Building Ensemble of Deep Networks: Convolutional Networks and Transformers

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 18 October 2023, accepted 27 October 2023, date of publication 6 November 2023, date of current version 10 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3330442

RESEARCH ARTICLE

Building Ensemble of Deep Networks: Convolutional Networks and Transformers

LORIS NANNI[1], ANDREA LOREGGIA[2], LEONARDO BARCELLONA[1], AND STEFANO GHIDONI[1]

[1]Department of Information Engineering (DEI), University of Padova, 35122 Padua, Italy
[2]Department of Information Engineering (DII), University of Brescia, 25123 Brescia, Italy

Corresponding author: Loris Nanni (loris.nanni@unipd.it)

This work was supported by NVIDIA through the Graphics Processing Unit (GPU) Grant Program.

ABSTRACT This paper presents a study on an automated system for image classification, which is based on the fusion of various deep learning methods. The study explores how to create an ensemble of different Convolutional Neural Network (CNN) models and transformer topologies that are fine-tuned on several datasets to leverage their diversity. The research question addressed in this work is whether different optimization algorithms can help in developing robust and efficient machine learning systems to be used in different domains for classification purposes. To do that, we introduce novel Adam variants. We employed these new approaches, coupled with several CNN topologies, for building an ensemble of classifiers that outperforms both other Adam-based methods and stochastic gradient descent. Additionally, the study combines the ensemble of CNNs with an ensemble of transformers based on different topologies, such as Deit, Vit, Swin, and Coat. To the best of our knowledge, this is the first work in which an in-depth study of a set of transformers and convolutional neural networks in a large set of small/medium-sized images is carried out. The experiments performed on several datasets demonstrate that the combination of such different models results in a substantial performance improvement in all tested problems. All resources are available at https://github.com/LorisNanni.

INDEX TERMS Convolutional neural networks, transformers, optimization, ensemble.

I. INTRODUCTION

The use of Convolutional Neural Networks (CNN) has revolutionized the field of image recognition, achieving impressive results in a wide range of applications [20]. Researchers have explored various ways to improve the architecture of these networks, including the combination of specialized layers to form new topologies. However, as neural networks become deeper, they are susceptible to problems such as the vanishing gradient and difficulties with optimization. In addition to enhancing the architecture of CNNs, finding robust and stable optimization algorithms is equally important to maximize performance [16]. Traditional optimization methods such as Gradient Descent (GD) and Stochastic Gradient Descent (SGD) have been widely used, with Adam being the more popular method due to its ability to improve the network's search for a solution. This has led to the development of many Adam-based variants. In this study, we propose new Adam-based variants and compare them to other optimization methods. We also explore the potential of transformers for building ensembles and compare different transformer topologies. Our experiments show that ensembles based on the combination of different approaches outperform state-of-the-art results in all tested problems. Despite the complexity of the proposed system, it requires minimal parameter tuning and works well in various problems without the need for specific pre-processing or optimization for each dataset. All the tested code and datasets are available for reproducibility.

The main contributions of this article are as follows:

i) We propose a set of new Adam-based variants useful during the training phase of the models.

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

ii) We define ensembles of different CNN models, trained with these new Adam-based variants, and different transformer topologies. Compared with existing machine learning methods, our ensembles provide competitive results in different domains.

iii) We provide an empirical evaluation of ensembles trained with standard and new Adam-based variants. Evaluating the performance on several metrics and datasets, we demonstrate that adopting different optimization algorithms and different network models is beneficial for ensembles.

iv) We introduce a testbed for evaluating Transformers ensembles, leveraging publicly accessible datasets, and showcasing a baseline ensemble. As a result, other researchers can readily compare their ensembles with our reported baseline, under the condition that they employ the same transformers with identical hyperparameters.

The remainder of the paper is structured as follows: Section II reports related works on the topic analyzed in this work. Section III introduces some basic notions about transformers, CNNs, the Adam approaches proposed in this work, as well as the datasets used for experiments. Section IV describes the experimental environments, the testing protocols, and the performance indicators; moreover, we suggest and discuss a set of experiments to evaluate our ensembles. In Section V, we discuss the value of our work providing more insights about the usefulness of the proposed approach. Section VI concludes this work and provides some further perspective on this research.

II. RELATED WORK

A. TRANSFORMERS IN IMAGE CLASSIFICATION

In image classification, Convolutional Neural Networks (CNNs) have been state-of-the-art for many years, and models such as ResNet [20] or EfficientNet [48] are still considered baseline approaches in many works. However, the recent introduction of transformers by Vaswani et al. [47] has changed this trend. Transformer models have achieved disruptive performance in the neural language processing field, making it the leading approach [6], [12], [28]. The improvement is attributed to the multi-head self-attention. After the encoding of words in tokens, the self-attention is computed as:

$$Attention(Q, K, V) = Softmax\left(\frac{QK^T}{\sqrt{D_k}}\right)V, \qquad (1)$$

where $Q \in R^{NxD_k}$ is the query matrix, $K \in R^{MxD_k}$ is key matrix and $V \in R^{MxD_v}$ is the value matrix. D_k is the dimension of the keys, with N and M denoting the lengths of queries and keys. The multi-head self-attention uses H different heads computing the attention, as in Eq. 1, and concatenating the output.

In image classification, Dosovitskiy et al. [14] introduced the Vision-Transformer (ViT) as the first pure transformer model to achieve results equivalent or superior to CNNs. ViT utilizes the encoder from Vaswani's original work [47] and stacks a multilayer perceptron to classify features extracted by the encoder. In this approach, the input tokens are image patches encoded in a latent space. However, training the ViT network is challenging due to its lack of inductive biases compared to CNNs [14], [46]. Moreover, the network struggles to aggregate local information. To address these challenges, ViT requires a pre-training on a dataset with millions of images, namely JFT-300M dataset.

A solution to ViT data issues is proposed in [46]. Their approach involves distilling knowledge acquired from a convolutional neural network, combined with actual ground-truth values, to enable transformers to learn the inductive bias from the distillation. Additionally, the authors introduced a "distillation token" to enhance the distillation process and lead to better performance with simpler training. The model is named Data-efficient Transformer (DeiT).

Swin [29], one of the top transformer models inspired by ViT, utilizes a hierarchical window structure to address scale-variation issues. The attention is computed within the local window, resulting in improved performance and execution time. The partitioning is gradually shifted along the network hierarchy to maintain interaction among different windows. The authors also demonstrated the effectiveness of Swin as a backbone in various vision tasks, including semantic segmentation.

Transformers have different characteristics compared to CNNs. According to [38], CNN models make decisions mainly based on texture, while ViT weights the shape more. Therefore, some authors have considered merging the two. An example is CoAtNet [10], which merges depthwise convolution with self-attention by stacking them vertically. With this approach, the model leverages CNNs inductive bias, Transformers capacity, and attention mechanism. CoAtNet may be considered a hybrid approach since it adopts a convolutional structure while inserting self-attention from [47].

B. ENSEMBLE OF IMAGE TRANSFORMERS

Vision transformers are gaining more attention for their ability to recognize long-range dependencies and their capacity [19], [27]. Despite the impressive results obtained by deep learning models, their performance can be further boosted through an ensemble of different models [26] or the same model with different weights [31], [40]. Each model of the ensemble can learn a representation that concentrates on particular aspects or representations of images, improving the final prediction, as demonstrated by the work of Savelli et al. [40] that improved small lesion detection using an ensemble of CNNs trained on different views of the same lesion. Kyathanahally et al. [26] used an ensemble of six CNNs to improve plankton classification by averaging the predictions of the components, following the demonstration by d'Ascoli et al. [18] that an average ensemble can reduce overfitting.

Despite the impact that vision transformers have had on image classification, only a few works have exploited their benefits in an ensemble approach [25], [35], [41]. In [21] the authors compared a vision transformer ensemble with a CNN ensemble for brain tumors. However, they found that the ensemble of vision transformer models performed worse than the convolutional ensemble. In [41] the authors proposed a model that exploits both ViT and convolutions, named Cvit, and created an ensemble of two models trained on the frequency domain and time domain to classify movement from EMG signals. The authors compared the proposed solution only with convolutional baselines and ViT. Recent work from Kyathanahally et al. [25] showed astonishing results in image classification among different datasets with an ensemble of three DeiT models. In [35], authors achieve enhanced ensemble performance for polyp segmentation by combining diverse transformer models (including HarDNet-MSEG, Polyp-PVT, and HSNet) through distinct training techniques and introducing a novel mask averaging method, resulting in superior segmentation results across multiple datasets. ETECADx [1] is an AI-based computer-aided diagnosis (CAD) framework for early breast cancer detection, which combines convolutional neural networks and the self-attention mechanism of vision transformer encoder. Another study [2] introduces a masked face recognition system, combining two Convolutional Neural Network (CNN) models and two Transformer models, which achieve better accuracy when ensembled, outperforming other models in recognizing masked faces.

In our work, we propose to combine different transformers and convolutional models, demonstrating their superiority on various datasets. Each model contributes to the final prediction through the weighted sum rule. Thus, the vision Transformers and CNNs are trained separately and their outputs are merged together. Similarly to [2] and [25], we find that Transformer models, exploiting the attention mechanism, boost the ensemble results when merged with CNNs.

C. ADAM OPTIMIZERS

A fundamental role in deep learning is played by optimizers. An optimizer in machine learning is a mathematical algorithm or technique that adjusts the internal parameters of a model to minimize or maximize a specified objective function, improving the model's performance on a given task. One of the most used optimizers is Adam [23]. It is used for gradient-based optimization of stochastic objective functions. It is a combination of two other optimization algorithms, namely, Adagrad [17] and RMSprop [23]. The key idea behind Adam is to adaptively estimate the first and second moments of the gradients in order to perform effective optimization.

Some authors proposed variations of Adam, creating new optimizers. An example is DiffGrad [16], proposed in 2019. It relies on the assumption that a decrease in the rate of gradient variations suggests the presence of a global minimum. The objective is to generate substantial strides when the gradient is undergoing large changes while taking smaller steps when the gradient is changing more gradually. Overall, DiffGrad has shown promising results in experiments on various image classification and object detection tasks, often outperforming Adam and other popular optimization methods in terms of both speed and accuracy.

In [37], three more Adam variations were proposed, namely DGrad, Cos and Exp. DGrad is a modified version of DiffGrad which considers the absolute difference between the current and the moving average of the element-wise squares of the parameter gradients. In this way, it is more robust to fluctuations in the difference between the gradients.

Cos is a modification of DGrad that incorporates a learning rate that varies in a cyclical manner [43], leading to an improvement in classification accuracy and typically requiring fewer iterations.

Exp is a modification of DGrad that includes two simple element-wise operations: product and exponential. The purpose of Exp formulation is to mitigate the effect of large variations in the gradient but also to allow the function to converge for small values.

BAS-ADAM [22] is an improved version of the Beetle Antennae Search (BAS) algorithm, enhancing convergence behavior and avoiding local-minima by adaptively adjusting step-sizes using the ADAM update rule, resulting in faster convergence and efficient optimization of non-convex functions compared to Particle Swarm Optimization (PSO) and the original BAS algorithm.

A recent Adam variation is AngularGrad [39] that exploits the direction/angle of consecutive gradients to adjust the learning rate. Thanks to angle direction, the optimization becomes smoother while keeping a good trade-off between speed and performance.

In this paper, three more Adam variations are proposed. Their objective is to search the solution space differently and vary the prediction made by the models. Changing optimization creates models suitable for ensemble.

III. MATERIALS AND METHODS

In this section, we describe the different components of the proposed ensemble and we detail the new Adam variants proposed in this study.

A. CONVOLUTIONAL NEURAL NETWORKS

CNNs are a type of deep neural network that was specifically designed for image classification, computer vision, and other related applications, such as medical image analysis [9], face identification, and object recognition, among others. CNNs are designed to operate in a similar way to the human brain by perceiving visual information [24]. Recently, the combination of these models in ensembles has been proven to be beneficial in terms of performance (see for instance, [8], [36], [40]).

Convolution involves sliding a small filter (also known as a kernel) over the input data, which is typically a two-dimensional grid of pixels in the case of images. The element-wise multiplication is performed between the values of the given filter (learnable weights) and the corresponding values in the input data. The resulting products are summed up to produce a single value. This process is repeated by sliding the filter over the entire input, with a certain stride, to produce a new output matrix called a feature map.

The convolution operation allows the neural network to detect patterns or features present in the input data. These patterns can be as simple as edges or corners in the case of images. As the network learns through training, it adjusts the weights of the filters to capture increasingly complex and abstract features, such as textures, shapes, or object parts. Convolutional layers are typically stacked in CNN architectures, and each layer learns to recognize different levels of features. This hierarchical feature extraction enables CNNs to understand the content of images in a way that is analogous to how the human visual system processes information [7].

In our experiments, various models pre-trained on ImageNet are tested and combined. The last layers of each model are modified to fit the number of classes of the target problem without freezing the weights of the previous layers. The models evaluated include ResNet50 [20], which is about 8 times deeper than VGGNet [20] and uses residual layers and global average pooling layers instead of fully connected layers, and EfficientNetB0 [45], which is designed for mobile devices and uses a multi-objective network search that optimizes accuracy.

B. VISION TRANSFORMERS

In our experiments, we utilized advanced transformer models for image classification. Due to the significant amount of data required for training transformers from scratch, we adopted models that already trained on ImageNet that were then adapted to the task at hand through fine-tuning. These pre-trained models were obtained from the Timm library,[1] including DeiT-Base with a patch dimension of 16, ViT-Base with a patch size of 16, Swin-Base with a patch size of 4, and CoAtNet with a continuous log-coordinate relative position bias removed. It is worth noting that ViT implementation in Timm was pre-trained on Imagenet-21k, whereas the others were not.

For each model, we adjusted the last layer to align the output with the number of categories in each tested dataset. To prevent overfitting, we retained a validation split with a 0.25 split ratio from each training set and resized the image dimension to 224 × 224 to match the required input dimension.

[1]https://timm.fast.ai/ - Last access, June 28 2023

TABLE 1. Description of the datasets used in this study.

Short Name	#Classes	#Samples	Image Size	Protocol	Ref
WHOI	22	6600	greyscale	50:50	[44]
ZooScan	20	3771	greyscale	2CV	[44]
Kaggle	38	14374	greyscale	5CV	[44]
Zoolake	35	17943	RGB	85:15	[26]
BG	3	300	RGB	5CV	[13]
LAR	4	1320	RGB	3CV	[34]
Deng	10	563	RGB	35:65	[11]
VIR	15	1500	greyscale	10CV	[37]
TEM	14	1245	RGB	80:20	[33]

C. TRAINING AND TEST PHASES

During the training phase, each CNN is trained by adopting an optimization algorithm that is chosen at random among the ones available. The training process includes 20 epochs with a mini-batch size of 30 patterns and a learning rate of 0.001. Data augmentation is applied by flipping and rescaling the images, but only if the size of the training set is less than 5000 images. Otherwise, no data augmentation is performed.

Transformers are trained following the pipeline adopted for the DeiT model in [25], replicating the procedure for all four models. Specifically, we trained the models using the AdamW optimizer with cosine annealing [30], and set a low learning rate of 10^{-4} coupled with a weight decay of 0.03 to preserve the learned network. We set the batch size to 32.

During the test phase, the outputs of all the models are combined to compute the overall prediction. Thus, each model of the ensemble contributes to the final prediction. The output of the models is not automatically mapped into a probability distribution. Thus, we applied a softmax function to the score of the last layer before creating the ensemble prediction. As a loss function, we used the standard cross-entropy, as opposed to [25] that used the weighted cross-entropy. If the minimum validation F1 score did not decrease after five consecutive epochs, we decreased the learning rate to 10^{-5} and 10^{-6} to improve convergence. The best-performing model with the highest F1-score in validation during the training was saved. During training, data were randomly flipped or rotated. To create the ensemble, we repeated the training procedure ten times for each model and each dataset.

In Figure 1, we show how the training and test phase are organized. This can be considered as a testbed for ensembles that we use for our experiments combining different topologies of CNN and transformer facing seven publicly available datasets. However, this structure is general and can be used to combine any number of topologies.

D. DATASETS

We assess the proposed ensembles by adopting several image classification benchmarks. Table 1 reports some information for each dataset: a short name, the number of classes and samples, the testing protocol, and the original reference. For the testing protocols, we adopt the following abbreviations:

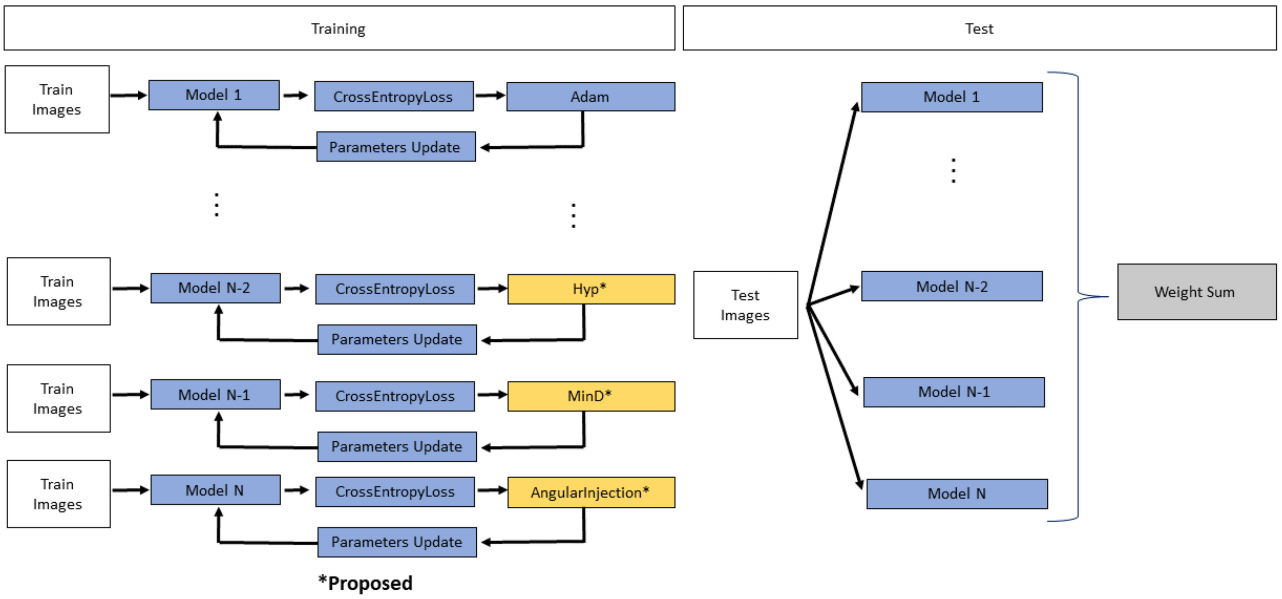


FIGURE 1. Example of an ensemble. Each CNN model is trained using an optimization algorithm that is randomly chosen a priori. During the test phase, the predictions are combined to compute the outcome of the ensemble.

- xCV indicates that an x-fold cross-validation has been adopted (e.g., 10CV means that a 10-fold cross-validation has been used);
- X:Z indicates the fractions of the dataset used for training and testing respectively.

The adopted datasets contain information from very different domains, in particular:

- WHOI consists of images, taken from Woods Hole Harbor water using Imaging FlowCytobot;
- ZooScan is composed of grayscale images that were acquired from the Bay of Villefranche-sur-mer using the Zooscan technology. The images were automatically cropped before classification to remove any artifacts caused by manual segmentation.
- Kaggle dataset is a subset of a larger dataset obtained using the ISIIS technology in the Straits of Florida and was used for the National Data Science Bowl 2015 competition.
- Zoolake is a dataset of microscopic plankton taxa images collected from the dual-magnification Scripps Plankton Camera in Lake Greifensee. The dataset was built using images acquired from wild plankton from 2018 to 2020.
- BG (Breast Grading Carcinoma) dataset contains images of size 1280 × 960 pixels and is divided into three classes representing grades 1-3 of invasive ductal carcinoma of the breast.
- LAR (Laryngeal dataset) contains patch images of size 100 × 100 pixels, divided equally into four classes: IPCL (tissue with intrapapillary capillary loops), Le (tissue with leukoplakia), Hbv (tissue with hypertrophic vessels), and He (healthy tissue).
- Deng contains images of pests commonly found on plants between Europe and Central Asia. It was created by collecting images from several online sources such as Insert Images, IPM images, Dave's Garden, and Mendeley Data.
- VIR comprises a total of 1500 Transmission Electron Microscopy images, each with a size of 41 × 41 pixels, of various virus types classified into fifteen distinct categories.
- TEM (Transmission Electron Microscopy) contains annotated transmission electron microscopy images (size of 1376 × 1032 or 2048 × 2048 pixels, depending on with which electron microscope they were captured) of 14 virus classes along with extracted image patches centered on virus particles.

E. NEW OPTIMIZATION METHODS

In this section, we introduce three new optimization methods. To better understand the proposed optimizers, it is worth recalling the Adam optimization algorithm and DGrad [37]. The update rule for Adam can be written as follows:

$$m_t = \beta_1 \times m_{t-1} + (1-\beta_1) \times g_t \tag{2}$$

$$v_t = \beta_2 \times v_{t-1} + (1-\beta_2) \times g_t^2 \tag{3}$$

$$m_{\hat{t}} = m_t / (1-\beta_1^t) \tag{4}$$

$$v_{\hat{t}} = v_t / (1-\beta_2^t) \tag{5}$$

$$\theta_t = \theta_{t-1} - \frac{\alpha \times m_{\hat{t}}}{(\sqrt{v_{\hat{t}}} + \epsilon)}$$

$$\beta_1 = 0.9;\ \beta_2 = 0.999;\ \alpha = 0.001 \tag{6}$$

where m_t is the first moment (mean) of the gradients up to time step t; v_t is the second moment (uncentered variance)

of the gradients up to time step t; g_t is the gradient at time step t; β_1 and β_2 are the decay rates for the first and second moments, respectively; α is the learning rate; ϵ is a small constant added for numerical stability; $m_{\hat{t}}$ and $v_{\hat{t}}$ are bias-corrected estimates of the first and second moments, respectively; θ_t is the current set of parameters at time step t.

DGrad [37] is a variation of Adam inspired by Diff-Grad [16]. It considers the absolute difference between the current and the moving average of the element-wise squares of the gradients (av_t), see Eq. (7)-(9). In this way, it is more robust to fluctuations in the difference between the gradients. For updating the parameters, Eq. (10) is applied using the definition, for the weighting factor, reported in Eq. (9):

$$\Delta ag_t = |g_t - av_t| \tag{7}$$

$$\Delta \widehat{ag}_t = \left(\frac{\Delta ag_t}{\max(\Delta ag_t)}\right) \tag{8}$$

$$\xi_t = \text{Sig}(4 \cdot \Delta \widehat{ag} t) \tag{9}$$

$$\theta_t = \theta_{t-1} - \xi_t \frac{\alpha \times m_{\hat{t}}}{(\sqrt{v_{\hat{t}}} + \epsilon)} \tag{10}$$

In the following paragraphs, the novel optimizers proposed in this paper are described in details.

1) HYPERBOLIC OPTIMIZER (HYP)

The first proposed optimizer is Hyperbolic (Hyp). It has a similar behavior of Exp [37], but does not mitigate the effects of large variations in gradient. Indeed, while the function described in Exp gets low values for great gradient differences, this approach simply calculates parameters as follows:

$$lr_t = \frac{-1}{(a\Delta ag_t + b)} + c \tag{11}$$

With $a = 10$, $b = 2/3$, and $c = 3/2$ we obtain the plot reported in Figure 2. The calculation of the final weighting factor of the learning rate is:

$$\xi_t = \frac{lr_t}{max(lr_t)} \tag{12}$$

then, the weighting factor of Eq.(12) is used in Eq.(10).

2) MIND OPTIMIZER

The second novel optimizer is called MinD and exploits two of the best optimizers in different ways. It calculates at every iteration ξ_{t1} using DGrad and ξ_{t2} using Exp. Then, the effective ξ_t is chosen by applying:

$$\xi_t = min(\xi_{t1}, \xi_{t2}) \tag{13}$$

then, Eq.(13) is applied in Eq.(10).

3) ANGULAR INJECTION OPTIMIZER (AI)

Angular Injection (AI) optimizer is based on AngularGrad [39] and injection [15]. It generates a score to control the step size based on the gradient angular information of previous

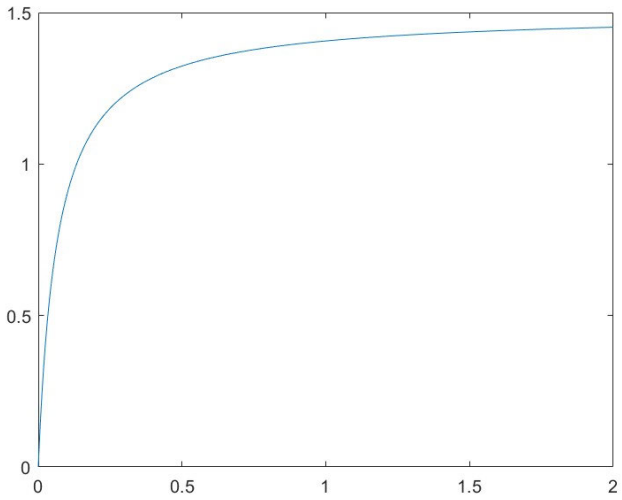


FIGURE 2. Plot of lr_t for $a = 10$, $b = 2/3$, and $c = 3/2$. On y-axis, the lr_t value. On the x-axis, the Δag_t value.

iterations. AI takes into account the information from the angle/direction of the gradient vector instead of just the magnitude of it. To exploit the change of gradients during the optimization steps, an angular coefficient was introduced (Eq. (14)):

$$A_t = tan^{-1}\frac{(g_t - g_{t-1})}{(1 + g_t g_{t-1})} \tag{14}$$

$$A_{min} = min(A_t, A_{t-1}) \tag{15}$$

$$lr_t = \frac{-1}{(aA_{min} + b)} + c \tag{16}$$

where $a = 10$, $b = 2/3$, and $c = 3/2$.

The weighting factor is:

$$\xi_t = \frac{lr_t}{max(lr_t)} \tag{17}$$

then, Eq.(17) is applied in Eq.(23).

In order to utilize the curvature information during optimization, the curvature information guided (weighted) second-order momentum is injected into first-order momentum:

$$g_s = g_t^2 \tag{18}$$

$$k = 2 \tag{19}$$

$$avg_t = \beta_1 \times avg_{t-1} + \frac{(1 - \beta_1)(g_t - delta \times g_s)}{k} \tag{20}$$

$$delta = \theta_{t-2} - \theta_{t-1} \tag{21}$$

$$avgsq_t = \beta_2 \times avgsq_{t-1} + (1 - \beta_2) \times g_s \tag{22}$$

$$step = \xi_t \times (\frac{avg_t}{\sqrt{avgsq_t} + \epsilon}) \tag{23}$$

$$\theta_t = \theta_{t-1} - step \cdot \frac{\alpha \times \sqrt{(1 - \beta_2^t)}}{(1 - \beta_1^t)} \tag{24}$$

The value *delta* in Eq.(21) represents the difference in the short-term of the parameters; the weighting factor ξ_t in

Eq. (23) is the one defined in Eq. (17); β_1 and β_2 are initialized as in Adam.

In this approach:

- in the first iteration, we use standard Adam;
- in the second we fix ξ_t=1 in Eq. (23), we use avg_t and $avgsq_t$ calculated using gradients obtained by standard Adam in the first iteration (i.e. using eq. (2) and (3));
- from the third iteration as in the method explained above.

F. WILCOXON SIGNED-RANK TEST

The Wilcoxon signed-rank test, introduced by Mann and Whitney in 1947 [32], is a statistical test designed for comparing paired data samples obtained from individual evaluations. Unlike parametric tests, this non-parametric test does not rely on assumptions about the underlying distribution of the data, such as a normal distribution. Instead, the Wilcoxon signed-rank test takes into account both the magnitudes and signs of the differences between paired observations.

This test serves as a non-parametric counterpart to the paired Student's t-test and is particularly useful when the population data does not follow a normal distribution. Its primary purpose is to assess whether two related paired samples are drawn from the same distribution. By analyzing the ranks assigned to the differences between the paired observations, the Wilcoxon signed-rank test provides a reliable means of evaluating the null hypothesis.

IV. EXPERIMENTAL ANALYSIS

In this section, we report the results of the experimental evaluation, considering different performance indicators for comparing the approaches. Moreover, validation of the superiority (using p-value) of one method over the others is provided by the Wilcoxon signed rank test. All the experiments were taken on a Windows Server 2019, with an Intel Core i9-10920X CPU, 3.5 GHz, and 256 GB RAM, we employed an Nvidia Titan RTX 24 GB, 1350 MHz. They are developed in Matlab 2022a/PyTorch.

Since CNNs require input images at a fixed size, we apply two different strategies for resizing plankton images. The first is sqr, in which the process of square resizing involves first padding the image to achieve a square dimension and subsequently resizing it to match the CNN input size. The second one is padding only (pad), in which the image is directly adjusted to match the CNN input size, without the intermediate square resizing step. It is important to note that the square resizing step becomes necessary only in specific scenarios where the original image dimensions exceed the dimensions of the CNN input size, prompting the image to undergo resizing. Padding is performed by adding white pixels to plankton images. Since we propose ensembles, half of the nets (in each ensemble) use sqr and the other half pad.

Our experiments involved a large number of training sessions applied to multiple topologies. As it is widely known, the behavior of the training loss should be carefully analyzed to check that the training phase was properly run and to measure the model convergence towards the desired task. In our context, a key element that should be highlighted is the influence of the optimizer on the behavior of the loss function during training. Figure 3 reports the loss while training a ResNet50 network on the Deng dataset for 20 epochs: in (a) the Adam optimizer was used, while in (b) and (c) the proposed Hyp and AI methods were employed, respectively. As it can be seen, the novel optimizers lead to a more homogeneous and faster convergence with respect to the Adam optimizer.

In all the experiments, each CNN and transformer network was trained several times using the standard Cross-Entropy loss. Each training leads to a network instance. Ensembles are created composing several instances and/or topologies. In the following, we denote with A+B the composition of networks A and B by sum rule. In the case of CNNs, each network is trained seven times – so, A+B means that both networks A and B were trained seven times on the same dataset, and the resulting 14 instances are combined by sum rule.

In the following, results labeled with SGD refer to the output of the fusion of 14 stand-alone CNNs trained using stocastic gradient descent and combined by the average rule. This ensembles is shaped in this way to ensure that it is comparable against the ensembles obtained by Hyp+MinD (ensembles that have size fourteen).

A. ENSEMBLES OF CONVOLUTIONAL NEURAL NETWORKS

We first run a set of experiments to get the performance of the ensemble of CNNs when using different Adam variants during the training phase. This was also useful for a comparison with the original methods that are reported as baselines. The results are reported in Table 2. In particular, we compare several Adam variants with the original Adam and SGD; for stand-alone Adam variant approaches, there are two values in each cell of the table:

- Average accuracy of seven stand-alone CNNs trained with the given optimization method;
- Fusion by average rule of seven stand-alone CNNs trained with the given optimization method.

From the results in Table 2 it can be noticed that the performance of the ensembles with the fusion by average rule is always better than the average accuracy of the stand-alone networks, providing another piece of evidence that the adoption of the ensemble is beneficial for the performance of the system. In Table 3, we report the performance on the plankton datasets of the most interesting Adam variants. The Adam variants, reported also in Table 3, outperform those not reported by a p-value of 0.001. However, all the Adam variants shown in Table 3 have similar performance. Moreover, in Tables 2 and 3 it can be noticed that Hyp+MinD outperforms both ensembles based on Adam and SGD with a p-value of 0.0001. Hyp+MinD+AI outperforms Hyp+MinD with a p-value of 0.005.

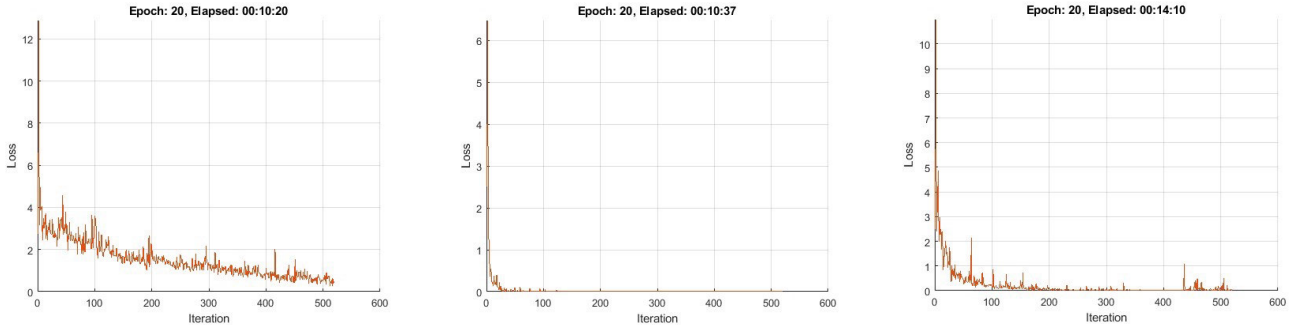


FIGURE 3. Loss function adopting different methods on the Deng dataset for 20 epochs using ResNet50. (a) Adam, (b) Hyp, and (c) AI.

B. ENSEMBLES OF TRANSFORMERS

The second set of experiments focuses on transformers – this was useful to get the performance of the ensembles and to compare them with the stand-alone original methods. We also tested transformer models trained with Adam variants, but we noticed no particular gains, therefore, for the sake of space, we have not reported them.

In Tables 4 and 5 we report the results obtained by the transformers. The following methods are reported in the table, where x represents the number of combined models:

- D(x), sum rule among x Deit – that is, x instances of the Deit transformer trained on the same dataset and combined by sum rule;
- S(x), sum rule among x Swin;
- V(x), sum rule among x Vit;
- C(x), sum rule among x Coat;
- CNNs is the ensemble of Hyp+MinD+AI coupled with both ResNet50 and EffNetB0;
- (D+V+S+C)(x), sum rule among x Deit, x Swin, x Vit and x Coat;
- CNNs+(D+V+S+C)(x), sum rule among CNNs, x Deit, x Swin, x Vit, and x Coat, before the fusion, the scores of each topology are normalized by the number of networks in the given ensemble, so the weight of CNNs is equal to that of C, S, V, and T.

Interestingly, there is no winner among stand-alone transformers and among the ensemble of transformers. However, it is interesting to notice once again that each set of 10 transformers outperforms its stand-alone transformer with a p-value of 0.002 (e.g., D(10) outperforms D(1) with a p-value of 0.002). On the other side, combining different transformer topologies does not seem useful as in the CNN case, for instance: (D+V+S+C)(2) behaves similarly to V(10) (sets of similar size); (D+V+S+C)(10) outperforms (D+V+S+C)(2) with a p-value of 0.002, however, this comparison is not fair given the different sizes of the two ensembles. Moreover, (D+V+S+C)(10) outperforms CNNs with a p-value of 0.002.

C. ENSEMBLES OF CNNS AND TRANSFORMERS

In the last rows of Tables 4 and 5, we report the results obtained by ensembles of CNNs and transformers. The small gain from ensembles of different topologies is probably due to the saturation of the models' performance on the datasets considered. In Table 6, we compare the best-performing approaches, considering the error under the ROC curve as a performance indicator. The fusion between CNNs and transformers outperforms both CNNs and transformers with a p-value of 0.01.

We tried to increase the size of the ensemble named "CNNs" by also using DenseNet201 and MobileNetV2 topologies. The results are reported in Tables 7 and 8. For the sake of computational time, we have run this test only on ZooLake and Deng datasets. It is interesting to notice that while the use of these CNN topologies improves the performance of the CNNs ensemble, the performance of the fusion of CNNs with the transformer ensemble remains similar. This suggests a possible plateau of performance reached with these approaches.

D. COMPARISON WITH SOTA AND ELAPSED TIME

We compared our approach against state-of-the-art (SOTA) approaches reported in the literature. Our proposed ensemble overcame the SOTA in many tested datasets. As the datasets we considered in our tests were used in hundreds of articles, we reported only a few articles in which SOTAs were obtained in those datasets to avoid creating huge tables. In addition, we only reported articles adopting the same test protocol we used, while we found other articles reporting better results but using different protocols, leading to an unfair comparison. Table 9 reports these results. For [42] we report the results obtained after 100 epochs to be coherent with our test protocol.

It is clear that our approach is not suitable for problems with strong computational constraints. We trained a large set of networks and achieved very good performance, however, we did not set any hyperparameters to optimize performance on specific datasets, in order to avoid overfitting and preserve

TABLE 2. Accuracy (in %) of ResNet50 and EffNetB0.

Model	Method	BG	LAR	Deng	VIR	TEM
ResNet50	Adam	86.57 89.67	92.15 96.29	74.02 83.37	76.85 86.40	85.47 92.00
	DiffGrad	89.00 91.67	93.01 95.91	81.15 90.55	80.08 89.47	84.11 91.84
	DGrad	89.29 92.67	91.07 94.85	81.36 90.61	78.13 90.00	84.98 91.68
	Cos	88.38 92.67	92.19 95.38	79.28 89.28	81.40 88.00	89.05 94.79
	Exp	92.00 **94.67**	94.37 96.67	92.12 94.48	82.07 88.53	89.37 95.42
	Hyp	91.95 94.00	94.22 96.67	93.18 **94.59**	85.19 90.27	91.80 95.63
	MinD	92.00 94.33	93.58 **97.20**	91.30 94.20	85.37 90.60	90.35 94.16
	AI	92.81 **94.67**	95.53 96.89	93.38 94.48	83.58 89.33	93.12 **95.84**
	Hyp+MinD	94.33	96.44	94.42	**91.73**	95.47
	Hyp+MinD+AI	**94.67**	97.12	**94.59**	91.33	95.53
	SGD	94.33	95.76	94.20	90.04	95.05
EffNetB0	Adam	90.62 93.33	94.17 **96.59**	86.58 93.20	84.35 91.13	90.01 94.47
	DiffGrad	92.43 94.33	94.59 96.29	87.55 93.20	84.53 90.80	90.26 94.79
	DGrad	92.62 94.67	94.34 96.44	86.49 92.93	84.23 **91.53**	91.04 95.00
	Cos	92.33 94.67	94.66 96.29	87.47 93.31	82.80 89.00	93.89 94.84
	Exp	94.48 **95.33**	94.72 96.36	93.36 94.20	85.08 89.27	93.58 95.16
	Hyp	94.10 94.33	93.80 95.91	93.27 93.59	87.39 91.27	93.77 95.79
	MinD	94.19 **95.33**	94.02 95.98	93.39 **94.25**	87.20 91.13	94.08 95.89
	AI	94.10 95.00	94.02 95.23	92.47 93.26	85.08 89.87	94.63 96.11
	Hyp+MinD	94.67	95.83	93.76	91.47	96.11
	Hyp+MinD+AI	95.00	96.14	93.70	91.40	**96.37**
	SGD	93.00	93.11	92.71	88.53	94.95

TABLE 3. Accuracy (in %) obtained by ResNet50 and EfficientNetB0 on plankton datasets.

Method	ResNet50				EffNetB0			
	WHOI	Kaggle	ZooScan	ZooLake	WHOI	Kaggle	ZooScan	ZooLake
Exp	94.97	94.00	88.15	96.9	95.66	93.93	88.47	96.45
Hyp	95.27	93.90	**88.71**	**96.93**	95.48	93.72	88.02	96.27
MinD	95.27	93.92	88.10	96.49	95.76	**94.04**	**88.87**	96.56
AI	**95.60**	94.12	88.07	96.64	95.33	93.56	87.67	**96.60**
Hyp+MinD	95.42	94.00	88.49	96.71	**95.79**	93.99	88.60	96.42
Hyp+MinD+AI	95.33	**94.20**	88.49	**96.93**	**95.79**	94.01	**88.87**	96.56
SGD	95.36	92.99	87.04	96.45	94.66	91.45	85.15	94.46

the generality of the proposed approach. Nevertheless, with current GPUs, even ensembles of more than 10 networks can classify dozens of images per second. In Table 10 we report the inference time, which is obviously higher when vision transformers are used; however, considering that these are able to analyze hundreds of images in a second, this is a reasonable time for many applications.

V. DISCUSSION ON ADAM VARIANTS

In this work, we have developed several optimization methods that address the problem of finding a good minimum in different ways, so it is useful to combine them in an ensemble. The proposed approaches work considering the absolute difference between the current and the moving average of the squares of the parameter gradients. In this way, it is more robust to fluctuations of the difference between the gradients of different iterations with respect to Adam and DiffGrad.

As detailed in the literature (e.g. [15]), an ideal parameter optimization method should follow the rules depicted in Figure 4, where Θ is the parameters tensor, *delta* is the difference of the parameters between two training iterations (see Eq. 21), g are the gradients. In the flat region (S1), an ideal optimizer should perform a large step in order to

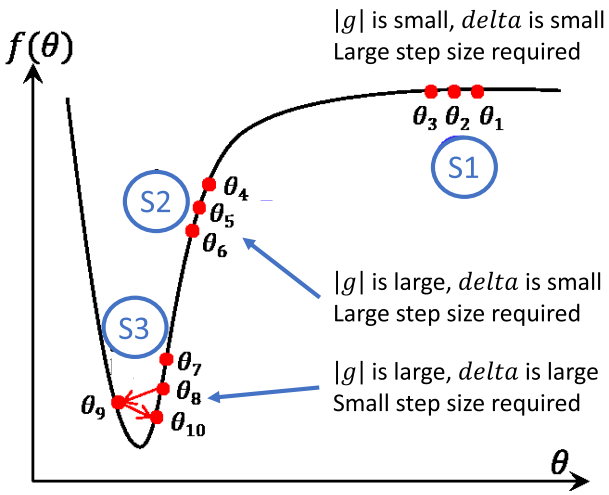


FIGURE 4. A common situation in optimization that illustrates the significance of adaptive parameter updates in the optimization process [49].

escape from the flat area. In the so-called "large gradient – small curvature" area (S2), for faster convergence, it is important to perform large step size. In the "steep and narrow valley" (S3), a minimum is found. In this area both gradients

TABLE 4. Accuracy (in %) and F1 obtained using transformers.

Models	BG		LAR		ZooLake		Deng		VIR		TEM	
	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
D(1)	93.3	0.931	**97.6**	**0.976**	96.6	0.863	93.1	0.941	90.2	0.895	95.8	0.948
S(1)	94.0	0.935	96.9	0.969	96.4	0.871	92.1	0.931	91.4	0.912	95.1	0.953
V(1)	**95.3**	**0.950**	**97.6**	**0.976**	96.4	0.861	**94.5**	**0.952**	90.8	0.906	95.5	0.939
C(1)	92.6	0.924	94.8	0.947	**96.7**	**0.876**	92.8	0.937	90.1	0.900	95.8	0.954
D(10)	**96.7**	**0.964**	97.6	0.976	96.9	0.871	94.7	0.954	91.5	0.912	96.5	0.956
S(10)	96.0	0.957	**97.9**	**0.979**	96.9	0.878	95.7	0.963	92.0	0.918	95.8	0.957
V(10)	95.3	0.950	97.6	0.976	**97.1**	0.880	**96.9**	**0.973**	92.9	0.926	96.4	0.957
C(10)	96.3	0.960	96.9	0.969	**97.1**	**0.884**	94.0	0.950	91.5	0.913	95.7	0.952
(D+V)(5)	95.7	0.954	98.0	0.980	97.2	0.884	94.6	0.954	92.3	0.921	**96.7**	0.962
(D+V)(10)	**96.0**	**0.957**	98.2	0.982	97.1	0.884	95.5	0.962	92.2	0.919	**96.7**	0.962
(D+V+S)(3)	95.0	0.947	**98.3**	**0.983**	**97.3**	0.888	95.9	0.965	92.5	0.923	96.4	0.960
(D+V+S+C)(2)	95.7	0.954	**98.3**	**0.983**	97.2	0.887	95.6	0.936	92.6	0.924	96.6	0.964
(D+V+S+C)(10)	**96.0**	**0.957**	97.9	0.979	**97.3**	**0.891**	**96.2**	**0.968**	92.7	0.925	96.2	0.962
CNNs	94.7	0.943	96.4	0.964	96.9	0.861	94.6	0.954	91.9	0.919	96.3	**0.965**
CNNs + (D+V+S+C)(2)	95.0	0.946	**98.3**	**0.983**	97.2	0.888	95.9	0.965	93.0	**0.929**	**96.7**	0.964
CNNs + (D+V+S+C)(10)	**95.3**	**0.950**	98.0	0.980	**97.3**	**0.889**	**96.4**	**0.969**	**93.1**	**0.929**	**96.7**	**0.965**

TABLE 5. Accuracy (in %) and F1 obtained using transformers.

Models	WHOI		Kaggle		ZooScan	
	Acc.	F1	Acc.	F1	Acc.	F1
D(1)	94.5	0.945	93.6	0.920	87.3	0.889
S(1)	95.6	0.956	**94.0**	**0.926**	87.4	**0.897**
V(1)	93.6	0.936	93.6	0.920	**87.8**	0.892
C(1)	**95.7**	**0.957**	93.8	0.922	87.3	0.892
D(10)	95.7	0.957	94.4	0.931	88.9	0.903
S(10)	95.9	0.959	**94.7**	**0.935**	**89.5**	**0.915**
V(10)	95.8	0.958	94.4	0.931	88.9	0.903
C(10)	**96.3**	**0.963**	**94.7**	**0.936**	89.1	0.910
(D+V)(5)	95.7	0.957	94.5	0.932	89.4	0.909
(D+V)(10)	95.8	0.958	94.6	0.933	89.5	0.909
(D+V+S)(3)	95.9	0.959	94.6	0.934	90.0	0.916
(D+V+S+C)(2)	**96.3**	**0.963**	94.7	0.936	89.8	0.915
(D+V+S+C)(10)	**96.3**	**0.963**	**94.8**	**0.938**	**90.1**	**0.918**
CNNs	95.9	0.959	94.5	0.932	89.5	0.911
CNNs + (D+V+S+C)(2)	**96.4**	**0.964**	94.8	0.937	90.0	0.917
CNNs + (D+V+S+C)(10)	**96.4**	**0.964**	**94.9**	**0.939**	**90.3**	**0.918**

TABLE 6. Error under the ROC curve (in %).

EUC	WHOI	Kaggle	ZooScan	BG	LAR	ZooLake	Deng	VIR	TEM
(D+V+S+C)(10)	0.226	0.417	0.917	**0.192**	0.033	0.320	0.191	0.965	0.241
CNNs	0.285	0.400	0.928	2.102	0.095	0.355	0.515	0.666	0.260
CNNs+(D+V+S+C)(10)	**0.210**	**0.367**	**0.796**	0.227	**0.030**	**0.272**	**0.180**	**0.532**	**0.228**

TABLE 7. Accuracy (in %) using more CNN topologies.

Accuracy	ZooLake	Deng
(D+V+S+C)(10)	97.3	96.2
CNNs	97.0	95.2
CNNs+(D+V+S+C)(10)	**97.4**	**96.4**

TABLE 8. Error under the ROC curve (in %), using more CNN topologies.

EUC	ZooLake	Deng
(D+V+S+C)(10)	0.320	0.191
CNNs	0.350	0.469
CNNs+(D+V+S+C)(10)	**0.272**	**0.178**

and *delta* are large, a small step size is required for finding the minimum and reducing the oscillations.

Taking this into consideration, the three proposed optimizers have the following properties:

- the Hyp optimizer is based on DGrad/DiffGrad, so the idea is that gradient is changing more gradually near the minimum, see S3 area when approaching the minimum. DGrad/DiffGrad are based on a sigmoid function that squashes every value between 0.5 and 1. Instead, Hyp is based on a function that squashes every value between 0 and 1 (see Eq. (12)), it takes larger steps in the S2 area and lower steps near the minimum. The main drawback of this approach is that when gradient variations are close to zero the parameters are updated only for a small value, so this approach could converge more slowly than

TABLE 9. Comparison vs. state-of-the-art methods (Accuracy in %).

Dataset	WHOI	ZooScan	Kaggle	ZooLake	BG	LAR	Deng	VIR	TEM
Here	**96.4**	**90.3**	**94.9**	97.3	95.3	**98.0**	**96.4**	**93.1**	**96.7**
[26]	96.1	89.8	94.7	**97.7**					
[31]	95.8	88.8	94.2						
[5]					94.3	96.8	95.0		
[11]							85.5		
[4]							95.2		
[37]					95.0	97.0	95.6	92.5	
[13]					**96.3**				
[42]									96.4
[3]									96.1
[33]									93.1

TABLE 10. Inference time (seconds) for a batch of 100 images using a NVidia Titan RTX.

Model	Images	Batch Inference
Resnet 50	100	0.22 s
Vit	100	0.43 s
DeiT	100	0.43 s
Swin	100	0.40 s
CoAtNet	100	0.46 s

DGrad/DiffGrad in the S1 area. However, unlike DGrad, we do not consider the difference between the gradients of two iterations but rather between the current iteration and the moving average of the element-wise squares of the gradients. In this way we have values less close to zero even when the gradient changes very slowly. Moreover, the step size is sufficiently large due to the small value of Eq. (5) in the denominator of Eq. (10).

- MinD is a minimum rule, it reduces and smoothes the steps, minimizing the risk of skipping a minimum. Again, the drawback of this approach could be a slower convergence in the S1 area.
- AI optimizer controls the step size based on the information from the angle/direction of the gradient vector instead of just its magnitude. As shown in Eq. (15), we take the minimum between two angular coefficients, in this way the AI optimizer is more robust to shape curvature changes. Moreover, we add a further step: the curvature information is used as a weight to inject the second-order momentum in the update rule, see Eq. (20) and Eq. (23). In the S1 area, the step size is sufficiently large due to the small value of the gradients and therefore of the denominator of Eq. (23). It should be noted that *delta* is applied only in Eq. (20), i.e. the numerator of Eq. (23), and not in its denominator; $delta < 0$ when $g_t > 0$ and $delta > 0$ when $g_t < 0$. In the S2 area, $|g_t|$ is large and the value of Eq. (20) is also sufficiently large even if *delta* is small, *delta* increases the value of Eq. (20), therefore we have a larger step with respect to Adam or Hyp. In the S3 area, $|g_t|$ and *delta* are both large, so (given their relationship highlited above) the injection takes a larger step with respect to Hyp. In conclusion, AI performs better than Hyp/DGrad in the S1 and S2 areas and worse in the S3 area. Due to this different behavior, we believe that these methods are suitable to be combined into an ensemble.

VI. CONCLUSION

In this paper, we combined CNNs based on different topologies and various Adam optimization methods for image classification. New Adam-based algorithms for deep network optimization are proposed for training a set of CNNs. In addition, we compare sets of CNNs with sets of transformers. The ensembles were compared and evaluated using different evaluation metrics. The best-performing ensemble, consisting of the CNN ensemble and the transformer ensemble, was shown to have the best performance, compared to the literature, on several benchmarks. In the future, we plan to evaluate ad-hoc Adam's variants on transformers. Future directions include refining distillation techniques, optimizing ensemble design, exploring data augmentation strategies, and diversifying data sources to mitigate ensemble limitations.

DECLARATION OF INTERESTS

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

ACKNOWLEDGMENT

The authors used a donated TitanX GPU to train deep networks used in this work. They would like to thank Giuliano Boscarin, Matteo Bassani, and Lorenzo Giannini who worked on this project as partial fulfillment of their bachelor's degree.

REFERENCES

[1] A. M. Al-Hejri, R. M. Al-Tam, M. Fazea, A. H. Sable, S. Lee, and M. A. Al-Antari, "ETECADx: Ensemble self-attention transformer encoder for breast cancer diagnosis using full-field digital X-ray breast images," *Diagnostics*, vol. 13, no. 1, p. 89, Dec. 2022.

[2] M. R. Al-Sinan, A. F. Haneef, and H. Luqman, "Ensemble learning using transformers and convolutional networks for masked face recognition," in *Proc. 16th Int. Conf. Signal-Image Technol. Internet-Based Syst. (SITIS)*, Oct. 2022, pp. 421–426.

[3] M. M. Ali, R. C. Joshi, M. K. Dutta, R. Burget, and A. Mezina, "Deep learning-based classification of viruses using transmission electron microscopy images," in *Proc. 45th Int. Conf. Telecommun. Signal Process. (TSP)*, 2022, pp. 174–178.

[4] E. Ayan, H. Erbay, and F. Varçın, "Crop pest classification with a genetic algorithm-based weighted ensemble of deep convolutional neural networks," *Comput. Electron. Agricult.*, vol. 179, Dec. 2020, Art. no. 105809.

[5] R. Bravin, L. Nanni, A. Loreggia, S. Brahnam, and M. Paci, "Varied image data augmentation methods for building ensemble," *IEEE Access*, vol. 11, pp. 8810–8823, 2023.

[6] T. B. Brown T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, and A. Askell, "Language models are few-shot learners," in *Proc. Adv. Neur. Inf. Process. Sys.*, vol. 33, 2020, pp. 1877–1901.

[7] S. A. Cadena, G. H. Denfield, E. Y. Walker, L. A. Gatys, A. S. Tolias, M. Bethge, and A. S. Ecker, "Deep convolutional models improve predictions of macaque v1 responses to natural images," *PLOS Comput. Biol.*, vol. 15, no. 4, Apr. 2019, Art. no. e1006897.

[8] C. Cornelio, M. Donini, A. Loreggia, M. S. Pini, and F. Rossi, "Voting with random classifiers (VORACE): Theoretical and experimental analysis," *Auto. Agents Multi-Agent Syst.*, vol. 35, no. 2, p. 22, Oct. 2021.

[9] P. Coupé, B. Mansencal, M. Clément, R. Giraud, B. D. de Senneville, V.-T. Ta, V. Lepetit, and J. V. Manjon, "AssemblyNet: A large ensemble of CNNs for 3D whole brain MRI segmentation," *NeuroImage*, vol. 219, Oct. 2020, Art. no. 117026.

[10] Z. Dai, H. Liu, Q. V. Le, and M. Tan, "CoAtNet: Marrying convolution and attention for all data sizes," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 3965–3977.

[11] L. Deng, Y. Wang, Z. Han, and R. Yu, "Research on insect pest image detection and recognition based on bio-inspired methods," *Biosystems Eng.*, vol. 169, pp. 139–148, May 2018.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[13] K. Dimitropoulos, P. Barmpoutis, C. Zioga, A. Kamas, K. Patsiaoura, and N. Grammalidis, "Grading of invasive breast carcinoma through Grassmannian VLAD encoding," *PLoS ONE*, vol. 12, no. 9, Sep. 2017, Art. no. e0185110.

[14] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[15] S. R. Dubey, S. H. S. Basha, S. K. Singh, and B. B. Chaudhuri, "Adainject: Injection based adaptive gradient descent optimizers for convolutional neural networks," 2022, *arXiv:2109.12504*.

[16] S. R. Dubey, S. Chakraborty, S. K. Roy, S. Mukherjee, S. K. Singh, and B. B. Chaudhuri, "DiffGrad: An optimization method for convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 4500–4511, Nov. 2020.

[17] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, Feb. 2011.

[18] S. d'Ascoli, M. Refinetti, G. Biroli, and F. Krzakala, "Double trouble in double descent: Bias and variance (S) in the lazy regime," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 2280–2290.

[19] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, Z. Yang, Y. Zhang, and D. Tao, "A survey on vision transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 87–110, Jan. 2023.

[20] 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.

[21] S. Hossain, A. Chakrabarty, T. R. Gadekallu, M. Alazab, and M. J. Piran, "Vision transformers, ensemble model, and transfer learning leveraging explainable AI for brain tumor detection and classification," *IEEE J. Biomed. Health Informat.*, early access, Apr. 12, 2023, doi: 10.1109/JBHI.2023.3266614.

[22] A. H. Khan, X. Cao, S. Li, V. N. Katsikis, and L. Liao, "BAS-ADAM: An Adam based approach to improve the performance of beetle antennae search optimizer," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 2, pp. 461–471, Mar. 2020.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 25, 2012, pp. 1–9.

[25] S. P. Kyathanahally, T. Hardeman, M. Reyes, E. Merz, T. Bulas, P. Brun, F. Pomati, and M. Baity-Jesi, "Ensembles of data-efficient vision transformers as a new paradigm for automated classification in ecology," *Sci. Rep.*, vol. 12, no. 1, p. 6243, Nov. 2022.

[26] S. P. Kyathanahally, T. Hardeman, E. Merz, T. Bulas, M. Reyes, P. Isles, F. Pomati, and M. Baity-Jesi, "Deep learning classification of lake zooplankton," *Frontiers Microbiology*, vol. 12, p. 3226, Nov. 2021.

[27] J. Li, J. Chen, Y. Tang, C. Wang, B. A. Landman, and S. K. Zhou, "Transforming medical imaging with transformers? A comparative review of key properties, current progresses, and future perspectives," *Med. Image Anal.*, vol. 85, Apr. 2023, Art. no. 102762.

[28] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[29] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9992–10002.

[30] I. Loshchilov and F. Hutter, "Fixing weight decay regularization in Adam," in *Proc. ICLR*, 2017, pp. 1–14.

[31] A. Lumini, L. Nanni, and G. Maguolo, "Deep learning for plankton and coral classification," *Appl. Comput. Informat.*, vol. 19, nos. 3–4, pp. 265–283, Jun. 2023.

[32] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *Ann. Math. Statist.*, vol. 18, no. 1, pp. 50–60, Mar. 1947.

[33] D. J. Matuszewski and I.-M. Sintorn, "TEM virus images: Benchmark dataset and deep learning classification," *Comput. Methods Programs Biomed.*, vol. 209, Sep. 2021, Art. no. 106318.

[34] S. Moccia, E. De Momi, M. Guarnaschelli, M. Savazzi, and A. Laborai, "Confident texture-based laryngeal tissue classification for early stage diagnosis support," *J. Med. Imag.*, vol. 4, no. 3, Sep. 2017, Art. no. 034502.

[35] L. Nanni, C. Fantozzi, A. Loreggia, and A. Lumini, "Ensembles of convolutional neural networks and transformers for polyp segmentation," *Sensors*, vol. 23, no. 10, p. 4688, May 2023.

[36] L. Nanni, A. Lumini, A. Loreggia, S. Brahnam, and D. Cuza, "Deep ensembles and data augmentation for semantic segmentation," in *Diagnostic Biomedical Signal and Image Processing Applications With Deep Learning Methods*. Amsterdam, The Netherlands: Elsevier, 2023, pp. 215–234.

[37] L. Nanni, A. Manfè, G. Maguolo, A. Lumini, and S. Brahnam, "High performing ensemble of convolutional neural networks for insect pest image detection," *Ecological Informat.*, vol. 67, Mar. 2022, Art. no. 101515.

[38] M. M. Naseer, K. Ranasinghe, S. H. Khan, M. Hayat, F. S. Khan, and M.-H. Yang, "Intriguing properties of vision transformers," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 23296–23308.

[39] S. K. Roy, M. E. Paoletti, J. M. Haut, S. R. Dubey, P. Kar, A. Plaza, and B. B. Chaudhuri, "AngularGrad: A new optimization technique for angular convergence of convolutional neural networks," 2021, *arXiv:2105.10190*.

[40] B. Savelli, A. Bria, M. Molinara, C. Marrocco, and F. Tortorella, "A multi-context CNN ensemble for small lesion detection," *Artif. Intell. Med.*, vol. 103, Mar. 2020, Art. no. 101749.

[41] S. Shen, X. Wang, F. Mao, L. Sun, and M. Gu, "Movements classification through sEMG with convolutional vision transformer and stacking ensemble learning," *IEEE Sensors J.*, vol. 22, no. 13, pp. 13318–13325, Jul. 2022.

[42] N. Sikder, M. A.-M. Khan, A. K. Bairagi, M. Masud, and A.-A. Nahid, "Heterogeneous virus classification using a functional deep learning model based on transmission electron microscopy images," Tech. Rep., 2023, doi: 10.13140/RG.2.2.27798.11849/1.

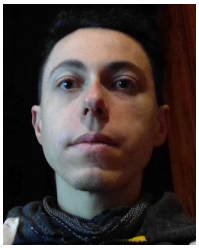
[43] L. N. Smith, "Cyclical learning rates for training neural networks," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 464–472.

[44] H. M. Sosik and R. J. Olson, "Automated taxonomic classification of phytoplankton sampled with imaging-in-flow cytometry," *Limnology Oceanogr., Methods*, vol. 5, no. 6, pp. 204–216, Jun. 2007.

[45] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[46] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 10347–10357.

[47] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 30, 2017, pp. 1–11.
[48] B. Yang, G. Bender, Q. V. Le, and J. Ngiam, "CondConv: Conditionally parameterized convolutions for efficient inference," in *Proc. Adv. Neural Inf. Process. Syst.*, 32, 2019, pp. 1–12.
[49] J. Zhuang, T. Tang, Y. Ding, S. C. Tatikonda, N. Dvornek, X. Papademetris, and J. Duncan, "Adabelief optimizer: Adapting stepsizes by the belief in observed gradients," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 18795–18806.



LORIS NANNI is currently an Associate Professor with the Department of Information Engineering, University of Padova. He carries out research with DEI, University of Padova, in the fields of biometric systems, pattern recognition, machine learning, image databases, and bioinformatics. He is the coauthor of more than 300 research articles. He has an H-index of 57 and more than 11,000 citations (Google Scholar). He has extensively served as a Referee for international journals (IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, *Pattern Recognition*, *Bioinformatics*, the *BMC Bioinformatics*, and the *Pattern Recognition Letters*) and projects.

ANDREA LOREGGIA received the master's degree (cum laude) from the University of Padova, in 2012, and the Ph.D. degree in computer science, in 2016. He is currently an Assistant Professor with the Department of Information Engineering, University of Brescia. His studies are dedicated to designing and providing tools for developing intelligent agents capable of representing and reasoning with preference and ethical-moral principles. His current research interest includes artificial intelligence spanning from knowledge representation to deep learning. He is a member of the UN/CEFACT Group of Experts, where he actively participates in the dissemination and sustainable development of technology.

LEONARDO BARCELLONA received the master's degree (laude) in computer engineering from the University of Padova, Italy, in 2021. He is currently pursuing the joint Ph.D. degree in artificial intelligence with the Intelligent and Autonomous Systems Laboratory, University of Padova, and Politecnico di Torino, Turin, Italy. His current research interests include semantic segmentation, few-shot learning, human pose estimation, robotics, and deep learning.

STEFANO GHIDONI received the M.Sc. degree in telecommunication engineering and the Ph.D. degree in information technologies from the University of Parma, in 2004 and 2008, respectively. During the Ph.D. degree, he worked in the field of human perception for autonomous driving. He is currently a Full Professor of computer vision with the Department of Information Engineering, University of Padova, Italy. His current research interests include computer vision and deep learning for human–robot collaboration, multimodal human perception, semantic segmentation, scene understanding, and medical imaging.

Open Access funding provided by 'Università degli Studi di Padova' within the CRUI CARE Agreement