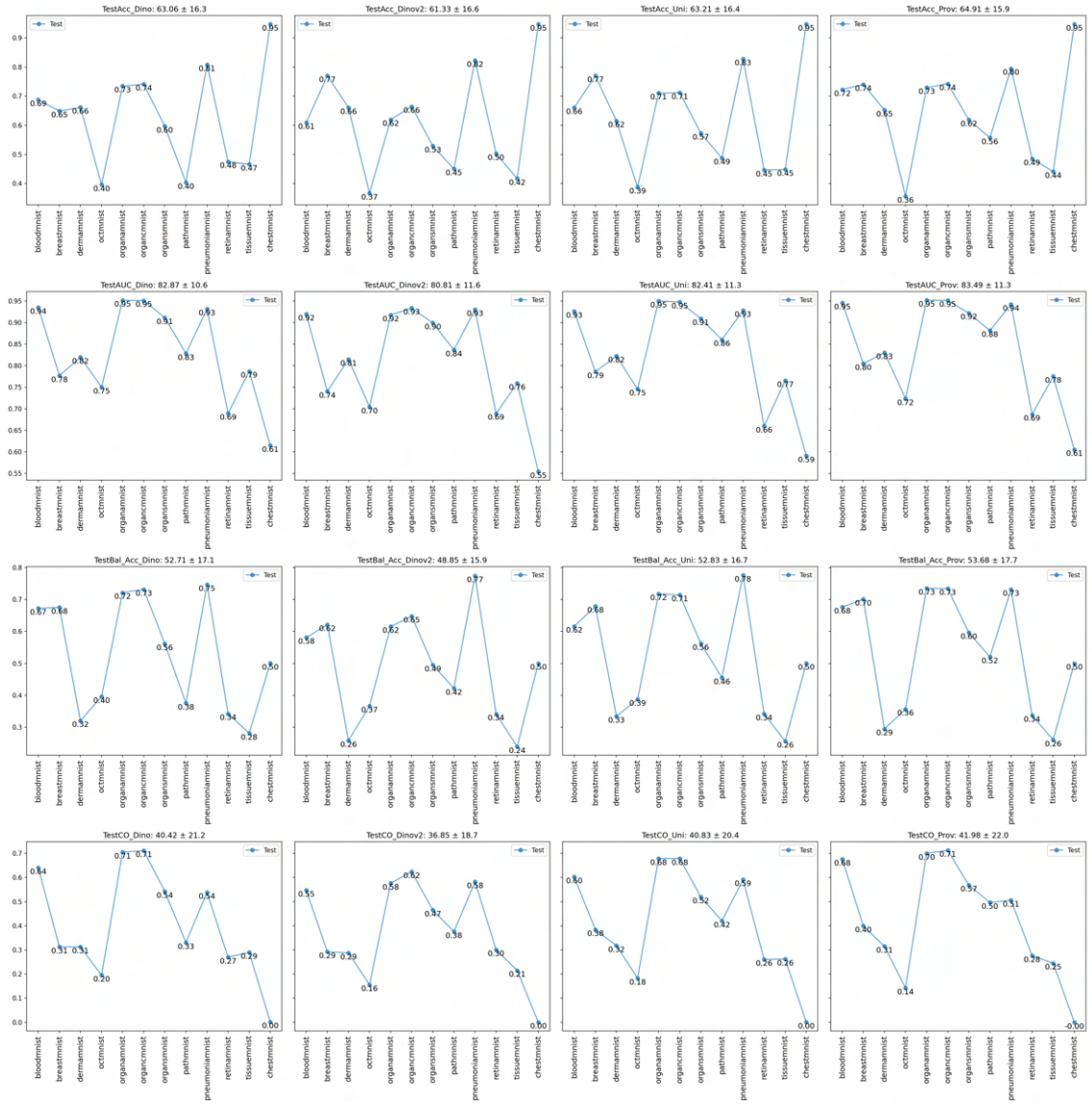


Figure 71: Every metric for 28×28 resolution with multi-domain multi-task pre-training with data augmentation for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

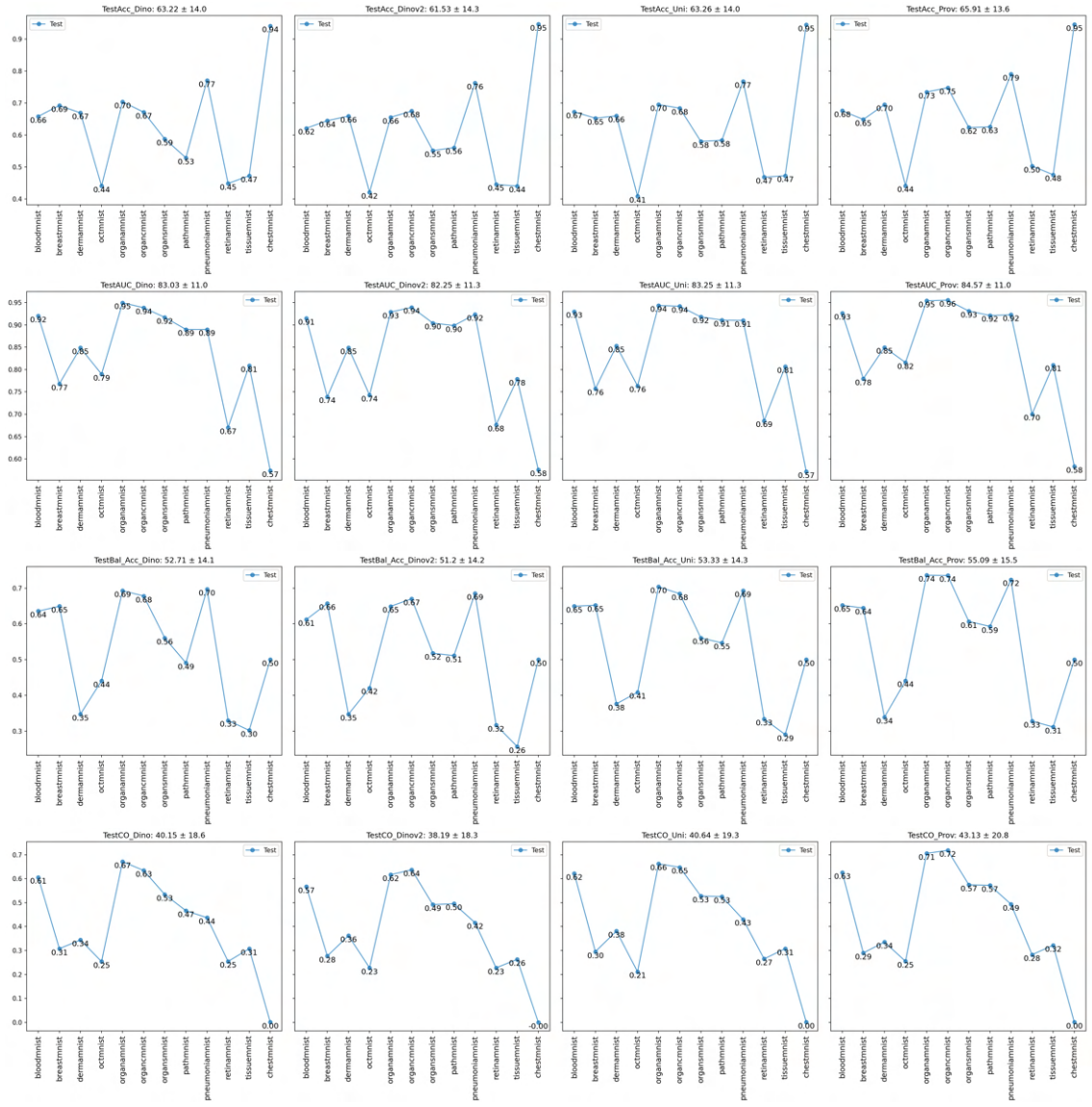


Figure 72: Every metric for 64×64 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

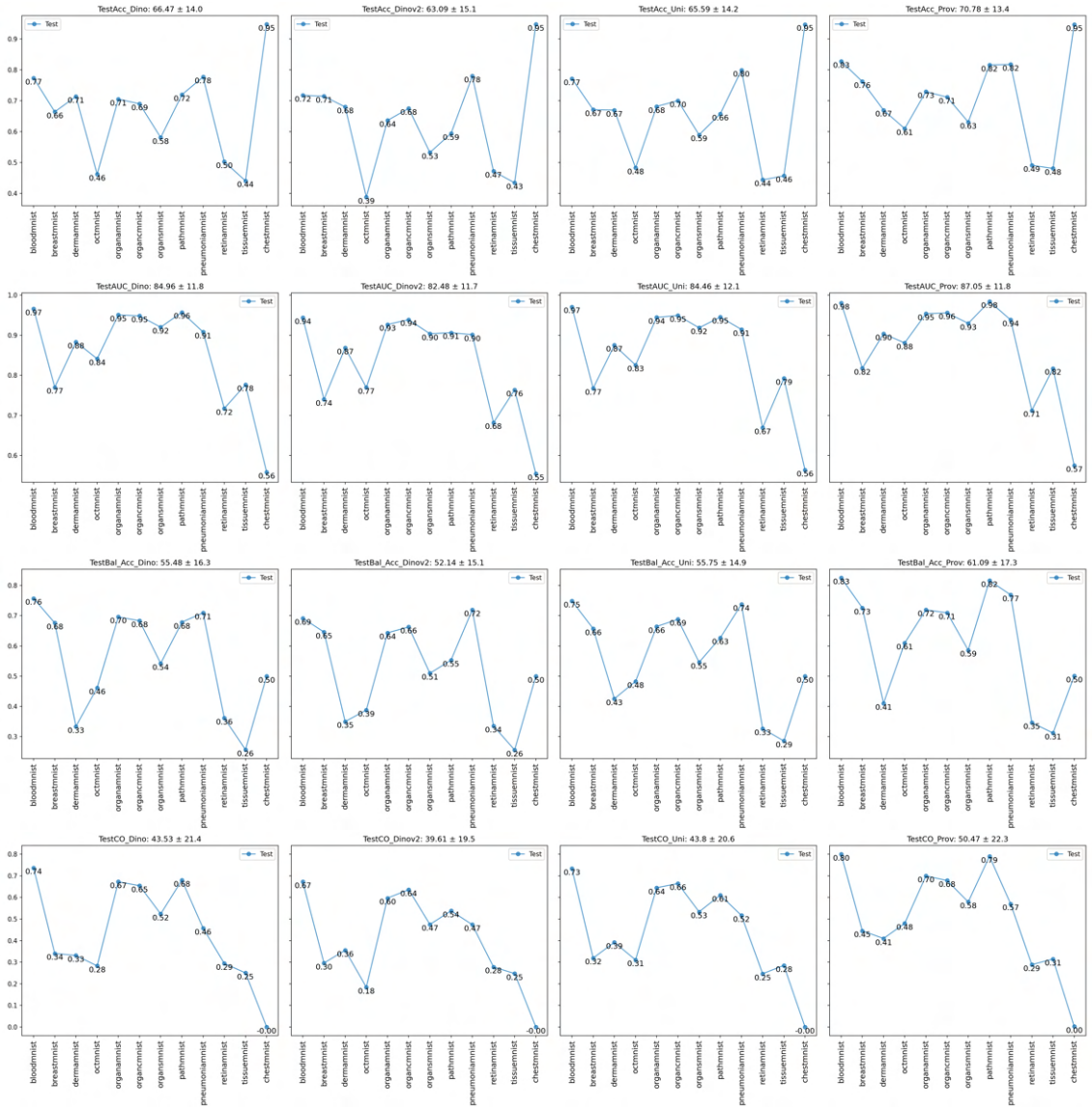


Figure 73: Every metric for 128×128 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

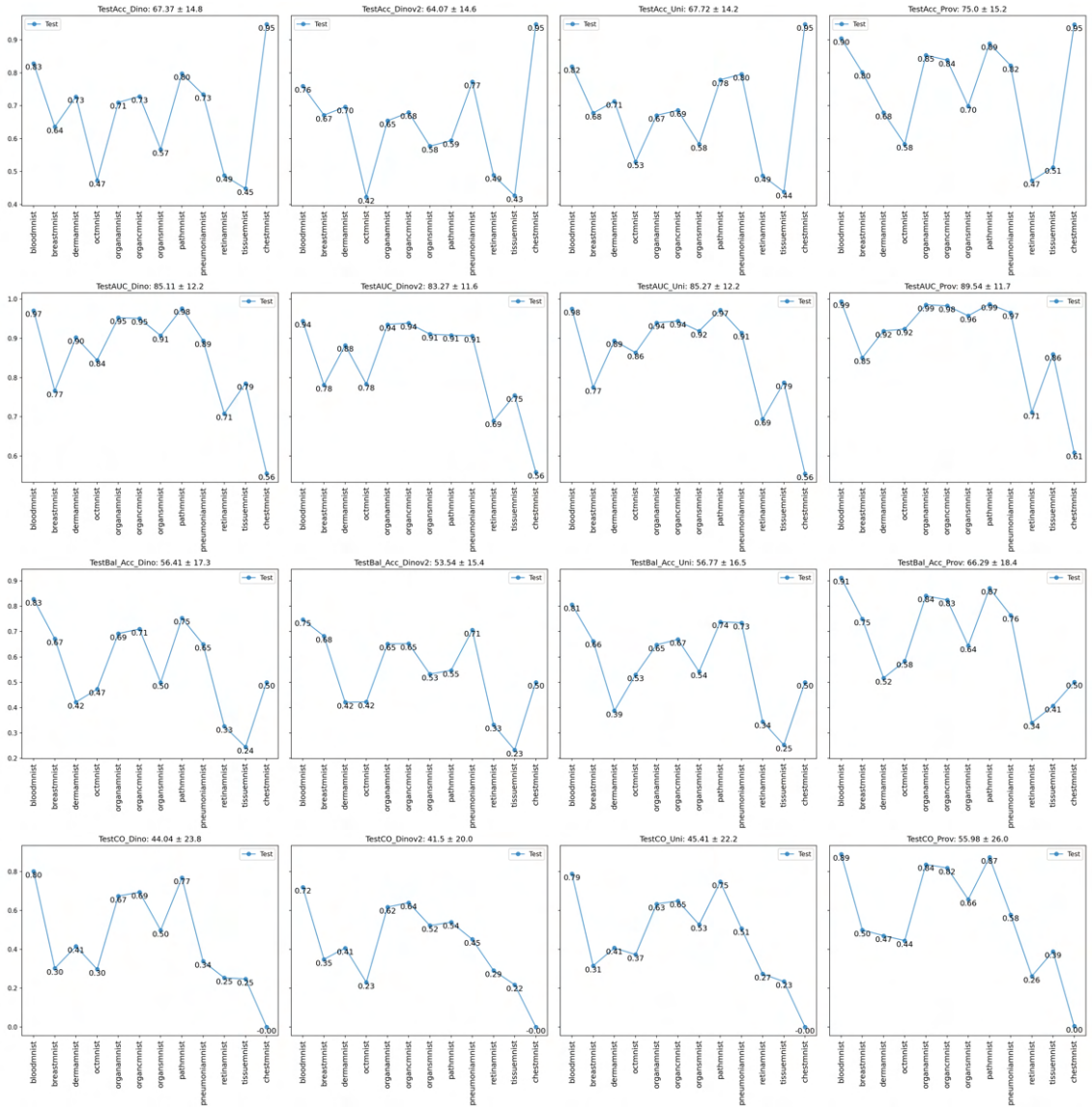


Figure 74: Every metric for 224×224 resolution with multi-domain multi-task pre-training for every backbone. The average and standard deviation over all 12 datasets is shown by the title of each plot.

## A.4 Balanced Accuracy Values for mm-PT with Prov-backbone at a Resolution of 224×224 across different severity Levels of MedMNIST-C

| | bloodmnist | breastmnist | chestmnist | dermamnist | octmnist | otganamnist | organcmnist | organsmnist | pathmnist | pneumoniamnist | retinamnist | tissuemnist |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **JPEG** | 98.38 | 61.47 | 50.85 | 56.55 | 88.5 | 95.67 | 94.2 | 78.01 | 95.45 | 85.38 | 31.88 | 61.94 |
| **Pixelate** | 98.15 | 75.31 | 50.97 | 58.13 | 84.2 | 95.56 | 94.33 | 78.26 | 95.76 | 84.1 | 31.24 | 63.06 |
| **Gaussian Noise** | - | - | 50.8 | 58.81 | - | 95.72 | 94.17 | 78.01 | - | 86.03 | 34.14 | - |
| **Speckle Noise** | - | 61.47 | 50.84 | 59.27 | 88.4 | 95.62 | 94.21 | 78.12 | - | 86.24 | 32.43 | - |
| **Impulse Noise** | - | - | 50.73 | 58.39 | - | 95.52 | 94.01 | 77.87 | - | 84.74 | - | 58.78 |
| **Shot Noise** | - | - | 50.44 | 57.05 | - | 95.62 | 94.09 | 78.11 | - | 86.45 | - | - |
| **Defocus Blur** | 97.84 | - | - | 50.31 | 88.4 | - | - | - | 90.25 | - | 30.23 | - |
| **Motion Blur** | 97.88 | 72.81 | - | 51.44 | 84.4 | - | - | - | 89.57 | - | 27.46 | - |
| **Gaussian Blur** | - | - | 50.95 | - | - | 91.67 | 90.36 | 68.74 | - | 81.97 | - | 62.36 |
| **Zoom Blur** | - | - | - | 55.07 | - | - | - | - | - | - | - | - |
| **Brightness Up** | 98.19 | 60.71 | 50.9 | 58.1 | - | 93.56 | 91.94 | 73.78 | 93.97 | 86.24 | - | 60.96 |
| **Brightness Down** | 98.39 | 74.75 | 50.77 | 58.69 | - | 94.43 | 91.83 | 77.09 | 94.62 | 84.53 | 32.27 | 59.65 |
| **Contrast Up** | 98.06 | - | 50.89 | 59.4 | - | 95.32 | 92.32 | 76.11 | 95.45 | 87.95 | - | 62.96 |
| **Contrast Down** | 98.36 | 62.66 | 50.85 | 58.63 | 86.6 | 95.49 | 93.95 | 78.42 | 95.42 | 83.25 | 31.88 | 62.13 |
| **Saturate** | 98.17 | - | - | - | - | - | - | - | 96.14 | - | - | - |
| **Stain Deposit** | 95.89 | - | - | - | - | - | - | - | 93.46 | - | - | - |
| **Bubble** | 96.83 | - | - | - | - | - | - | - | 95.7 | - | - | - |
| **Gamma Corr. Up** | - | - | 50.85 | - | - | 94.96 | 93.03 | 77.56 | - | 88.38 | - | - |
| **Gamma Corr. Down** | - | - | 50.89 | - | - | 95.38 | 93.91 | 77.1 | - | 83.68 | - | - |
| **Black Corners** | - | - | - | 50.31 | - | - | - | - | - | - | - | - |
| **Characters** | - | - | - | 58.23 | - | - | - | - | - | - | - | - |

Table 34: A detailed summary of benchmark results showing balanced accuracy for each dataset and each type of corruption for severity 1. The results are based on the best training scheme: Prov mm-PT at a resolution of 224×224.

| | bloodmnist | breastmnist | chestmnist | dermamnist | octmnist | otganamnist | organcmnist | organsmnist | pathmnist | pneumoniamnist | retinamnist | tissuemnist |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **JPEG** | 98.02 | 62.66 | 50.83 | 55.67 | 86.4 | 95.67 | 94.16 | 77.95 | 94.61 | 84.96 | 32.52 | 60.9 |
| **Pixelate** | 98.03 | 80.58 | 50.82 | 56.79 | 71.5 | 95.71 | 94.27 | 78.38 | 95.42 | 83.89 | 32.79 | 63.11 |
| **Gaussian Noise** | - | - | 50.53 | 59.88 | - | 95.52 | 93.77 | 77.1 | - | 85.6 | 34.1 | - |
| **Speckle Noise** | - | 60.28 | 50.57 | 57.82 | 81.8 | 95.62 | 94.09 | 77.87 | - | 86.03 | 31.6 | - |
| **Impulse Noise** | - | - | 50.42 | 58.53 | - | 95.28 | 93.91 | 77.62 | - | 85.17 | - | 56.4 |
| **Shot Noise** | - | - | 50.11 | 52.61 | - | 95.66 | 93.64 | 77.56 | - | 86.24 | - | - |
| **Defocus Blur** | 96.95 | - | - | 46.26 | 86.4 | - | - | - | 87.54 | - | 29.45 | - |
| **Motion Blur** | 96.29 | 78.82 | - | 51.56 | 79.0 | - | - | - | 89.26 | - | 28.32 | - |
| **Gaussian Blur** | - | - | 50.97 | - | - | 88.21 | 86.47 | 63.99 | - | 77.82 | - | 61.98 |
| **Zoom Blur** | - | - | - | 52.69 | - | - | - | - | - | - | - | - |
| **Brightness Up** | 97.68 | 59.52 | 50.84 | 52.14 | - | 91.47 | 88.84 | 68.02 | 91.47 | 83.25 | - | 59.14 |
| **Brightness Down** | 97.54 | 78.2 | 50.65 | 52.76 | - | 93.37 | 90.22 | 75.84 | 93.72 | 80.0 | 32.17 | 56.15 |
| **Contrast Up** | 97.85 | - | 50.84 | 57.51 | - | 94.69 | 91.25 | 74.11 | 94.47 | 89.02 | - | 62.06 |
| **Contrast Down** | 98.3 | 63.41 | 50.79 | 54.48 | 79.9 | 95.12 | 93.25 | 78.03 | 93.93 | 80.81 | 33.47 | 60.95 |
| **saturate** | 97.91 | - | - | - | - | - | - | - | 95.96 | - | - | - |
| **Stain Deposit** | 93.37 | - | - | - | - | - | - | - | 92.26 | - | - | - |
| **Bubble** | 95.99 | - | - | - | - | - | - | - | 95.37 | - | - | - |
| **Gamma Corr. Up** | - | - | 50.85 | - | - | 94.42 | 92.02 | 77.15 | - | 89.23 | - | - |
| **Gamma Corr. Down** | - | - | 50.85 | - | - | 94.85 | 93.34 | 76.0 | - | 80.81 | - | - |
| **Black Corners** | - | - | - | 48.56 | - | - | - | - | - | - | - | - |
| **Characters** | - | - | - | 58.94 | - | - | - | - | - | - | - | - |

Table 35: A detailed summary of benchmark results showing balanced accuracy for each dataset and each type of corruption for severity 2. The results are based on the best training scheme: Prov mm-PT at a resolution of 224×224.

| | bloodmnist | breastmnist | chestmnist | dermamnist | octmnist | otganamnist | organcmnist | organsmnist | pathmnist | pneumoniamnist | retinamnist | tissuemnist |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **JPEG** | 97.41 | 59.09 | 50.76 | 54.64 | 84.8 | 95.64 | 93.92 | 77.86 | 89.54 | 84.74 | 31.71 | 57.05 |
| **Pixelate** | 98.07 | 80.26 | 50.91 | 56.09 | 75.7 | 95.52 | 94.16 | 77.99 | 94.58 | 82.39 | 30.62 | 62.2 |
| **Gaussian Noise** | - | - | 50.24 | 55.71 | - | 94.37 | 92.49 | 75.31 | - | 85.17 | 34.88 | - |
| **Speckle Noise** | - | 61.47 | 50.33 | 54.27 | 75.2 | 95.03 | 93.1 | 76.17 | - | 85.6 | 32.1 | - |
| **Impulse Noise** | - | - | 50.13 | 55.34 | - | 95.02 | 93.47 | 76.63 | - | 84.96 | - | 54.13 |
| **Shot Noise** | - | - | 50.05 | 49.42 | - | 95.09 | 93.41 | 76.3 | - | 85.38 | - | - |
| **Defocus Blur** | 94.47 | - | - | 42.84 | 83.5 | - | - | - | 80.94 | - | 28.21 | - |
| **Motion Blur** | 96.18 | 69.49 | - | 47.49 | 68.1 | - | - | - | 88.95 | - | 26.92 | - |
| **Gaussian Blur** | - | - | 51.03 | - | - | 84.03 | 81.39 | 59.52 | - | 73.46 | - | 61.48 |
| **Zoom Blur** | - | - | - | 50.05 | - | - | - | - | - | - | - | - |
| **Brightness Up** | 94.75 | 58.33 | 50.71 | 48.62 | - | 87.73 | 83.73 | 60.9 | 88.21 | 79.53 | - | 57.5 |
| **Brightness Down** | 95.26 | 83.58 | 50.49 | 41.7 | - | 91.62 | 87.7 | 73.53 | 92.35 | 72.95 | 31.95 | 51.37 |
| **Contrast Up** | 97.31 | - | 50.73 | 54.81 | - | 91.39 | 87.45 | 68.12 | 93.21 | 89.44 | - | 59.65 |
| **Contrast Down** | 97.84 | 69.36 | 50.69 | 50.9 | 74.1 | 94.08 | 91.69 | 76.32 | 90.76 | 76.58 | 33.82 | 58.54 |
| **Saturate** | 97.57 | - | - | - | - | - | - | - | 95.42 | - | - | - |
| **Stain Deposit** | 90.38 | - | - | - | - | - | - | - | 91.11 | - | - | - |
| **Bubble** | 95.12 | - | - | - | - | - | - | - | 94.8 | - | - | - |
| **Gamma Corr. Up** | - | - | 50.85 | - | - | 92.9 | 89.84 | 75.39 | - | 89.74 | - | - |
| **Gamma Corr. Down** | - | - | 50.75 | - | - | 93.86 | 92.1 | 73.75 | - | 76.45 | - | - |
| **Black Corners** | - | - | - | 47.92 | - | - | - | - | - | - | - | - |
| **Characters** | - | - | - | 57.27 | - | - | - | - | - | - | - | - |

Table 36: A detailed summary of benchmark results showing balanced accuracy for each dataset and each type of corruption for severity 3. The results are based on the best training scheme: Prov mm-PT at a resolution of 224×224.

| | bloodmnist | breastmnist | chestmnist | dermamnist | octmnist | otganamnist | organcmnist | organsmnist | pathmnist | pneumoniamnist | retinamnist | tissuemnist |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **JPEG** | 96.4 | 59.09 | 50.7 | 51.49 | 82.5 | 95.54 | 94.07 | 77.56 | 85.24 | 83.89 | 32.37 | 51.89 |
| **Pixelate** | 97.74 | 76.0 | 50.59 | 54.56 | 72.0 | 95.35 | 94.28 | 78.19 | 94.14 | 81.11 | 31.44 | 61.55 |
| **Gaussian Noise** | - | - | 50.03 | 50.01 | - | 92.46 | 90.54 | 72.61 | - | 85.17 | 32.84 | - |
| **Speckle Noise** | - | 60.28 | 50.03 | 46.33 | 64.7 | 93.02 | 91.48 | 73.08 | - | 85.09 | 32.42 | - |
| **Impulse Noise** | - | - | 50.05 | 50.56 | - | 94.16 | 92.6 | 75.29 | - | 84.74 | - | 52.34 |
| **Shot Noise** | - | - | 50.0 | 44.44 | - | 93.87 | 92.29 | 74.46 | - | 84.74 | - | - |
| **Defocus Blur** | 86.41 | - | - | 40.59 | 79.8 | - | - | - | 65.25 | - | 26.59 | - |
| **Motion Blur** | 92.45 | 72.74 | - | 41.54 | 56.0 | - | - | - | 83.23 | - | 26.1 | - |
| **Gaussian Blur** | - | - | 51.05 | - | - | 80.09 | 74.76 | 55.01 | - | 68.46 | - | 60.08 |
| **Zoom Blur** | - | - | - | 49.67 | - | - | - | - | - | - | - | - |
| **Brightness Up** | 92.06 | 56.7 | 50.61 | 41.85 | - | 82.37 | 76.81 | 54.01 | 86.59 | 71.79 | - | 55.41 |
| **Brightness Down** | 83.94 | 80.64 | 50.36 | 31.92 | - | 87.93 | 84.63 | 70.9 | 91.55 | 66.67 | 30.49 | 45.42 |
| **Contrast Up** | 96.76 | - | 50.66 | 53.5 | - | 88.56 | 84.66 | 64.77 | 91.01 | 89.23 | - | 56.64 |
| **Contrast Down** | 94.52 | 72.49 | 50.53 | 42.2 | 61.9 | 93.13 | 90.69 | 75.02 | 88.05 | 72.52 | 33.65 | 55.5 |
| **Saturate** | 97.09 | - | - | - | - | - | - | - | 94.27 | - | - | - |
| **Stain Deposit** | 90.35 | - | - | - | - | - | - | - | 90.22 | - | - | - |
| **Bubble** | 94.33 | - | - | - | - | - | - | - | 93.95 | - | - | - |
| **Gamma Corr. Up** | - | - | 50.77 | - | - | 90.28 | 87.17 | 72.93 | - | 90.38 | - | - |
| **Gamma Corr.Down** | - | - | 50.64 | - | - | 91.81 | 89.61 | 70.26 | - | 66.32 | - | - |
| **Black Corners** | - | - | - | 45.46 | - | - | - | - | - | - | - | - |
| **Characters** | - | - | - | 56.43 | - | - | - | - | - | - | - | - |

Table 37: A detailed summary of benchmark results showing balanced accuracy for each dataset and each type of corruption for severity 4. The results are based on the best training scheme: Prov mm-PT at a resolution of 224×224.

| | bloodmnist | breastmnist | chestmnist | dermamnist | octmnist | otganamnist | organcmnist | organsmnist | pathmnist | pneumoniamnist | retinamnist | tissuemnist |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **JPEG** | 89.9 | 57.89 | 50.64 | 44.88 | 70.3 | 95.24 | 93.05 | 77.24 | 78.34 | 81.54 | 32.65 | 45.12 |
| **Pixelate** | 96.86 | 74.75 | 50.1 | 52.34 | 54.9 | 95.27 | 94.21 | 77.81 | 90.96 | 79.4 | 30.66 | 58.92 |
| **Gaussian Noise** | - | - | 50.01 | 43.17 | - | 89.36 | 87.63 | 69.58 | - | 84.74 | 35.23 | - |
| **Speckle Noise** | - | 61.9 | 50.0 | 42.14 | 56.5 | 88.82 | 87.03 | 68.77 | - | 83.25 | 33.43 | - |
| **Impulse Noise** | - | - | 50.0 | 42.75 | - | 93.0 | 91.01 | 72.96 | - | 83.68 | - | 50.83 |
| **Shot Noise** | - | - | 49.99 | 38.11 | - | 91.77 | 89.98 | 70.4 | - | 83.38 | - | - |
| **Defocus Blur** | 77.27 | - | - | 37.01 | 75.0 | - | - | - | 44.48 | - | 26.93 | - |
| **Motion Blur** | 87.43 | 65.35 | - | 35.77 | 51.6 | - | - | - | 67.56 | - | 24.91 | - |
| **Gaussian Blur** | - | - | 51.07 | - | - | 71.28 | 62.74 | 47.51 | - | 63.12 | - | 58.55 |
| **Zoom Blur** | - | - | - | 47.77 | - | - | - | - | - | - | - | - |
| **Brightness Up** | 87.18 | 55.51 | 50.49 | 36.74 | - | 76.96 | 69.23 | 49.29 | 84.54 | 65.94 | - | 53.61 |
| **Brightness Down** | 60.74 | 74.25 | 50.2 | 22.47 | - | 83.0 | 80.79 | 67.94 | 90.93 | 59.83 | 27.91 | 38.2 |
| **Contrast Up** | 95.5 | - | 50.48 | 49.47 | - | 84.86 | 80.85 | 61.25 | 86.24 | 89.62 | - | 49.5 |
| **Contrast Down** | 81.02 | 78.13 | 50.26 | 35.09 | 55.4 | 91.84 | 88.87 | 73.3 | 84.37 | 53.21 | 26.85 | 45.07 |
| **Saturate** | 96.07 | - | - | - | - | - | - | - | 91.36 | - | - | - |
| **Stain Deposit** | 85.8 | - | - | - | - | - | - | - | 88.02 | - | - | - |
| **Bubble** | 92.27 | - | - | - | - | - | - | - | 92.61 | - | - | - |
| **Gamma Corr. Up** | - | - | 50.69 | - | - | 86.63 | 84.65 | 70.0 | - | 89.06 | - | - |
| **Gamma Corr. down** | - | - | 50.36 | - | - | 77.35 | 73.22 | 52.45 | - | 51.07 | - | - |
| **Black Corners** | - | - | - | 45.28 | - | - | - | - | - | - | - | - |
| **Characters** | - | - | - | 56.94 | - | - | - | - | - | - | - | - |

Table 38: A detailed summary of benchmark results showing balanced accuracy for each dataset and each type of corruption for severity 5. The results are based on the best training scheme: Prov mm-PT at a resolution of 224×224.

# Bibliography

Andrea Acevedo, Anna Merino, Santiago Alférez, Ángel Molina, Laura Boldú, and José Rodellar. A dataset of microscopic peripheral blood cell images for development of automatic recognition systems. *Data in Brief*, 30:105474, 2020. ISSN 2352-3409. doi: https://doi.org/10.1016/j.dib.2020.105474. URL `https://www.sciencedirect.com/science/article/pii/S2352340920303681`.

Walid Al-Dhabyani, Mohammed Gomaa, Hussien Khaled, and Aly Fahmy. Dataset of breast ultrasound images. *Data in Brief*, 28:104863, 2020. ISSN 2352-3409. doi: https://doi.org/10.1016/j.dib.2019.104863. URL `https://www.sciencedirect.com/science/article/pii/S2352340919312181`.

Chalachew Alemayehu, Geoffrey Mitchell, and Jane Nikles. Barriers for conducting clinical trials in developing countries-a systematic review. *International journal for equity in health*, 17:1–11, 2018.

Michela Antonelli, Annika Reinke, Spyridon Bakas, Keyvan Farahani, Annette Kopp-Schneider, Bennett A. Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M. Summers, Bram van Ginneken, Michel Bilello, Patrick Bilic, Patrick F. Christ, Richard K. G. Do, Marc J. Gollub, Stephan H. Heckers, Henkjan Huisman, William R. Jarnagin, Maureen K. McHugo, Sandy Napel, Jennifer S. Golia Pernicka, Kawal Rhode, Catalina Tobon-Gomez, Eugene Vorontsov, James A. Meakin, Sebastien Ourselin, Manuel Wiesenfarth, Pablo Arbeláez, Byeonguk Bae, Sihong Chen, Laura Daza, Jianjiang Feng, Baochun He, Fabian Isensee, Yuanfeng Ji, Fucang Jia, Ildoo Kim, Klaus Maier-Hein, Dorit Merhof, Akshay Pai, Beomhee Park, Mathias Perslev, Ramin Rezaiifar, Oliver Rippel, Ignacio Sarasua, Wei Shen, Jaemin Son, Christian Wachinger, Liansheng Wang, Yan Wang, Yingda Xia, Daguang Xu, Zhanwei Xu, Yefeng Zheng, Amber L. Simpson, Lena Maier-Hein, and M. Jorge Cardoso. The medical segmentation decathlon. *Nature Communications*, 13(1), July 2022. ISSN 2041-1723. doi: 10.1038/s41467-022-30695-9. URL `http://dx.doi.org/10.1038/s41467-022-30695-9`.

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems*, pages 41–48, 2007.

Mourad Azhari, Altaf Alaoui, Zakia Achraoui, Badia Ettaki, and Jamal Zerouaoui. Adaptation of the random forest method: Solving the problem of pulsar search. In *Proceedings of the 4th International Conference on Smart City Applications (SCA '19)*, pages 1–6. ACM, 2019. doi: 10.1145/3368756.3369025.

Marius Ludwig Bachmeier. Designing a benchmark and leaderboard system for assessing the generalizability of deep learning approaches for medical image classification using the medmnist+ dataset collection. October 2024.

Kamal Berahmand, Fatemeh Daneshfar, Elaheh Sadat Salehi, Yuefeng Li, and Yue Xu. Autoencoders and their applications in machine learning: a survey. *Artif. Intell. Rev.*, 57(2):28, February 2024. URL `http://dblp.uni-trier.de/db/journals/air/air57.html#BerahmandDSLX24`.

Predrag Bilic, Peter Christ, Heng Li, Eugene Vorontsov, Avner Ben-Cohen, Georgios Kaissis, Adam Szeskin, Cedric Jacobs, Gabriel E H Mamani, Guillaume Chartrand, Fabian Lohöfer, Johannes W Holch, Wolfgang Sommer, Frank Hofmann, Anja Hostettler, Nir Lev-Cohain, Michal Drozdzal, Meir Amitai, Roberto Vivanti, Jarek Sosna, Ivan Ezhov, Ashwin Sekuboyina, Francisco Navarro, Florian Kofler, Jörg C Paetzold, Sushant Shit, Xue Hu, Jitka Lipková, Matthias Rempfler, Maxime Piraud, Jonas Kirschke, Benedikt Wiestler, Zhiyong Zhang, Christoph Hülsemeyer, Max Beetz, Florian Ettlinger, Matteo Antonelli, Woong Bae, Miguel Bellver, Lei Bi, Huan Chen, Grzegorz Chlebus, Eric B Dam, Qi Dou, Chao Fu, Bogdan Georgescu, Xavier Giró-I-Nieto, Fred Gruen, Xuejun Han, Fang Heng, Johannes Hesser, Julian H Moltz, Christian Igel, Fabian Isensee, Peter Jäger, Fei Jia, Karthik C Kaluva, Manish Khened, Ian Kim, Joon-Hyuk Kim, Seungwoo Kim, Sarah Kohl, Tadeusz Konopczynski, Alok Kori, Gokul Krishnamurthi, Feng Li, Heng Li, Jian Li, Xue Li, John Lowengrub, Jing Ma, Klaus H Maier-Hein, Kristof Maninis, Heike Meine, Dorothea Merhof, Ankit Pai, Mikkel Perslev, Jens Petersen, Jordi Pont-Tuset, Jing Qi, Xiaogang Qi, Oliver Rippel, Kevin Roth, Ignasi Sarasua, Andreas Schenk, Zhongwei Shen, Juan Torres, Christian Wachinger, Cheng Wang, Lena Weninger, Jun Wu, De Xu, Xiaoqing Yang, Shiwei Yu, Yidan Yuan, Ming Yue, Lin Zhang, Jorge Cardoso, Spyridon Bakas, Rolf Braren, Volker Heinemann, Chandan Pal, Anthony Tang, Slaheddine Kadoury, Luis Soler, Bram van Ginneken, Hossam Greenspan, Leo Joskowicz, and Bjoern Menze. The liver tumor segmentation benchmark (lits). *Medical Image Analysis*, 84:102680, 2023. doi: 10.1016/j.media.2022.102680.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ B. Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen Creel, Jared Quincy Davis, Dorottya Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, and et al. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258, 2021. URL `https://arxiv.org/abs/2108.07258`.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020. URL `https://arxiv.org/abs/2005.14165`.

Erik Brynjolfsson and Andrew McAfee. The business of artificial intelligence. *Harvard Business Review*, 2017. URL `https://hbr.org/2017/07/the-business-of-artificial-intelligence`.

Aldo Bustos, Antonio Pertusa, José M Salinas, and María de la Iglesia-Vayá. Padchest: A large chest x-ray image dataset with multi-label annotated reports. *Medical image analysis*, 66:101797, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *CoRR*, abs/2104.14294, 2021. URL `https://arxiv.org/abs/2104.14294`.

Richard J. Chen, Tong Ding, Ming Y. Lu, Drew F. K. Williamson, Guillaume Jaume, Bowen Chen, Andrew Zhang, Daniel Shao, Andrew H. Song, Muhammad Shaban, Mane Williams, Anurag Vaidya, Sharifa Sahai, Lukas Oldenburg, Luca L. Weishaupt, Judy J. Wang, Walt Williams, Long Phi Le, Georg Gerber, and Faisal Mahmood. A general-purpose self-supervised model for computational pathology, 2023. URL `https://arxiv.org/abs/2308.15474`.

Dan Claudiu Ciresan, Ueli Meier, Jonathan Masci, and Jürgen Schmidhuber. High-performance neural networks for visual object classification. In *Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167. ACM, 2008.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

Jiayu Ding, Shuming Ma, Li Dong, Xingxing Zhang, Shaohan Huang, Wenhui Wang, Nanning Zheng, and Furu Wei. Longnet: Scaling transformers to 1,000,000,000 tokens, 2023. URL `https://arxiv.org/abs/2307.02486`.

Sebastian Doerrich, Tobias Archut, Francesco Di Salvo, and Christian Ledig. Integrating knn with foundation models for adaptable and privacy-aware image classification. In *Proceedings of the 2024 IEEE International Symposium on Biomedical Imaging (ISBI)*, pages 1–5, Athens, Greece, 2024a. IEEE. doi: 10.1109/isbi56570.2024.10635560.

Sebastian Doerrich, Francesco Di Salvo, Julius Brockmann, and Christian Ledig. Rethinking model prototyping through the medmnist+ dataset collection. *CoRR*, abs/2404.15786, 2024b. URL `https://arxiv.org/abs/2404.15786`.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. URL `https://arxiv.org/abs/2010.11929`.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Simon Graham, Quoc Dang Vu, Mostafa Jahanifar, Shan E Ahmed Raza, Fayyaz Minhas, David Snead, and Nasir Rajpoot. One model is all you need: Multi-task learning enables simultaneous histology image segmentation and classification. *Medical Image Analysis*, 83:102685, 2023. ISSN 1361-8415. doi: https://doi.org/10.1016/j.media.2022.102685. URL `https://www.sciencedirect.com/science/article/pii/S1361841522003139`.

Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2): 245–258, 2017. ISSN 0896-6273. doi: https://doi.org/10.1016/j.neuron.2017.06.011. URL `https://www.sciencedirect.com/science/article/pii/S0896627317305093`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL `http://arxiv.org/abs/1512.03385`.

Ming He, Guangyi Lv, Weidong He, Jianping Fan, and Guihua Zeng. Deepme: Deep mixture experts for large-scale image classification. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 722–728. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/100. URL `https://doi.org/10.24963/ijcai.2021/100`. Main Track.

Dan Hendrycks and Thomas G Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations (ICLR)*, 2019.

Sara Hooker. The hardware lottery. *CoRR*, abs/2009.06489, 2020. URL `https://arxiv.org/abs/2009.06489`.

Sadegh Bafandeh Imandoust and Mohammad Bolandraftar. Application of k-nearest neighbor (knn) approach for predicting economic events: Theoretical background. *International Journal of Engineering Research and Applications*, 3(5):605–610, 2013.

Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Bohan Haghgoo, Robyn Ball, Katie Shpanskaya, et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33 (01):590–597, 2019.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1), 2021. ISSN 2227-7080. doi: 10.3390/technologies9010002. URL `https://www.mdpi.com/2227-7080/9/1/2`.

Javatpoint. Machine learning - support vector machine algorithm, 2023. URL `https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm`. Accessed: 2024-10-29.

Michael I. Jordan and Tom M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015. doi: 10.1126/science.aaa8415.

Andreas Kaplan and Michael Haenlein. Siri, siri, in my hand: Who's the fairest in the land? on the interpretations, illustrations, and implications of artificial intelligence. *Business Horizons*, 62(1):15–25, 2019. ISSN 0007-6813. doi: https://doi.org/10.1016/j.bushor.2018.08.004. URL `https://www.sciencedirect.com/science/article/pii/S0007681318301393`.

Johannes N Kather, Julian Krisam, Phattarachai Charoentong, Tim Luedde, Eike Herpel, Christian A Weis, Thomas Gaiser, Andreas Marx, Nikolaus A Valous, Daniel Ferber, Laura Jansen, Carmen C Reyes-Aldasoro, Ingo Zörnig, Daniel Jäger, Hermann Brenner, Jenny Chang-Claude, Michael Hoffmeister, and Norbert Halama. Predicting survival from colorectal cancer histology slides using deep learning: A retrospective multicenter study. *PLOS Medicine*, 16(1):e1002730, 2019. doi: 10.1371/journal.pmed.1002730.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30, pages 3146–3154, 2017.

Daniel S. Kermany, Michael Goldbaum, Wenjia Cai, Carolina C.S. Valentim, Huiying Liang, Sally L. Baxter, Alex McKeown, Ge Yang, Xiaokang Wu, Fangbing

Yan, Justin Dong, Made K. Prasadha, Jacqueline Pei, Magdalene Y.L. Ting, Jie Zhu, Christina Li, Sierra Hewett, Jason Dong, Ian Ziyar, Alexander Shi, Runze Zhang, Lianghong Zheng, Rui Hou, William Shi, Xin Fu, Yaou Duan, Viet A.N. Huu, Cindy Wen, Edward D. Zhang, Charlotte L. Zhang, Oulan Li, Xiaobo Wang, Michael A. Singer, Xiaodong Sun, Jie Xu, Ali Tafreshi, M. Anthony Lewis, Huimin Xia, and Kang Zhang. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell*, 172(5):1122–1131.e9, 2018. ISSN 0092-8674. doi: https://doi.org/10.1016/j.cell.2018.02.010. URL `https://www.sciencedirect.com/science/article/pii/S0092867418301545`.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6980`.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Ethan David, Ian Stavness, Wenhao Guo, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Hanna Wallach, Stacey Carmichael, Mingyu Derek Yin, and Percy Liang. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114, 2012. URL `https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html`.

Teerath Kumar, Alessandra Mileo, Rob Brennan, and Malika Bendechache. Image data augmentation approaches: A comprehensive survey and future directions, 2023. URL `https://arxiv.org/abs/2301.02830`.

Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015. doi: 10.1038/nature14539.

Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *CoRR*, abs/1601.07996, 2016. URL `http://arxiv.org/abs/1601.07996`.

Petro Liashchynskyi and Pavlo Liashchynskyi. Grid search, random search, genetic algorithm: A big comparison for NAS. *CoRR*, abs/1912.06059, 2019. URL `http://arxiv.org/abs/1912.06059`.

Ruhan Liu, Xiangning Wang, Qiang Wu, Ling Dai, Xi Fang, Tao Yan, Jaemin Son, Shiqi Tang, Jiang Li, Zijian Gao, Adrian Galdran, J.M. Poorneshwaran, Hao Liu, Jie Wang, Yerui Chen, Prasanna Porwal, Gavin Siew Wei Tan, Xiaokang Yang, Chao Dai, Haitao Song, Mingang Chen, Huating Li, Weiping Jia, Dinggang Shen, Bin Sheng, and Ping Zhang. Deepdrid: Diabetic retinopathy—grading and image quality estimation challenge. *Patterns*, 3(6):100512, 2022a. ISSN 2666-3899. doi: https://doi.org/10.1016/j.patter.2022.100512. URL `https://www.sciencedirect.com/science/article/pii/S2666389922001040`.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496. ACL, 2019.

Yibin Liu, Shanshan Yong, Chunjiu He, Xin'an Wang, Zhenyu Bao, Jinhan Xie, and Xing Zhang. An earthquake forecast model based on multi-station pca algorithm. *Applied Sciences*, 12(7):3311, 2022b. doi: 10.3390/app12073311.

V Ljosa, KL Sokolnicki, and AE Carpenter. Annotated high-throughput microscopy image sets for validation. *Nature Methods*, 9(7):637, 2012. doi: 10.1038/nmeth.2083. URL `https://doi.org/10.1038/nmeth.2083`. Erratum in: Nat Methods. 2013 May;10(5):445.

Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017. URL `http://arxiv.org/abs/1711.05101`.

James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, and Angela Hung Byers. Big data: The next frontier for innovation, competition, and productivity. Technical report, McKinsey Global Institute, June 2011.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. The natural language decathlon: Multitask learning as question answering. In *arXiv preprint arXiv:1806.08730*, 2018.

John McCarthy. What is artificial intelligence? 2004.

Tom M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018. ISSN 1051-2004. doi: https://doi.org/10.1016/j.dsp.2017.10.011. URL `https://www.sciencedirect.com/science/article/pii/S1051200417302385`.

Michael Moor, Imon Banerjee, Zhiwen Fan Abad, Michael D Abrámoff, Babak Alipanahi, Katherine P Andriole, Amélie Arnaud, Hyun Jung Bae, Robyn L Ball, Mike Bastian, et al. Foundation models for generalist medical artificial intelligence. *Nature*, 616(7956):259–265, 2023.

Michael Nielsen. *Neural Networks and Deep Learning*. Self-published, 2015. URL http://neuralnetworksanddeeplearning.com/.

William Stafford Noble. What is a support vector machine? *Nature Biotechnology*, 24:1565–1567, 2006. doi: 10.1038/nbt1206-1565.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193.

Ravindra Parmar. Training deep neural networks, 2018. URL https://towardsdatascience.com/training-deep-neural-networks-9fdb1964b964. Accessed: 2024-10-22.

Narinder Singh Punn and Sonali Agarwal. Automated diagnosis of covid-19 with limited posteroanterior chest x-ray images using fine-tuned deep neural networks. *Applied Intelligence*, 2020. ISSN 1573-7497. doi: 10.1007/s10489-020-01900-3.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021a. URL https://arxiv.org/abs/2103.00020.

Alec Radford, Jong Wook Kim, Karthik Hallacy, et al. Learning transferable visual models from natural language supervision. *International Conference on Machine Learning*, 2021b.

Stuart J. Russell and Peter Norvig. *Artificial Intelligence - A Modern Approach, Third International Edition*. Pearson Education, 2010. ISBN 978-0-13-207148-2. URL http://vig.pearsoned.com/store/product/1,1207,store-12521_isbn-0136042597,00.html.

Francesco Di Salvo, Sebastian Doerrich, and Christian Ledig. Medmnist-c: Comprehensive benchmark and improved classifier robustness by simulating realistic image corruptions, 2024. URL https://arxiv.org/abs/2406.17536.

Iqbal H. Sarker. Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Comput. Sci.*, 2(6), August 2021a. doi: 10.1007/s42979-021-00815-1. URL https://doi.org/10.1007/s42979-021-00815-1.

Iqbal H. Sarker. Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science*, 2(3):160, 2021b. doi: 10.1007/s42979-021-00592-x.

Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. LAION-400M: open dataset of clip-filtered 400 million image-text pairs. *CoRR*, abs/2111.02114, 2021. URL `https://arxiv.org/abs/2111.02114`.

Motoharu Sonogashira, Michihiro Shonai, and Masaaki Iiyama. High-resolution bathymetry by deep-learning-based image superresolution. *PLOS ONE*, 15(7): e0235487, 2020. doi: 10.1371/journal.pone.0235487. URL `https://doi.org/10.1371/journal.pone.0235487`.

Konstantin Stacke, Gabriel Eilertsen, Jonas Unger, and Claes Lundström. Measuring domain shift for deep learning in histopathology. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 937–946, 2021.

Jeyan Thiyagalingam, Mallikarjun Shankar, Geoffrey Fox, and Tony Hey. Scientific machine learning benchmarks. *Nature Reviews Physics*, 4, 04 2022. doi: 10.1038/s42254-022-00441-7.

Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. Descriptor : The ham 10000 dataset , a large collection of multi-source dermatoscopic images of common pigmented skin lesions. 2018. URL `https://api.semanticscholar.org/CorpusID:263789934`.

Turing. Ultimate battle between deep learning and machine learning, 2024. URL `https://www.turing.com/kb/ultimate-battle-between-deep-learning-and-machine-learning`. Accessed: 2024-10-22.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, 2018.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, pages 3261–3275, 2019.

Dequan Wang, Xiaosong Wang, Lilong Wang, Mengzhang Li, Qian Da, Xiaoqiang Liu, Xiangyu Gao, Jun Shen, Junjun He, Tian Shen, Qi Duan, Jie Zhao, Kang Li, Yu Qiao, and Shaoting Zhang. Medfmc: A real-world dataset and benchmark for foundation model adaptation in medical image classification, 2023. URL `https://doi.org/10.6084/m9.figshare.c.6476047.v1`. Figshare Collection.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M. Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3462–3471, 2017. doi: 10.1109/CVPR.2017.369.

Ross Wightman. Pytorch image models, 2019. URL `https://github.com/huggingface/pytorch-image-models`. Accessed: 2024-12-14.

Stefano Woerner, Arthur Jaques, and Christian F. Baumgartner. A comprehensive and easy-to-use multi-domain multi-task medical imaging meta-dataset (medimeta), 2024. URL `https://arxiv.org/abs/2404.16000`.

Hanwen Xu, Naoto Usuyama, Jaspreet Bagga, Sheng Zhang, Rajesh Rao, Tristan Naumann, Cliff Wong, Zelalem Gero, Javier González, et al. A whole-slide foundation model for digital pathology from real-world data. *Nature*, 630:183–194, 2024. doi: 10.1038/s41586-024-06623-y.

Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2 - a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(1), January 2023. ISSN 2052-4463. doi: 10.1038/s41597-022-01721-8. URL `http://dx.doi.org/10.1038/s41597-022-01721-8`.

Matei Zaharia, Andy Chen, Ali Ghodsi, et al. Benchmarking ai – machine learning systems, 2024. URL `https://mlsysbook.ai`. Accessed: 2024-09-17.

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov Liu, Sebastian Ruder, and Lucas Beyer. A large-scale study of representation learning with the visual task adaptation benchmark. In *arXiv preprint arXiv:1910.04867*, 2019.

Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning*. Cambridge University Press, 2023a. `https://D2L.ai`.

Xinyu Zhang, Wei Chen, Haoyuan Li, Zheng Cao, and Hongyan Hu. Predicting and analyzing road traffic injury severity using boosting-based ensemble learning models with shapley additive explanations. *Accident Analysis Prevention*, 180: 106924, 2023b.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 30(5):881–898, 2017.

## Declaration of Authorship

Ich erkläre hiermit gemäß §9 Abs. 12 APO, dass ich die vorstehende Abschlussarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Des Weiteren erkläre ich, dass die digitale Fassung der gedruckten Ausfertigung der Abschlussarbeit ausnahmslos in Inhalt und Wortlaut entspricht und zur Kenntnis genommen wurde, dass diese digitale Fassung einer durch Software unterstützten, anonymisierten Prüfung auf Plagiate unterzogen werden kann.

| | |
|---|---|
| Place, Date | Signature |