OPEN ACCESS
OPUS AUGSBURG

Universität Augsburg
Universitätsbibliothek

An analysis on ensemble learning optimized medical image classification with deep convolutional neural networks

Dominik Müller, Inaki Soto-Rey, Frank Kramer

Angaben zur Veröffentlichung / Publication details:

Müller, Dominik, Inaki Soto-Rey, and Frank Kramer. 2022. "An analysis on ensemble learning optimized medical image classification with deep convolutional neural networks." *IEEE Access* 10: 66467–80. https://doi.org/10.1109/access.2022.3182399.

Nutzungsbedingungen / Terms of use: **CC BY 4.0**

Dieses Dokument wird unter folgenden Bedingungen zur Verfügung gestellt: / This document is made available under these conditions:
CC-BY 4.0: Creative Commons: Namensnennung
Weitere Informationen finden Sie unter: / For more information see:
https://creativecommons.org/licenses/by/4.0/deed.de

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 16 May 2022, accepted 1 June 2022, date of publication 13 June 2022, date of current version 27 June 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3182399

An Analysis on Ensemble Learning Optimized Medical Image Classification With Deep Convolutional Neural Networks

DOMINIK MÜLLER[1,2], IÑAKI SOTO-REY[2], AND FRANK KRAMER[1], (Associate Member, IEEE)
[1]IT-Infrastructure for Translational Medical Research, University of Augsburg, 86159 Augsburg, Germany
[2]Medical Data Integration Center, Institute for Digital Medicine, University Hospital Augsburg, 86156 Augsburg, Germany

Corresponding author: Dominik Müller (dominik.mueller@uni-a.de)

This work was supported in part by the DIFUTURE Project funded by the German Ministry of Education and Research (Bundesministerium für Bildung und Forschung, BMBF) under Grant FKZ01ZZ1804E.

ABSTRACT Novel and high-performance medical image classification pipelines are heavily utilizing ensemble learning strategies. The idea of ensemble learning is to assemble diverse models or multiple predictions and, thus, boost prediction performance. However, it is still an open question to what extent as well as which ensemble learning strategies are beneficial in deep learning based medical image classification pipelines. In this work, we proposed a reproducible medical image classification pipeline for analyzing the performance impact of the following ensemble learning techniques: Augmenting, Stacking, and Bagging. The pipeline consists of state-of-the-art preprocessing and image augmentation methods as well as 9 deep convolution neural network architectures. It was applied on four popular medical imaging datasets with varying complexity. Furthermore, 12 pooling functions for combining multiple predictions were analyzed, ranging from simple statistical functions like unweighted averaging up to more complex learning-based functions like support vector machines. Our results revealed that Stacking achieved the largest performance gain of up to 13% F1-score increase. Augmenting showed consistent improvement capabilities by up to 4% and is also applicable to single model based pipelines. Cross-validation based Bagging demonstrated significant performance gain close to Stacking, which resulted in an F1-score increase up to +11%. Furthermore, we demonstrated that simple statistical pooling functions are equal or often even better than more complex pooling functions. We concluded that the integration of ensemble learning techniques is a powerful method for any medical image classification pipeline to improve robustness and boost performance.

INDEX TERMS Medical image classification, ensemble learning, deep learning, medical imaging, stacking, bagging, test-time augmentation.

I. INTRODUCTION

The field of automated medical image analysis has seen rapid growth in recent years [1]–[3]. The utilization of deep neural networks became one of the most popular and widely applied algorithms for computer vision tasks [2]. A starting point for this trend relies on deep convolutional neural network architectures. These architectures demonstrated powerful prediction capabilities and achieved similar performance as clinicians [2], [4]. The integration of deep learning based automated medical image analysis in the clinical routine is currently a highly popular research topic. The subfield medical image classification (MIC) aims to label a complete image to predefined classes, e.g. to a diagnosis or a condition. The idea is to use these models as clinical decision support for clinicians in order to improve diagnosis reliability or automate time-consuming processes [2], [5].

Recent studies showed that the most successful and accurate MIC pipelines are also heavily based on ensemble learning strategies [6]–[12]. In the machine learning field, the aim is to find a suitable hypothesis that maximizes prediction correctness. However, finding the optimal hypothesis is difficult which is why the strategy was evolved to combine multiple hypotheses into a superior predictor closer to an optimal hypothesis. In the context of deep convolutional neural networks, hypotheses are represented through fitted

The associate editor coordinating the review of this manuscript and approving it for publication was Kumaradevan Punithakumar.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

neural network models. Thus, ensemble learning is defined as the combination of models to yield better prediction performance. The integration of ensemble learning strategies in a deep learning based pipeline is called deep ensemble learning. Various recent studies successfully utilized this strategy to improve the performance and robustness of their MIC pipeline [6]–[16]. The underlying techniques of these deep ensemble learning based pipelines are ranging from the combination of different model types like in the studies Rajaraman *et al.* [17] and Hameed *et al.* [11] to inference improvement of a single model like in Galdran *et al.* [18]. Furthermore, medical imaging datasets are commonly quite small, which is why ensemble learning techniques for efficient training data usage are especially popular as demonstrated in Ju *et al.* [13] and Müller *et al.* [19]. Empirically, ensemble learning based pipelines tend to be superior according to the assumption that the assembling of diverse models has the advantage to combine their strengths in focusing on different features whereas balancing out the individual incapability of a model [13], [20]–[22]. However, it is still an open question to what extent as well as which ensemble learning strategies are beneficial in deep learning based MIC pipelines. Even so, the field and idea of general ensemble learning is not novel, the impact of ensemble learning strategies in deep learning based classification has not been adequately analyzed in the literature, yet. Whereas multiple authors provide extensive reviews on general ensemble learning like Ganaiea *et al.* [22], only a handful of works started to survey the deep ensemble learning field. While Cao *et al.* reviewed deep learning based ensemble learning methods specifically in bioinformatics [23], Sagi and Rokach [24], Ju *et al.* [13], and Kandel *et al.* [25] started to provide descriptions or analysis on general deep ensemble learning methods.

In this study, we push towards setup a reproducible analysis pipeline to reveal the impact of ensemble learning techniques on medical image classification performance with deep convolution neural networks. By computing the performance of multiple ensemble learning techniques, we want to compare them to a baseline pipeline and, thus, identify possible performance gain. Furthermore, we explore the possible performance impact on multiple medical datasets from diverse modalities ranging from histology to X-ray imaging. Our experiments aim to help understand the beneficial as well as unfavorable influences of different ensemble learning techniques on model performance. This study contributes to the field of deep ensemble learning and provides the missing overview of state-of-the-art ensemble learning techniques for deep learning based MIC.

Our manuscript is organized as follows: Section 1 introduces medical image classification, the field of ensemble learning and our research question. In Section 2, we describe our proposed pipeline including the datasets, preprocessing methods, deep convolutional neural network architectures, ensemble learning strategies, and pooling functions. In Section 3, we report the experimental results and discuss these in detail in Section 4. In Section 5, we conclude our paper and give insights on future work. The Appendix contains further information on the availability of our trained models, all result data and the code used in this research.

II. METHODS AND MATERIALS

A. DATASETS

For increased result reliability and robustness, we analyzed multiple public MIC datasets. The datasets differ in sample size, modality, feature type of interest and noisiness. An overview of all datasets can be seen in Table 1, as well as exemplary samples in Figure 1.

1) CHMNIST

The image analysis of histological slides is an essential part in the field of pathology. The CHMNIST dataset consists of image patches generated from histology slides of patients with colorectal cancer [26], [27]. These patches were annotated in eight distinct classes: Tumor epithelium, simple stroma (homogeneous composition), complex stroma (containing single tumor cells and/or immune cells), immune cells, debris (including necrosis, hemorrhage and mucus), normal mucosal glands, adipose tissue and background (no tissue) [26], [27]. The dataset contains in total 5,000 images in Red-Green-Blue (RGB) color encoding with 625 images for each class and a unified resolution of 150 × 150 pixels. The slides were generated via an Aperio ScanScope microscope with a 20x magnification from the pathology archive of University Medical Center Mannheim and Heidelberg University [26].

2) COVID

X-ray imaging is one of the key modalities in the field of medical image analysis and is crucial in modern healthcare. Furthermore, X-ray imaging is a widely favored alternative to reverse transcription polymerase chain reaction testing for the coronavirus disease (COVID-19) [28], [29]. Researchers from Qatar, Doha, Dhaka, Bangladesh, Pakistan and Malaysia have created a dataset of thorax X-ray images for COVID-19 positives cases along with healthy control and other viral pneumonia cases [28]. The X-ray scans were gathered and annotated from 6 different radiographic databases or sources like the Italian Society of Medical and Interventional Radiology (SIRM) COVID-19 Database [28], [30]. The dataset consists of in total 2,905 grayscale images with 219 COVID-19 positive, 1,345 viral pneumonia and 1,341 control cases.

3) ISIC

Melanoma, appearing as pigmented lesions on the skin, is a major public health problem with more than new 300,000 cases per year and is responsible for the majority of skin cancer deaths [31]. Dermoscopy is the field of early melanoma detection, which can be either performed manually by expert visual inspection or automatically by MIC

TABLE 1. Overview of utilized datasets with descriptive details and sampling distributions. The noisiness of a dataset is a subjective impression based on the best achieved performance in the literature for this dataset.

Dataset	Modality	Noisiness	Classes	Number of Samples			
				model-train	model-val	ensemble-train	testing
CHMNIST	Histology	Small	8	3,250	501	500	749
COVID	X-ray	Small	3	1,889	291	290	435
ISIC	Dermoscopy	Medium	8	16,466	2,533	2,533	3,799
DRD	Ophthalmoscopy	Strong	5	22,832	3,513	3,513	5,268

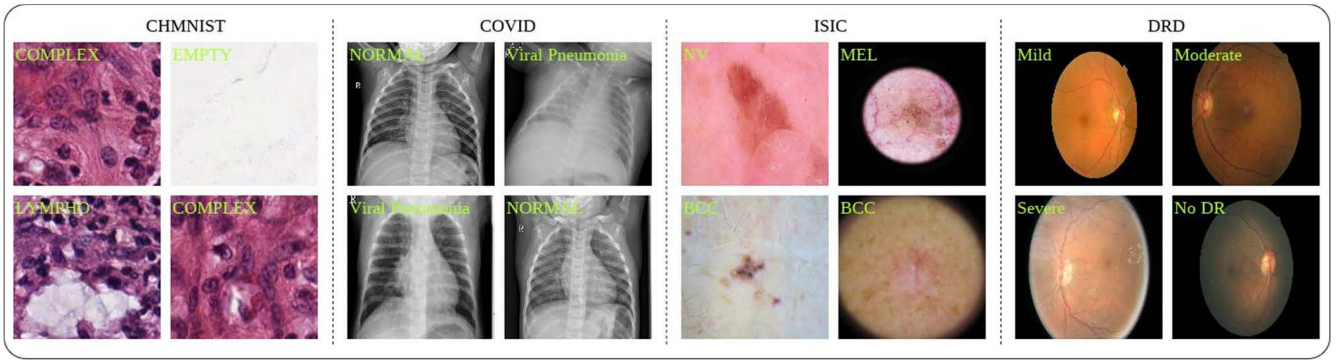

FIGURE 1. Exemplary sections of the four used datasets used in this analysis: CHMNIST (histology), COVID (X-ray), ISIC (dermoscopy) and DRD (ophthalmoscopy).

via high-resolution cameras. The International Skin Imaging Collaboration (ISIC) hosts the largest publicly available collection of quality-controlled images of skin lesions [31]. The 2019 release of their archive consists of in total 25,331 RGB images which were classified in the following 8 classes: Melanoma (MEL), melanocytic nevus (NV), basal cell carcinoma (BCC), actinic keratosis (AK), benign keratosis (BKL), dermatofibroma (DF), vascular lesion (VASC) and squamous cell carcinoma (SCC) [32]–[34].

4) DRD

Diabetic retinopathy is the leading cause of blindness and is estimated to affect over 93 million people worldwide [35]. The detection of diabetic retinopathy is mostly done via a time-consuming manual inspection by a clinician or ophthalmologist with the help of a fundus camera [19]. In order to contribute to research for automated diabetic retinopathy detection (DRD) algorithms, the California Healthcare Foundation and EyePACS created a public dataset consisting of 35,126 RGB fundus images [35], [36]. These were annotated in the following five classes according to disease severity: No DR, Mild, Moderate, Severe, Proliferative DR. It has to be noted that the authors pointed out the real-world aspect of this dataset which includes various types of noise like artifacts, out of focus, under-/overexposed images and incorrect annotations [35].

B. SAMPLING AND PREPROCESSING

In order to ensure a reliable evaluation of our models, we sampled each dataset with the following distribution strategy: For model training, 65% of each dataset was used (called 'model-train') whereas 10% of all samples were used as a validation set during the training process (called 'model-val') to allow validation monitoring for callback strategies. The only exception for this 'model-train' and 'model-val' sampling strategy occurred in the Bagging experiment, in which the two sets were combined and sampled according to a 5-fold cross-validation (75% in total of a dataset with 60% as training and 15% as validation for each fold). For possible training of ensemble learning pooling methods, another 10% of a total dataset was reserved (called 'ensemble-train'). For the final in detail evaluation on a separate hold-out set, the remaining 15% of each dataset was sampled as testing set (called 'testing').

We applied the following preprocessing methods for enhancement of the pattern-finding process of our deep learning models as well as to increase data variability. Our pipeline utilized extensive real-time (also called online-) image augmentation during the training phase to allow the model seeing novel and unique images in each epoch. The augmentation was performed with Albumentations [37] and consisted of the following techniques: Flipping, rotations as well as alterations in brightness, contrast, saturation, and hue. Furthermore, all images were squared padded for avoiding aspect ratio loss. In the posterior resizing, the image resolutions were reduced to the model architecture default input sizes, which were commonly 224 × 224 pixels except for EfficientNetB4 with 380 × 380, as well as InceptionResNetV2 and Xception with 299 × 299 pixels [38]–[40]. Before passing the images into the model, we applied value intensity normalization. The intensities were zero-centered via Z-Score normalization based on the mean and standard deviation computed on the ImageNet dataset [41].

TABLE 2. Overview of configurations for applied preprocessing techniques and neural network models in the presented medical image classification pipeline.

Preprocessing			
Online Image Augmentation		**Subfunctions**	
Flipping	Random Probability: 50%	Padding	To square ratio
Rotations	Random 90° Probability: 50%	Resize	To 224^2 or 299^2 or 380^2 (depending on architecture)
Brightness, Contrast, Saturation and Hue	-0.1 to +0.1 factor range Probability: 50%	Standardize	Z-Score normalization
Neural Network - General			
Loss	Focal Loss	Class Weights	Computed on train set via n_samples / (n_classes * bincount(y))
Activation Output	Softmax	Optimizer	Adam
Learning Rate	Initialized at 1e-04 for frozen-layer epochs Initialized at 1e-05 for unfrozen-layer epochs	Dynamic Learning Rate	Decreasing up to 1e-07 by a factor of 0.1 each time after 8 epochs without validation loss improvement
Epochs	Max. 1000	Batch size	28
Transfer Learning	ImageNet	Number of frozen Epochs	10
Model Checkpoints	Best computed loss on validation set (model-val) during training	Early Stopping	After 15 epochs without validation loss improvement
Training Monitoring	Tensorboard and CSV dumps for logging		
Neural Network - Architecture dependent Input Size			
DenseNet121	244 x 244 x 3	MobileNetV2	244 x 244 x 3
ResNeXt101	244 x 244 x 3	ResNet101	244 x 244 x 3
VGG16	244 x 244 x 3	EfficientNetB4	380 x 380 x 3
InceptionResNetV2	299 x 299 x 3	Xception	299 x 299 x 3
Vanilla	244 x 244 x 3		

C. DEEP LEARNING MODELS

For computer vision tasks like image classification, deep convolutional neural networks are state-of-the-art and unmatched in accuracy and robustness [5], [42]–[44]. Rather than focusing on a single model architecture for our analysis, we trained diverse classification architectures to ensure result reliability. The following architecture were selected: DenseNet121 [45], EfficientNetB4 [39], InceptionResNetV2 [40], MobileNetV2 [46], ResNeXt101 [47], ResNet101 [48], VGG16 [49], Xception [38] and a custom Vanilla architecture for comparison. The Vanilla architecture consisted of 4 convolutional layers with each followed by a max-pooling layer. The utilized classification head for all architectures applied a global average pooling, a dense layer with linear activation, a dropout layer, and another dense layer with a softmax activation function for the final class probabilities. The selected architectures represent the large diversity of popular and widely applied types of deep learning models for image classification. These strongly vary in the number of model parameters as well as neural network layers, input sizes, underlying composition techniques as well as functionality principles, and overall complexity. This allows a clearer analysis of the ensemble learning impact without architecture-related biases. In terms of general complexity, the utilized architectures have the following numbers of model parameters: Vanilla with $0.5{*}10^6$, DenseNet121 with $7.0{*}10^6$, EfficientNetB4 with $17.7{*}10^6$, InceptionResNetV2 with $54.3{*}10^6$, MobileNetV2 with $2.3{*}10^6$, ResNet101 with $42.7{*}10^6$, ResNeXt101 with $42.2{*}10^6$, VGG16 with $14.7{*}10^6$, and Xception with $20.8{*}10^6$. Further details on the architectures and their differences can be found in the excellent reviews of Bressem *et al.* [50] and Alzubaidi *et al.* [51]. For implementation, we used our in-house developed framework AUCMEDI which is built on TensorFlow [52]. Architecture and other implementation details are summarized in Table 2.

We utilized a transfer learning strategy by pretraining all models on the ImageNet dataset [41]. For the fitting process, the architecture layers were frozen at first except for the classification head and unfrozen, again, for fine-tuning. Whereas the frozen transfer learning phase was performed for 10 epochs using the Adam optimization with an initial learning rate of 1-E04, the fine-tuning phase stopped after a maximal training time of 1000 epochs (including the 10 epochs

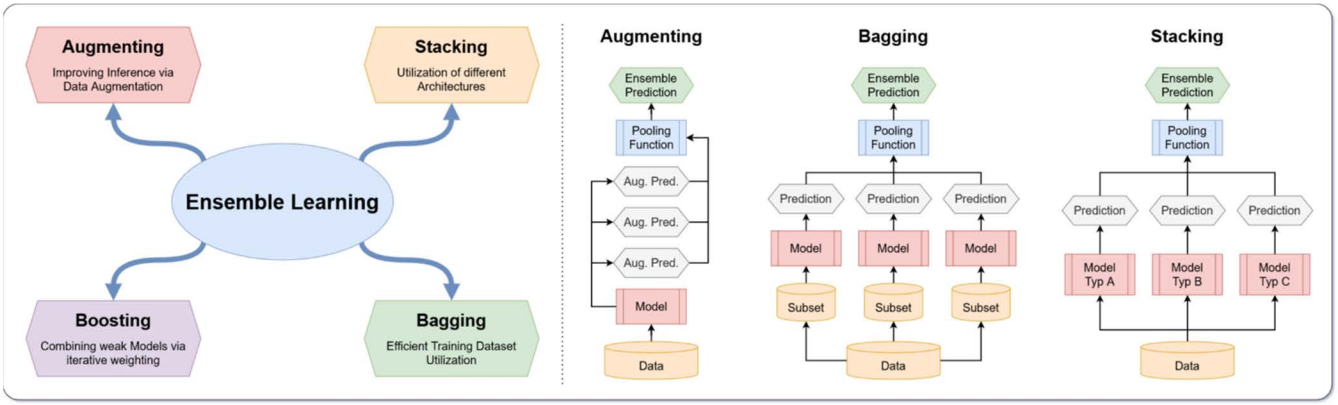

FIGURE 2. **Illustration showing the four ensemble learning techniques: Augmenting, Stacking, Boosting and Bagging. Except for Boosting, all techniques are commonly utilized in modern deep convolutional neural network pipelines.**

for transfer learning). The fine-tuning phase also utilized a dynamic learning rate for the Adam optimization [53] starting from 1-E05 to a maximum decrease to 1-E07 by a decreasing factor of 0.1 after 8 epochs without improvement on the monitored validation loss. As loss function for model training, we used the weighted Focal loss from Lin *et al.* [54].

$$\mathrm{FL}\,(p_t) = -\alpha_t\,(1-p_t)^{\gamma}\log(p_t) \tag{1}$$

In the above formula, p_t is the probability for the correct ground truth class t, γ a tunable focusing parameter (which we set to 2.0) and α_t the associated weight for class t [54]. The class weights were computed based on the class distribution in the corresponding 'model-train' sampling set. Furthermore, an early stopping and model checkpoint technique was applied for the fine-tuning phases, stopping after 15 epochs without improvement and saving the best model measured based on validation loss monitoring. The complete analysis was performed with a batch size of 28 and run parallelized on a workstation with 4x NVIDIA Titan RTX with each 24GB VRAM, Intel(R) Xeon(R) Gold 5220R CPU @ 2.20GHz with 96 cores and 384GB RAM.

D. ENSEMBLE LEARNING TECHNIQUES

As stated in the Introduction, deep ensemble learning is traditionally defined as building an ensemble of multiple predictions originating from different deep convolutional neural network models [22]. However, recent novel techniques necessitate redefining ensemble learning in the deep learning context as combining information, most commonly predictions, for a single inference. This information or predictions can either originate from multiple distinct models or just a single model. In this analysis, we explored the performance impact of the ensemble learning techniques: Augmenting, Bagging, and Stacking. We excluded the Boosting technique, which is also commonly used in general ensemble learning. The reason for this is that Boosting is not feasibly applicable for image classification with deep convolutional neural networks due to the extreme increase in training time [22], [24]. An overview diagram of the four techniques can be seen in Figure 2. For comparison, we setup Baseline models for all architectures to identify possible performance gain or loss tendencies through the ensemble learning techniques.

1) AUGMENTING

The Augmenting technique, often called test-time data augmentation, can be defined as the application of reasonable image augmentation prior to inference [55]–[60]. Through augmentation, multiple images of the same sample can be generated and then be used to compute multiple predictions. The aim of augmenting is to reduce the risk of incorrect predictions based on overfitting or too strict pattern learning [56], [57], [59]. In our analysis, we reused the Baseline models and applied random rotations as well as mirroring on all axes for inference. For each sample, 15 randomly augmented images were created, and their predictions were combined through an unweighted Mean as pooling function.

2) STACKING

In contrast to single algorithm approaches, the ensemble of different deep convolutional neural network architectures (also called inhomogeneous ensemble learning) showed strong benefits for overall performance [10], [22], [24], [25], [61]. This kind of ensemble learning is more complex and can consist of even different computer vision tasks [10], [22], [25]. The idea of the Stacking technique is to utilize these diverse and independent models by stacking another machine learning algorithm on top of these predictions. In our analysis, we reused the Baseline models consisting of various architectures as an ensemble for stacking the pooling functions directly on top of these inhomogeneous models.

3) BAGGING

Homogeneous model ensembles can be defined as multiple models consisting of the same algorithm, hyperparameters, or architecture [16], [22]. The Bagging technique is based on improved training dataset sampling and a popular

homogeneous ensemble learning technique. In contrast to a standard single training/validation split, which results in a single model, Bagging consists of training multiple models on randomly drawn subsets from the dataset. In practice, a k-fold cross-validation is applied on the dataset resulting in k models [62]. In our analysis, we applied a 5-fold cross-validation for Bagging as described in the sub-section 2.2 Sampling and Preprocessing, which resulted in five models for each architecture. The predictions of these five models for a single sample were combined via multiple pooling functions.

E. POOLING FUNCTIONS

In order to combine the ensemble of predictions into a single one, we studied several different methods and algorithms. A prediction consisted of the softmax normalized probability of each class for an unknown sample. For the Bagging and Stacking technique, the following pooling functions were analyzed: Best Model, Decision Tree, Gaussian Process classifier, Global Argmax, Logistic Regression, Majority Vote Soft and Hard, Unweighted and Weighted Mean, Naïve Bayes, Support Vector Machine, and k-Nearest Neighbors [63]. For the Augmenting technique, only the Unweighted Mean was used as pooling function.

Basic pooling functions were custom implemented, whereas more complex algorithms were integrated from scikit-learn [63]. The Best Model is selecting the best scoring model according to the F1 score on the 'ensemble-train' sampling set. Decision Trees were trained with Gini impurity as information gain function [64]. Gaussian Process classifier was based on Laplace approximation with a 'one-vs-rest' multi-class strategy. Global Argmax was defined as selecting the class with the highest probability across all predictions and zeroing the remaining classes. For Logistic Regression training, the 'newton-cg' solver and L2 regularization were used with a multinomial multi-class strategy [65]. The Majority Vote Soft variant sums up all probabilities per class and then softmax normalizes them across all classes, whereas the Majority Vote Hard variant utilizes traditional class voting in which the class with the highest probability is used for each prediction as vote. The Unweighted Mean straightforward averages the class probabilities across predictions, whereas the Weighted Mean performs a weighted averaging according to the achieved F1 score of the model on the 'ensemble-train' sampling set. The Naïve Bayes was implemented as the Complement variant described by Rennie *et al.* [66]. The Support Vector Machine classifier was based on the standard implementation from LIBSVM [67]. For the k-Nearest Neighbors classifier, a number of five neighbors was utilized.

F. EVALUATION

For evaluation, we utilized the packages pandas [68], scikit-learn [63], and plotnine [69] for visualization.

The performance scores were calculated class-wise and averaged by the unweighted mean. The following community-standard scores were used: Accuracy, F1-score, Sensitivity (also called True Positive Rate), False Positive Rate (FPR), and area under the receiver operating characteristic curve (AUC & ROC). The supplementary contains various additional metrics like Top-1/3-Error, Specificity, and others.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2}$$

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{3}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{4}$$

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

All metrics are based on the confusion matrix for binary classification, where TP, FP, TN, and FN represent the true positive, false positive, true negative, and false negative rate, respectively [70]. For the AUC and ROC curve computation, classifier confidence for predictions was also utilized [71].

III. RESULTS

The total training time of the complete analysis took around 1,215 hours with the following distribution per technique: Baseline 213 hours (17.5%), Augmenting 0.00 hours (0%), Stacking less than 0.09 hours (0%), and Bagging 1,002 hours (82.5%). It has to be noted that the Augmenting and Stacking techniques were based on the Baseline models but did not require extensive additional training time. The Baseline revealed an average training time by mean across all architectures of 45 minutes for COVID, 47 minutes for CHMNIST, 302 minutes for ISIC, and 1,026 minutes for DRD, whereas the Vanilla architecture had the lowest training time on average across datasets with 246 minutes and ResNet101 the highest with 522 minutes. Further details on training times for all architectures and phases can be found in the supplementary.

All training processes for the deep learning convolutional neural network models did not require the entire 1000 epochs and instead were early stopped after an average of 51 epochs. On median the epoch distribution looked like the following: For Baseline CHMNIST 54, COVID 48 ISIC 64, and DRD 37. For Bagging CHMNIST 53, COVID 48, ISIC 68, and DRD 37. Through validation monitoring during the training, no overfitting was observed. The training and validation loss function revealed no significant distinction from each other. The individual fitting plots for all models are attached in the Appendix.

A. BASELINE

The Baseline revealed the performance of various state-of-the-art architectures without the usage of any ensemble learning technique. This resulted in an average F1-score by a median of 0.95 for CHMNIST, 0.96 for COVID, 0.72 for ISIC, and 0.43 for DRD. The architectures shared overall a similar performance depending on the dataset noisiness. According to their F1-score, the best architectures were EfficientNetB4 and ResNet101 in CHMNIST, ResNeXt101 in COVID, ResNet101 and ResNeXt101 in ISIC, as well as

TABLE 3. Achieved results of the Baseline approach showing the Accuracy (Acc.), F1-score, Sensitivity (Sens.), and AUC on image classification for each architecture and dataset.

Method	CHMNIST				COVID				ISIC				DRD			
	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC
DenseNet121	0.99	0.95	0.95	1.0	0.98	0.97	0.97	1.0	0.95	0.72	0.76	0.96	0.84	0.43	0.48	0.78
EfficientNetB4	0.99	**0.96**	0.96	1.0	0.97	0.95	0.96	0.99	0.95	0.74	0.79	0.97	0.84	**0.45**	0.53	0.81
Inception-ResNetV2	0.99	0.95	0.95	1.0	0.98	0.96	0.96	1.0	0.95	0.72	0.74	0.96	0.84	0.42	0.5	0.78
MobileNetV2	0.98	0.9	0.9	0.99	0.97	0.95	0.96	0.99	0.94	0.68	0.73	0.95	0.82	0.32	0.41	0.72
ResNeXt101	0.99	0.95	0.95	1.0	0.99	**0.99**	0.98	1.0	0.96	**0.76**	0.78	0.97	0.83	0.43	0.52	0.79
ResNet101	0.99	**0.96**	0.96	1.0	0.98	0.98	0.98	1.0	0.95	**0.76**	0.79	0.97	0.84	**0.45**	0.48	0.79
VGG16	0.99	0.95	0.95	1.0	0.98	0.98	0.97	1.0	0.94	0.71	0.74	0.96	0.82	0.43	0.49	0.78
Vanilla	0.95	0.82	0.82	0.98	0.91	0.79	0.79	0.94	0.86	0.27	0.34	0.8	0.74	0.16	0.27	0.59
Xception	0.98	0.94	0.94	1.0	0.97	0.96	0.95	1.0	0.95	0.74	0.77	0.96	0.82	0.4	0.48	0.78

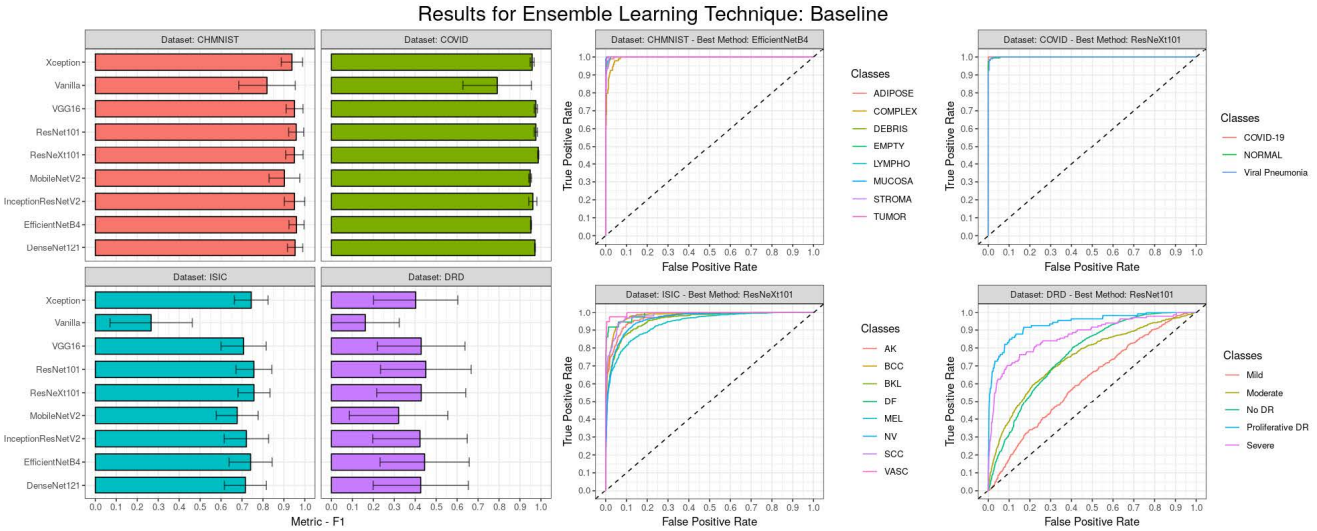

FIGURE 3. Performance results for the Baseline pipeline showing (LEFT) the achieved F1-score with its confidence interval on classification for each architecture and (RIGHT) class-wise ROC curves for the best performing architecture (according to F1-score) on each dataset.

EfficientNetB4 and ResNet101 in DRD. The smaller architectures like Vanilla and MobileNetV2 performed the worst. More details are shown in Table 3.

The ROC curves in Figure 3 revealed only marginal performance differences of classes in the CHMNIST, COVID, and ISIC dataset. However, DRD showed significant differences in Accuracy between classes whereas the detection of 'Mild' samples had the lowest performance.

B. AUGMENTING

By integrating the ensemble learning technique Augmenting for the inference based on the Baseline models, it was possible to obtain the following average F1-scores by median: 0.95 for CHMNIST, 0.97 for COVID, 0.74 for ISIC, and 0.43 for DRD. More details are shown in Table 4. Thus, there was only a marginal performance increase for the CHMNIST, and ISIC dataset compared to the Baseline. However, in the comparison of the best possible score between Augmenting and Baseline, a performance impact of 0% for CHMNIST, −1% for COVID, +3% for ISIC, and +4% for DRD was measured according to the F1-score.

The ranking between best-performing architectures revealed no drastic change. Especially, the EfficientNetB4 and ResNet101 achieved the highest performance similar to the Baseline, as well as the smaller architectures like Vanilla and MobileNetV2 the lowest. The ROC curves in Figure 4 resulted in equivalent model Accuracy variance between classes and datasets as the Baseline.

C. STACKING

For the Stacking technique, several pooling functions were successfully applied for combining the predictions of all Baseline architectures and resulted in the following average F1-scores by median: 0.96 for CHMNIST, 0.98 for COVID, 0.81 for ISIC, and 0.48 for DRD. More details are shown in Table 5. Compared with the median F1-score of

TABLE 4. Achieved results of the Augmenting approach showing the Accuracy (Acc.), F1-score, Sensitivity (Sens.), and AUC on image classification for each architecture and dataset.

Method	CHMNIST				COVID				ISIC				DRD			
	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC
DenseNet121	0.99	0.95	0.95	1.0	0.99	**0.98**	0.98	1.0	0.95	0.73	0.77	0.97	0.85	0.43	0.5	0.79
EfficientNetB4	0.99	**0.96**	0.96	1.0	0.97	0.95	0.95	0.99	0.96	**0.77**	0.81	0.97	0.85	0.45	0.55	0.82
Inception-ResNetV2	0.99	0.95	0.95	1.0	0.98	0.97	0.97	1.0	0.95	0.74	0.76	0.97	0.85	0.44	0.52	0.79
MobileNetV2	0.98	0.92	0.92	0.99	0.97	0.96	0.96	0.99	0.94	0.71	0.76	0.96	0.84	0.34	0.42	0.74
ResNeXt101	0.99	0.95	0.95	1.0	0.99	**0.98**	0.98	1.0	0.96	**0.77**	0.8	0.97	0.84	0.43	0.52	0.8
ResNet101	0.99	0.95	0.95	1.0	0.98	**0.98**	0.98	1.0	0.95	**0.77**	0.8	0.97	0.85	**0.47**	0.51	0.8
VGG16	0.99	0.95	0.95	1.0	0.98	**0.98**	0.97	1.0	0.95	0.72	0.75	0.96	0.83	0.45	0.51	0.79
Vanilla	0.95	0.82	0.82	0.98	0.9	0.77	0.8	0.93	0.86	0.27	0.35	0.8	0.74	0.16	0.26	0.59
Xception	0.99	0.95	0.95	1.0	0.98	0.97	0.97	1.0	0.95	0.76	0.77	0.97	0.83	0.42	0.5	0.8

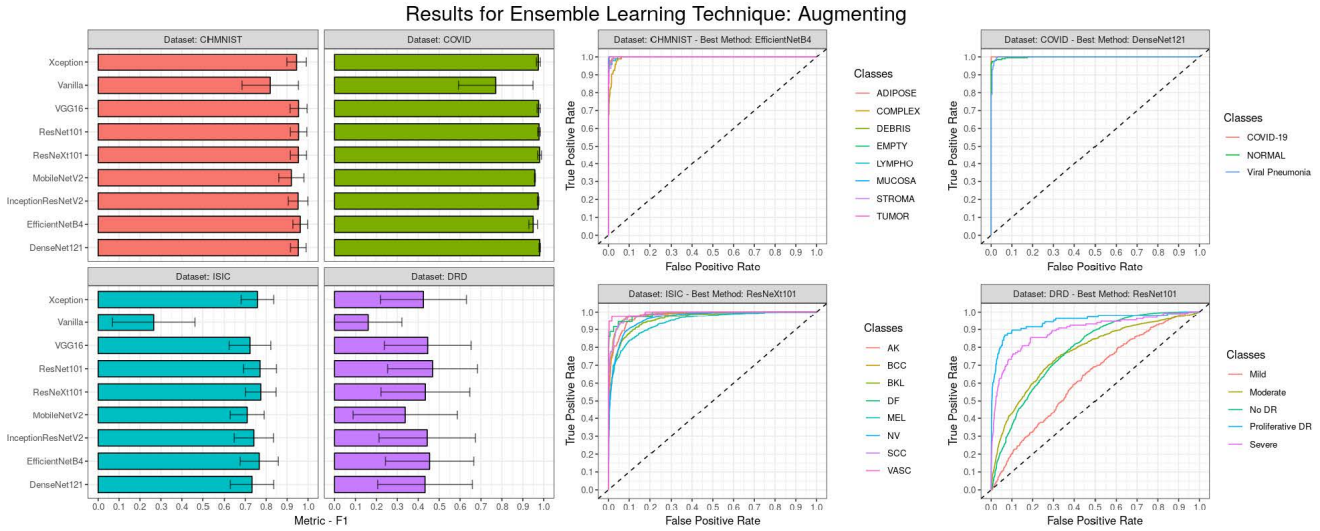

FIGURE 4. Performance results for the Augmenting approach showing (LEFT) the achieved F1-score with its confidence interval on classification for each architecture and (RIGHT) class-wise ROC curves for the best performing architecture (according to F1-score) on each dataset.

the Baseline, a performance impact of +1% for CHMNIST, +2% for COVID, +13% for ISIC, and +12% for DRD was measured. Additional to the median performance comparison, the pooling function 'Best Model' was also used as a benchmark without the usage of ensemble learning, which was inferior of up to 0.08 in Accuracy, 0.06 in F1, 0.06 in Sensitivity, and 0.04 in AUC compared with the best pooling function.

According to their F1-score, the best pooling functions were Naïve Bayes in CHMNIST, Majority Voting Soft, Mean Un-/Weighted, Naïve Byes and k-Nearest Neighbor in COVID, Gaussian Process, Majority Voting Soft, Mean Un-/Weighted and Support Vector Machine in ISIC, as well as Majority Voting Soft and Mean Un-/Weighted in DRD. The Decision Tree pooling function performed the worst and had a performance difference of up to −0.02 Accuracy, −0.09 F1, −0.09 Sensitivity, and −0.15 AUC compared to the 'Best Model' from the Baseline.

The ROC curves of the Stacking approach (illustrated in Figure 5) showed the same trend of class-wise performance differences as the Baseline, but with better precision results especially in the ISIC and DRD dataset.

D. BAGGING

By training new models based on a 5-fold cross-validation, it was possible to analyze the effects of Bagging on prediction capability. The predictions of five models per architecture were combined using various pooling functions. In this experiment, the five models of the EfficientNetB4 architecture archived the highest F1-scoring and were selected for further result reporting and representation of the Bagging approach. The evaluation of the merged predictions of these models showed the following averaged F1-score results by median: 0.96 for CHMNIST, 0.98 for COVID, 0.8 for ISIC, and 0.47 for DRD. In comparison with the Baseline, the following performance impact was measured: +1% for CHMNIST,

TABLE 5. Achieved results of the Stacking approach showing the Accuracy (Acc.), F1-score, Sensitivity (Sens.), and AUC on image classification for each technique and dataset.

Method	CHMNIST				COVID				ISIC				DRD			
	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC
Best Model	0.99	0.96	0.96	1.0	0.98	0.97	0.97	1.0	0.96	0.76	0.78	0.97	0.84	0.45	0.48	0.79
Decision Tree	0.98	0.94	0.94	0.97	0.98	0.96	0.97	0.98	0.94	0.67	0.69	0.82	0.87	0.41	0.42	0.64
Gaussian Process	0.99	0.96	0.96	0.99	0.99	0.98	0.98	1.0	0.96	**0.82**	0.79	0.97	0.92	0.48	0.44	0.83
Global Argmax	0.99	0.96	0.96	1.0	0.99	0.98	0.98	1.0	0.96	0.79	0.82	0.94	0.85	0.44	0.54	0.71
Logistic Regression	0.99	0.96	0.96	1.0	0.99	0.98	0.98	1.0	0.96	0.81	0.79	0.98	0.92	0.48	0.45	0.83
Majority Voting Hard	0.99	0.95	0.95	0.99	0.99	0.98	0.98	1.0	0.96	0.8	0.81	0.96	0.87	0.48	0.52	0.79
Majority Voting Soft	0.99	0.96	0.96	1.0	0.99	**0.99**	0.98	1.0	0.96	**0.82**	0.82	0.97	0.87	**0.49**	0.53	0.81
Mean Unweighted	0.99	0.96	0.96	1.0	0.99	**0.99**	0.98	1.0	0.96	**0.82**	0.82	0.98	0.87	**0.49**	0.53	0.82
Mean Weighted	0.99	0.96	0.96	1.0	0.99	**0.99**	0.98	1.0	0.96	**0.82**	0.82	0.98	0.87	**0.49**	0.53	0.82
Naive Bayes	0.99	**0.97**	0.97	0.99	0.99	**0.99**	0.98	1.0	0.96	0.81	0.81	0.97	0.9	0.36	0.41	0.79
Support Vector Machine	0.99	0.96	0.95	1.0	0.99	0.98	0.98	1.0	0.96	**0.82**	0.8	0.97	0.92	0.47	0.42	0.8
k-Nearest Neighbor	0.99	0.96	0.96	0.99	0.99	**0.99**	0.98	0.99	0.96	0.81	0.79	0.93	0.91	0.43	0.4	0.74

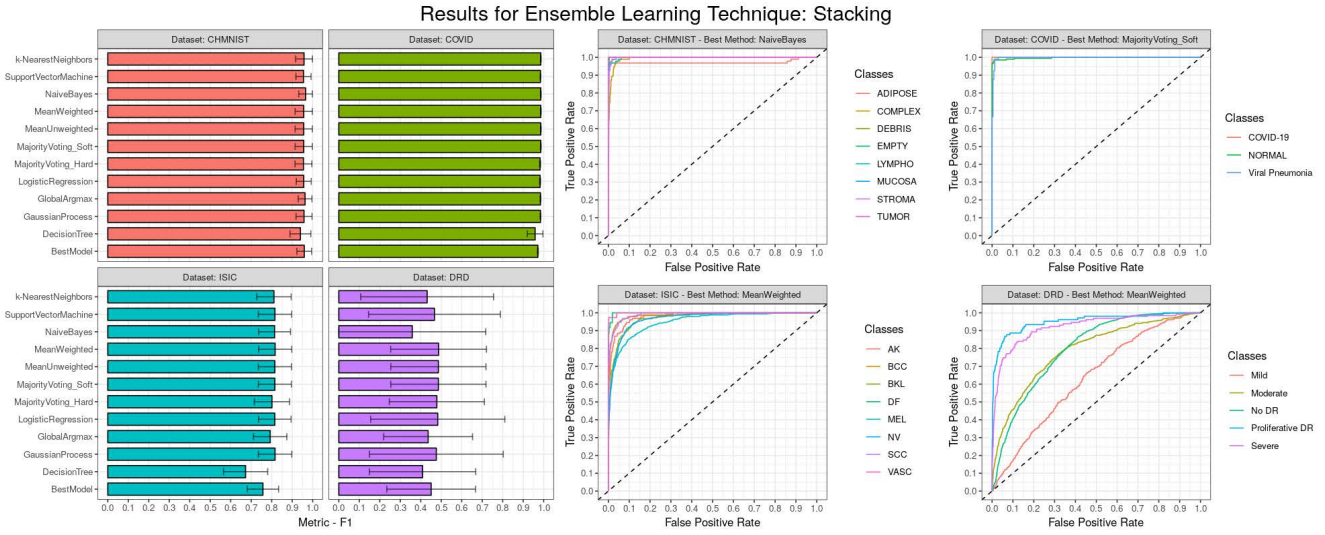

FIGURE 5. Performance results for the Stacking approach showing (LEFT) the achieved F1-score with its confidence interval on classification for each pooling function and (RIGHT) class-wise ROC curves for the best performing method (according to F1-score) on each dataset.

+2% for COVID, +11% for ISIC, and +9% for DRD. More details for the Bagging results can be seen in Table 6.

On the contrary to the previous ensemble learning approaches, the 'Best Model' pooling function represents not the best validation scoring Baseline model but instead the best model from the 5-fold cross-validation. The ranking between best-performing pooling functions for the EfficientNetB4 5-fold cross-validation revealed close grouping around the same score. In the CHMNIST and COVID set, all pooling functions except for Decision Trees achieved an F1-score of 0.96 and 0.98, respectively. Overall, the pooling based on Mean, Majority Voting, Gaussian Process, and Logistic Regression resulted in the highest performance on average. On the other hand, Decision Tree and Naïve Bayes obtained the lowest F1-scores.

In Figure 6, the ROC curves indicate an overall equal or superior performance compared to the Baseline, but a slightly inferior performance in CHMNIST dataset. Notably, the ISIC dataset indicates a stronger model Accuracy variance between classes compared to the Baseline ROC curves.

IV. DISCUSSION

In this work, we setup a reproducible pipeline for analyzing the impact of ensemble learning techniques on MIC performance with deep convolutional neural networks. We implemented Augmenting, Bagging as well as Stacking and compared them to a Baseline to compute performance gain on various metrics like F1-score, Sensitivity, AUC, and Accuracy. Our analysis proved that the integration of ensemble learning techniques can significantly boost classification

TABLE 6. Achieved results of the Bagging approach showing the Accuracy (Acc.), F1-score, Sensitivity (Sens.), and AUC on image classification for each technique and dataset. The Bagging technique was applied on the EfficientNetB4 architecture, which showed the highest F1-score performance.

Method	CHMNIST				COVID				ISIC				DRD			
	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC	Acc.	F1	Sens.	AUC
Best Model	0.99	**0.96**	0.96	1.0	0.99	**0.98**	0.98	1.0	0.95	0.76	0.78	0.97	0.86	0.46	0.53	0.82
Decision Tree	0.98	0.92	0.93	0.96	0.97	0.96	0.95	0.96	0.94	0.68	0.66	0.81	0.86	0.4	0.41	0.63
Gaussian Process	0.99	**0.96**	0.96	0.99	0.99	**0.98**	0.98	1.0	0.96	**0.81**	0.8	0.97	0.92	0.47	0.46	0.83
Global Argmax	0.99	**0.96**	0.96	0.99	0.99	**0.98**	0.98	1.0	0.96	0.78	0.82	0.95	0.84	0.45	0.56	0.72
Logistic Regression	0.99	**0.96**	0.96	0.99	0.99	**0.98**	0.98	1.0	0.96	**0.81**	0.8	0.97	0.92	0.47	0.46	0.84
Majority Voting Hard	0.99	**0.96**	0.96	0.98	0.99	**0.98**	0.98	0.99	0.96	0.8	0.81	0.94	0.85	**0.48**	0.55	0.77
Majority Voting Soft	0.99	**0.96**	0.96	1.0	0.99	**0.98**	0.98	1.0	0.96	0.8	0.82	0.97	0.85	**0.48**	0.56	0.81
Mean Unweighted	0.99	**0.96**	0.96	1.0	0.99	**0.98**	0.98	1.0	0.96	0.8	0.82	0.98	0.85	**0.48**	0.56	0.83
Mean Weighted	0.99	**0.96**	0.96	1.0	0.99	**0.98**	0.98	1.0	0.96	0.8	0.82	0.98	0.85	**0.48**	0.56	0.83
Naive Bayes	0.99	**0.96**	0.96	0.99	0.99	**0.98**	0.98	1.0	0.96	0.8	0.82	0.97	0.9	0.37	0.43	0.8
Support Vector Machine	0.99	**0.96**	0.96	0.99	0.99	**0.98**	0.98	1.0	0.96	0.79	0.76	0.95	0.92	0.46	0.43	0.79
k-Nearest Neighbor	0.99	**0.96**	0.95	0.99	0.99	**0.98**	0.98	0.99	0.96	0.79	0.78	0.92	0.91	0.44	0.43	0.74

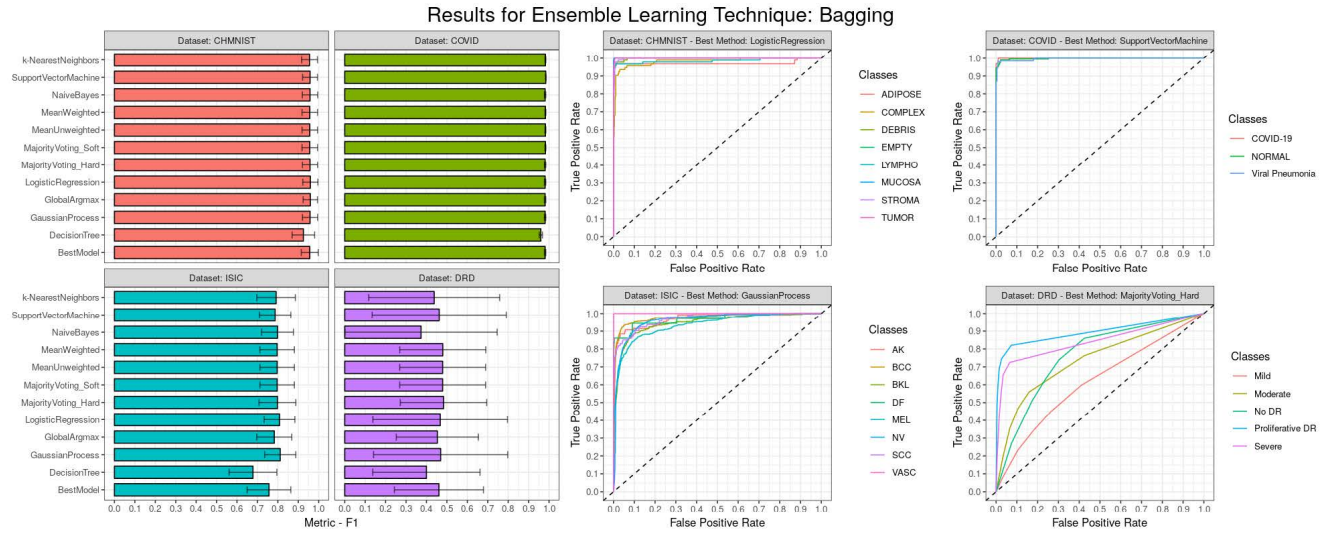

FIGURE 6. Performance results for the Bagging approach showing (LEFT) the achieved F1-score with its confidence interval on classification for each pooling function and (RIGHT) class-wise ROC curves for the best performing method (according to F1-score) on each dataset. The Bagging technique was applied on the EfficientNetB4 architecture, which showed the highest F1-score performance across all architectures.

performance from deep convolutional neural network models. As in Figure 7 summarized, our results showed a performance gain ranking from highest to lowest for the following ensemble learning techniques: Stacking, Bagging, and Augmenting.

The ensemble learning technique with the highest performance gain was Stacking, which applies pooling functions on top of different deep convolutional neural network architectures. Various state-of-the-art MIC pipelines heavily utilize a Stacking based pipeline structure to optimize performance by combining novel architectures or differently trained models [6], [10], [12], [19], [72]. This results in higher inference quality and bias or error reduction by using the prediction information of diverse methods. Our analysis also revealed that, according to F1-score results, simple pooling functions like averaging by Mean or a Soft Majority Vote results in an equally strong or even higher performance gain compared to more complex pooling functions like Support Vector Machines or Logistic Regressions. However, according to Accuracy results, the more complex pooling functions obtained higher scores. This indicates that more simple pooling functions are still based on the penalty strategy of the models which were trained with a class weighted loss function in our experiments. Thus, our results of simple pooling functions still optimize for class balanced metrics like F1-score or Sensitivity. On the other hand, more complex pooling functions with a separated training process focused on optimizing overall true cases including true negatives

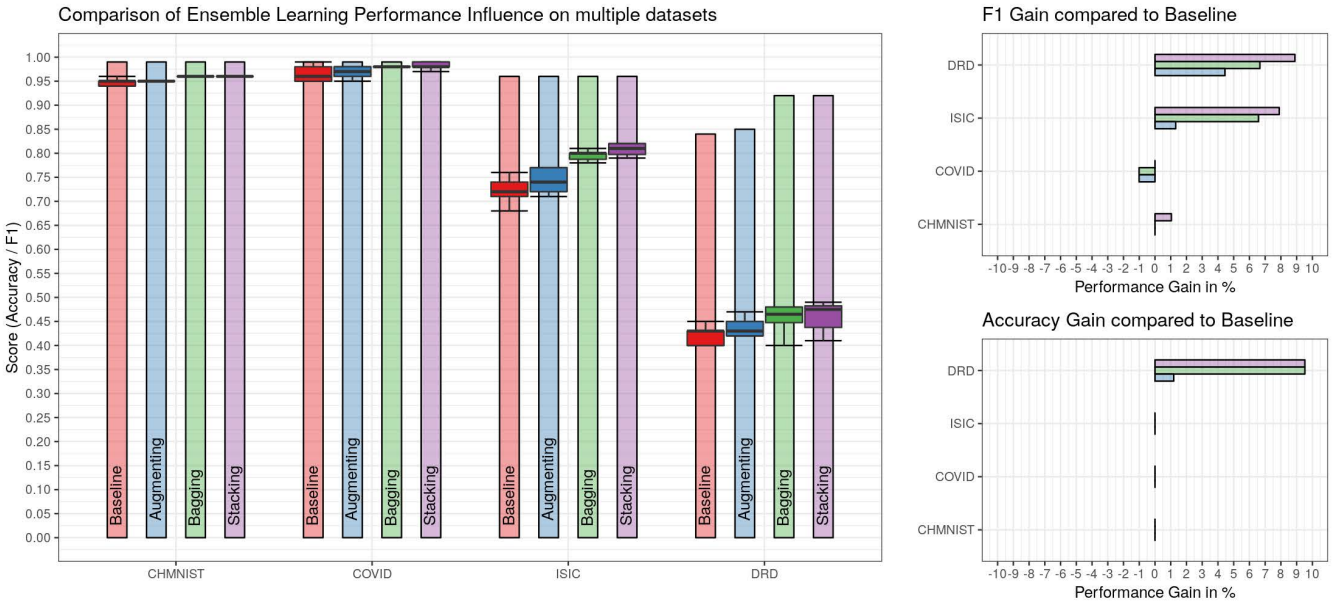

FIGURE 7. **Summary of all experiments to identify performance impact of ensemble learning techniques on medical image classification. LEFT: Bar plots showing the maximum achieved Accuracy across all methods for each ensemble learning technique and dataset: Baseline (red), Augmenting (blue), Bagging (green) and Stacking (purple). Additionally, the distribution of achieved F1-scores by the various methods is illustrated with box plots. RIGHT: Computed performance impact between the best scoring method of the Baseline and the best scoring method of the applied ensemble learning technique for each dataset. The performance impact is represented as performance gain in % between F1-scores (RIGHT TOP) as well as Accuracies (RIGHT BOTTOM). The color mapping of the ensemble learning techniques is equal to Figure 7 LEFT (Augmenting: Blue; Bagging: Green; Stacking: Purple).**

which resulted in better scoring on unbalanced metrics like Accuracy. Apart from that, other recent studies which analyzed the impact of Stacking also support our hypothesis that Stacking can significantly improve individual deep convolutional neural network model performance by up to 10% [7], [13], [22], [25]. With a similar experiment design as in our work, Kandel *et al.* demonstrated Stacking impact on a musculoskeletal fracture dataset analyzing pooling functions based on statistics as well as probability [25].

The Augmenting technique demonstrated to be an efficient ensemble learning approach. In nearly all our experiments, it was possible to improve the performance by another few percent through reducing overfitting bias in predictions. In theory, this should be already avoided with standard data augmentation during the training process. Although, our experiments indicated that the increased image variability through Augmenting could lead to adverse performance influences if applied on models based on small-sized datasets with a high risk of being overfitted. Especially in medical imaging, in which small datasets are common, this effect should be considered if Augmenting is applied and can also act as a strong indicator for overfitting. Nevertheless, strong performing MIC pipelines revealed that model performance can be significantly boosted with inference Augmenting [57]–[60]. Recent studies from Kandel *et al.* [59] and Shanmugam *et al.* [57] also analyzed the performance impact in detail of Augmenting on MIC and proved strong as well as consistent improvement, especially for low scoring models. In contrast to other ensemble learning techniques, Augmenting can be quickly integrated into pipelines without the need for additional training of various deep convolutional neural network or machine learning models. Thus, also a single model pipeline can benefit from this ensemble learning technique. However, performance gain from Augmenting is strongly influenced by applied augmentation methods and medical context in a dataset. Molchanov *et al.* tried to solve this issue with a greedy policy search to find the optimal Augmentation configuration [60]. A limitation of our analysis was that Augmenting was implemented and studied utilizing only an unweighted Mean as pooling function. However, other simple pooling functions without model training requirements like Global Argmax or Majority Vote Soft/Hard could have led to a stronger performance increase by Augmenting and should be analyzed as future work.

Nowadays, Bagging is one of the most widely used ensemble learning techniques and utilized in several state-of-the-art pipelines and top-performing benchmark submissions in MIC [11], [13], [14], [19], [22], [73]. In compliance, our experiments Bagging showed a strong performance increase for large datasets and no or marginal performance decrease in small datasets. Similar to Stacking, Bagging was able to significantly improve prediction capability for complex datasets like ISIC and DRD. We interpreted the possible detrimental effects in COVID and CHMNIST that the fewer data used for model training through cross-validation sampling had a considerable impact on performance in smaller datasets. Especially in small medical datasets with rare and unique morphological cases, excluding these can have a strong neg-

ative impact on performance. This is why our large datasets like ISIC and DRD with adequate feature presentations in all sampled folds revealed persistent performance improvement. Studies like Dwork *et al.* [74] analyzed this behavior and concluded that cross-validation based strategies comprise sustainable overfitting risk [75]. Based on our results, Bagging showed to have a high risk of drifting away from an optimal bias-variance tradeoff. According to Geman *et al.* [76], the bias-variance tradeoff is the right balance between bias and variance in a machine learning model in order to obtain the optimal generalizable model [76]. Whereas increased bias results into the risk of underfitting, increased variance can lead to overfitting. Cross-validation based Bagging boosts efficient data usage and, thus, the variance of a model. However, it has to be noted that the bias-variance tradeoff is still on active discussion in the research community for its correctness in deep learning [77], [78]. Furthermore, Bagging requires extensive additional training time to obtain multiple models. In the field of deep learning, training a higher number of models can lead to an extremely time-consuming process. For this reason, we specified our analysis on a 5-fold cross-validation. Still, our analysis results for Bagging are limited thereby based on the specification on using only 5-folds. Further research is needed on the impact of fold number or sampling size on performance and model generalizability in deep learning based MIC. Nevertheless, we concluded that Bagging is a powerful but complex to utilize ensemble learning technique and that its effectiveness is highly depended on sufficient feature representation in the sampled cross-validation folds. To avoid harmful folds with missing feature representation, we promote in-detail dataset analysis with manual annotation supported sampling (stratified) or using a higher k-fold to increase training sets and, thus, reduce the risk of excluding samples with unique morphological features.

V. CONCLUSION

In this paper, we analyzed the impact of the most widely used ensemble learning techniques on medical image classification performance: Augmenting, Stacking, and Bagging. We setup a reproducible experiment pipeline, evaluated the performance through multiple metrics, and compared these techniques with a Baseline to identify possible performance gain. Our results revealed that Stacking was able to achieve the largest performance gain in our medical image classification pipeline. Augmenting showed consistent improvement capabilities on non-overfitting models and has the advantage to be applicable to also single model based pipelines. Cross-validation based Bagging demonstrated significant performance gain close to Stacking, but reliant on sampling with sufficient feature representation in all folds. Additionally, we showed that simple statistical pooling functions like Mean or Majority Voting are equal or often even better than more complex pooling functions like Support Vector Machines. Overall, we concluded that the integration of ensemble learning techniques is a powerful method for MIC pipeline improvement and performance boosting. As a best practice, Stacking based pipeline builds utilizing multiple architectures showed continuous and strong performance improvement, whereas the gain of other ensemble learning techniques is based on datasets preconditions. As future research, we plan to further analyze the impact of the number of folds in cross-validation based Bagging techniques, integrate more pooling functions in Augmenting, and extend our analysis on deep learning Boosting approaches. Furthermore, the applicability of explainable artificial intelligence techniques for ensemble learning based medical image classification pipelines with multiple models is still an open research field and requires further research.

APPENDIX

The code for this article was implemented in Python (platform independent) and is available under the GPL-3.0 License at the following GitHub repository: https://github.com/frankkramer-lab/ensmic.

All data generated and analyzed during this study is available in the following Zenodo repository: https://doi.org/10.5281/zenodo.6457912.

ACKNOWLEDGMENT

The authors would like to thank Edmund Müller, Dennis Hartmann, Philip Meyer, and Peter Parys for their useful comments and support. They also want to thank Johannes Raffler and Ralf Huss for sharing their GPU hardware with them which was used for this work.

CONFLICTS OF INTEREST

None declared.

REFERENCES

[1] G. Wang, "A perspective on deep imaging," *IEEE Access*, vol. 4, pp. 8914–8924, 2016, doi: 10.1109/ACCESS.2016.2624938.

[2] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. W. M. van der Laak, B. van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Med. Image Anal.*, vol. 42, pp. 60–88, Dec. 2017, doi: 10.1016/j.media.2017.07.005.

[3] D. Shen, G. Wu, and H. Suk, "Deep learning in medical image analysis," *Annu. Rev. Biomed. Eng.*, vol. 19, pp. 221–248, Jun. 2017, doi: 10.1146/annurev-bioeng-071516-044442.

[4] K. Lee, J. Zung, P. Li, V. Jain, and H. S. Seung, "Superhuman accuracy on the SNEMI3D connectomics challenge," 2017, *arXiv:1706.00120*.

[5] M. Puttagunta and S. Ravi, "Medical image analysis based on deep learning approach," *Multimedia Tools Appl.*, vol. 80, no. 16, pp. 24365–24398, Jul. 2021, doi: 10.1007/s11042-021-10707-4.

[6] Y. Yang, Y. Hu, X. Zhang, and S. Wang, "Two-stage selective ensemble of CNN via deep tree training for medical image classification," *IEEE Trans. Cybern.*, early access, Mar. 11, 2021, doi: 10.1109/TCYB.2021.3061147.

[7] D. Xue, X. Zhou, C. Li, Y.-D. Yao, M. M. Rahaman, and J. Zhang, "An application of transfer learning and ensemble learning techniques for cervical histopathology image classification," *IEEE Access*, vol. 8, pp. 104603–104618, 2020, doi: 10.1109/ACCESS.2020.2999816.

[8] R. Logan, B. G. Williams, M. F. da Silva, A. Indani, N. Schcolnicov, A. Ganguly and S. J. Miller, "Deep convolutional neural networks with ensemble learning and generative adversarial networks for Alzheimer's disease image data classification," *Frontiers Aging Neurosci.*, vol. 13, p. 497, Aug. 2021, doi: 10.3389/fnagi.2021.720226.

[9] S. Rajaraman, J. Siegelman, P. O. Alderson, L. S. Folio, L. R. Folio, and S. K. Antani, "Iteratively pruned deep learning ensembles for COVID-19 detection in chest X-rays," *IEEE Access*, vol. 8, pp. 115041–115050, 2020. [Online]. Available: https://twitter.com/ChestImaging

[10] M. Mohammed, H. Mwambi, I. B. Mboya, M. K. Elbashir, and B. Omolo, "A stacking ensemble deep learning approach to cancer type classification based on TCGA data," *Sci. Rep.*, vol. 11, no. 1, p. 15626, Dec. 2021, doi: 10.1038/s41598-021-95128-x.

[11] Z. Hameed, S. Zahia, B. Garcia-Zapirain, J. J. Aguirre, and A. M. Vanegas, "Breast cancer histopathology image classification using an ensemble of deep learning models," *Sensors*, vol. 20, no. 16, p. 4373, Aug. 2020, doi: 10.3390/s20164373.

[12] A. Das, S. K. Mohapatra, and M. N. Mohanty, "Design of deep ensemble classifier with fuzzy decision method for biomedical image classification," *Appl. Soft Comput.*, vol. 115, Jan. 2022, Art. no. 108178, doi: 10.1016/j.asoc.2021.108178.

[13] C. Ju, A. Bibaut, and M. van der Laan, "The relative performance of ensemble methods with deep convolutional neural networks for image classification," *J. Appl. Statist.*, vol. 45, no. 15, pp. 2800–2818, Nov. 2018, doi: 10.1080/02664763.2018.1441383.

[14] D. Sułot, D. Alonso-Caneiro, P. Ksieniewicz, P. Krzyzanowska-Berkowska, and D. R. Iskander, "Glaucoma classification based on scanning laser ophthalmoscopic images using a deep learning ensemble method," *PLoS ONE*, vol. 16, no. 6, Jun. 2021, Art. no. e0252339, doi: 10.1371/journal.pone.0252339.

[15] Z. Qiao, A. Bae, L. M. Glass, C. Xiao, and J. Sun, *FLANNEL: Focal Loss Based Neural Network Ensemble for COVID-19 Detection*. New York, NY, USA: Oxford Univ. Press, Mar. 2021, doi: 10.1093/jamia/ocaa280.

[16] J. Zhang, Y. Wang, Y. Sun, and G. Li, "Strength of ensemble learning in multiclass classification of rockburst intensity," *Int. J. Numer. Anal. Methods Geomechanics*, vol. 44, no. 13, pp. 1833–1853, Sep. 2020, doi: 10.1002/nag.3111.

[17] S. Rajaraman, G. Zamzmi, and S. K. Antani, "Novel loss functions for ensemble-based medical image classification," *PLoS ONE*, vol. 16, no. 12, Dec. 2021, Art. no. e0261307, doi: 10.1371/journal.pone.0261307.

[18] A. Galdran, G. Carneiro, and M. A. González Ballester, "Balanced-mixup for highly imbalanced medical image classification," in *Proc. Med. Image Comput. Comput. Assist. Intervent. (MICCAI)* (Lecture Notes in Computer Science), vol. 12905. Cham, Switzerland: Springer, 2021, doi: 10.1007/978-3-030-87240-3_31.

[19] D. Müller, I. Soto-Rey, and F. Kramer, "Multi-disease detection in retinal imaging based on ensembling heterogeneous deep learning models," in *Stud. Health Technol. Informat.*, vol. 283, pp. 23–31, Sep. 2021, doi: 10.3233/SHTI210537.

[20] P. Sollich and A. Krogh, "Learning with ensembles: How over-fitting can be useful," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 1995, pp. 1–7, doi: doi/10.5555/2998828.2998855.

[21] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Mach. Learn.*, vol. 51, no. 2, pp. 181–207, May 2003, doi: 10.1023/A:1022859003006.

[22] M. A. Ganaie, M. Hu, A. K. Malik, M. Tanveer, and P. N. Suganthan, "Ensemble deep learning: A review," 2021, *arXiv:2104.02395*.

[23] Y. Cao, T. A. Geddes, J. Y. H. Yang, and P. Yang, "Ensemble deep learning in bioinformatics," *Nature Mach. Intell.*, vol. 2, no. 9, pp. 500–508, Sep. 2020, doi: 10.1038/s42256-020-0217-y.

[24] O. Sagi and L. Rokach, "Ensemble learning: A survey," in *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*. vol. 8, no. 4, Hoboken, NJ, USA: Wiley, Jul. 2018, doi: 10.1002/widm.1249.

[25] I. Kandel, M. Castelli, and A. Popovič, "Comparing stacking ensemble techniques to improve musculoskeletal fracture image classification," *J. Imag.*, vol. 7, no. 6, p. 100, Jun. 2021, doi: 10.3390/jimaging7060100.

[26] J. N. Kather, C.-A. Weis, F. Bianconi, S. M. Melchers, L. R. Schad, T. Gaiser, A. Marx, and F. G. Zöllner, "Multi-class texture analysis in colorectal cancer histology," *Sci. Rep.*, vol. 6, no. 1, pp. 1–11, Jun. 2016, doi: 10.1038/srep27988.

[27] J. N. Kather, F. G. Zöllner, F. Bianconi, S. M. Melchers, L. R. Schad, T. Gaiser, A. Marx, and C.-A. Weis, "Collection of textures in colorectal cancer histology [Data set]," Zenodo, 2016, doi: 10.5281/zenodo.53169.

[28] M. E. Chowdhury, T. Rahman, A. Khandakar, R. Mazhar, M. A. Kadir and Z. B. Mahbub, "Can AI help in screening viral and COVID-19 pneumonia?" *IEEE Access*, vol. 8, pp. 132665–132676, 2020, doi: 10.1109/ACCESS.2020.3010287.

[29] T. Rahman, A. Khandakar, Y. Qiblawey, A. Tahir, S. Kiranyaz, S. B. A. Kashem, M. T. Islam, S. Al Maadeed, S. M. Zughaier, M. S. Khan, and M. E. H. Chowdhury, "Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images," *Comput. Biol. Med.*, vol. 132, May 2021, Art. no. 104319, doi: 10.1016/j.compbiomed.2021.104319.

[30] Italian Society of Medical and Interventional Radiology. (2020). *COVID-19—Medical segmentation*. Accessed: May 2021 [Online]. Available: http://medicalsegmentation.com/covid19/

[31] J. Ferlay, M. Colombet, I. Soerjomataram, C. Mathers, D. M. Parkin, M. Piñeros, A. Znaor, and F. Bray, "Estimating the global cancer incidence and mortality in 2018: GLOBOCAN sources and methods," *Int. J. Cancer*, vol. 144, no. 8, pp. 1941–1953, Apr. 2019, doi: 10.1002/ijc.31937.

[32] M. Combalia, N. C. F. Codella, V. Rotemberg, B. Helba, V. Vilaplana, O. Reiter, C. Carrera, A. Barreiro, A. C. Halpern, S. Puig, and J. Malvehy, "BCN20000: Dermoscopic lesions in the wild," 2019, *arXiv:1908.02288*.

[33] N. C. F. Codella *et al.*, "Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (ISBI), hosted by the international skin imaging collaboration (ISIC)," in *Proc. IEEE 15th Int. Symp. Biomed. Imag. (ISBI)*, Apr. 2018, pp. 168–172.

[34] P. Tschandl, C. Rosendahl, and H. Kittler, "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Sci. Data*, vol. 5, no. 1, pp. 1–9, Aug. 2018, doi: 10.1038/sdata.2018.161.

[35] *Diabetic Retinopathy Detection | Kaggle*. Accessed: Oct. 2021. [Online]. Available: https://www.kaggle.com/c/diabetic-retinopathy-detection/overview

[36] J. Cuadros and G. Bresnick, "EyePACS: An adaptable telemedicine system for diabetic retinopathy screening," *J. Diabetes Sci. Technol.*, vol. 3, no. 3, pp. 509–516, May 2009, doi: 10.1177/193229680900300315.

[37] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, p. 125, Feb. 2020, doi: 10.3390/info11020125.

[38] F. Chollet, *Xception: Deep Learning With Depthwise Separable Convolutions*. Piscataway, NJ, USA: Institute of Electrical and Electronics Engineers, Nov. 2017, doi: 10.1109/CVPR.2017.195.

[39] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," 2019, *arXiv:1905.11946*.

[40] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Dec. 2016, pp. 2818–2826, doi: 10.1109/CVPR.2016.308.

[41] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, pp. 211–252, 2015, doi: 10.1007/s11263-015-0816-y.

[42] H. P. Chan, R. K. Samala, L. M. Hadjiiski, and C. Zhou, "Deep learning in medical image analysis," in *Advances in Experimental Medicine and Biology*. vol. 1213. Cham, Switzerland: Springer, 2020, pp. 3–21.

[43] L. Cai, J. Gao, and D. Zhao, "A review of the application of deep learning in medical image classification and segmentation," *Ann. Transl. Med.*, vol. 8, no. 11, p. 713, Jun. 2020, doi: 10.21037/atm.2020.02.44.

[44] A. Singh, S. Sengupta, and V. Lakshminarayanan, "Explainable deep learning models in medical image analysis," *J. Imag.*, vol. 6, no. 6, p. 52, Jun. 2020, doi: 10.3390/jimaging6060052.

[45] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jan. 2017, pp. 2261–2269.

[46] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jan. 2018, pp. 4510–4520.

[47] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Nov. 2017, pp. 5987–5995, doi: 10.1109/CVPR.2017.634.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[49] K. Simonyan and A. Zisserman, (Sep. 2015). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. Accessed: Dec. 17, 2021. [Online]. Available: http://www.robots.ox.ac.U.K./

[50] K. K. Bressem, L. C. Adams, C. Erxleben, B. Hamm, S. M. Niehues, and J. L. Vahldiek, "Comparing different deep learning architectures for classification of chest radiographs," *Sci. Rep.*, vol. 10, no. 1, p. 13590, Dec. 2020, doi: 10.1038/s41598-020-70479-z.
[51] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, p. 53, Dec. 2021, doi: 10.1186/s40537-021-00444-8.
[52] M. Abadi *et al.*, (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: https://www.tensorflow.org/
[53] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[54] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Aug. 2017.
[55] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, pp. 1–48, Dec. 2019, doi: 10.1186/s40537-019-0197-0.
[56] J. Calvo-Zaragoza, J. R. Rico-Juan, and A.-J. Gallego, "Ensemble classification from deep predictions with test data augmentation," *Soft Comput.*, vol. 24, no. 2, pp. 1423–1433, Jan. 2020, doi: 10.1007/s00500-019-03976-7.
[57] D. Shanmugam, D. Blalock, G. Balakrishnan, and J. Guttag, "Better aggregation in test-time augmentation," 2020, *arXiv:2011.11156*.
[58] M. Seçkin Ayhan and P. Berens, "Test-time data augmentation for estimation of heteroscedastic aleatoric uncertainty in deep neural networks," in *Proc. Int. Conf. Med. Imag. Deep Learn.*, 2018. [Online]. Available: https://openreview.net/pdf?id=rJZz-knjz
[59] I. Kandel and M. Castelli, "Improving convolutional neural networks performance for image classification using test time augmentation: A case study using Mura dataset," *Health Inf. Sci. Syst.*, vol. 9, no. 1, p. 33, Dec. 2021, doi: 10.1007/s13755-021-00163-7.
[60] D. Molchanov, A. Lyzhov, Y. Molchanova, A. Ashukha, and D. Vetrov, "Greedy policy search: A simple baseline for learnable test-time augmentation," in *Proc. 36th Conf. Uncertainty Artif. Intell. (UAI PMLR)*, 2020, pp. 1308–1317.
[61] Y. Yang, Y. Hu, X. Zhang, and S. Wang, "Two-stage selective ensemble of CNN via deep tree training for medical image classification," *IEEE Trans. Cybern.*, early access, Mar. 11, 2021, doi: 10.1109/TCYB.2021.3061147.
[62] B. Ghojogh and M. Crowley, "The theory behind overfitting, cross validation, regularization, bagging, and boosting: Tutorial," 2019, *arXiv:1905.12787*.
[63] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.
[64] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
[65] C. J. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. Math. Softw.*, vol. 23, no. 4, pp. 550–560, 199, doi: 10.1145/279232.279236.
[66] J. D. M. Rennie, L. Shih, J. Teevan, and D. R. Karger, "Tackling the poor assumptions of naive Bayes Text classifiers," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, pp. 616–623.
[67] C.-C. Chang and C.-J. Lin, *LIBSVM: A Library for Support Vector Machines*. Accessed: Nov. 2021. [Online]. Available: www.csie.ntu.edu.tw/
[68] W. Mckinney, "Data structures for statistical computing in Python," in *Proc. 9th Python Sci. Conf.*, 2010, pp. 51–56.
[69] H. Kibirige *et al.*, "has2k1/plotnine: v0.8.0 (v0.8.0)," Zenodo, 2021, doi: 10.5281/zenodo.4636791.
[70] J. Lever, M. Krzywinski, and N. Altman, "Classification evaluation," *Nature Methods*, vol. 13, no. 8, pp. 603–604, Jul. 2016, doi: 10.1038/nmeth.3945.
[71] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006, doi: 10.1016/j.patrec.2005.10.010.
[72] E. Ahn, A. Kumar, D. Feng, M. Fulham, and J. Kim, "Unsupervised feature learning with k-means and an ensemble of deep convolutional neural networks for medical image classification," 2019, *arXiv:1906.03359*.
[73] Y. Liu and F. Long, *Acute Lymphoblastic Leukemia Cells Image Analysis With Deep Bagging Ensemble Learning* (Lecture Notes in Bioengineering). Cham, Switzerland: Springer, 2019, pp. 113–121.
[74] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth, "Generalization in adaptive data analysis and holdout reuse," in *Proc. Adv. Neural Inf. Process. Syst.*, Jun. 2015, pp. 2350–2358.
[75] Y. Xu and R. Goodacre, "On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning," *J. Anal. Test.*, vol. 2, no. 3, pp. 249–262, Jul. 2018, doi: 10.1007/s41664-018-0068-2.
[76] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural Comput.*, vol. 4, no. 1, pp. 1–58, Jan. 1992, doi: 10.1162/neco.1992.4.1.1.
[77] Z. Yang, Y. Yu, C. You, J. Steinhardt, and Y. Ma, "Rethinking bias-variance trade-off for generalization of neural networks," in *Proc. 37th Int. Conf. Mach. Learn. (ICML)*, Feb. 2020, pp. 10698–10708.
[78] B. Neal, S. Mittal, A. Baratin, V. Tantia, M. Scicluna, S. Lacoste-Julien, and I. Mitliagkas, "A modern take on the bias-variance tradeoff in neural networks," 2018, *arXiv:1810.08591*.

DOMINIK MÜLLER received the bachelor's degree in bioinformatics from the Technische Universität München and the master's degree in bioinformatics from Ludwig-Maximilians-Universität München, Germany, in 2018. He is currently pursuing the Ph.D. degree in medical informatics with the University of Augsburg, Germany. His research is about automated medical image classification and segmentation with a focus on application, automatization, reproducibility, and simplification via framework development.

IÑAKI SOTO-REY received the bachelor's degree in computer engineering from the Complutense University in Madrid, Spain, the master's degree in computer engineering from the University of Oulu, Finland, and the Ph.D. degree in the field of medical informatics from the University of Münster, Germany, focusing his research on the secondary use of electronic health records for medical research and the evaluation of eHealth systems. He currently leads the Data Integration Center, University Hospital Augsburg, Germany.

FRANK KRAMER (Associate Member, IEEE) received the B.S. and M.S. degrees in computer science from the University of Erlangen, Germany, in 2009, and the Ph.D. degree in biomedical informatics from the University of Göttingen, Germany, in 2014. He is currently a Professor of medical informatics with the University of Augsburg, Germany. His research has been focused on the integration of clinical data, biomedical prior knowledge, and omics data for systems medicine as well as the development of infrastructures for data integration in clinical research with regard to personalized medicine.