Journal of the Brazilian Computer Society, 2026, 32:1, doi: 10.5753/jbcs.2026.5898

Meta-Learning based Few-Shot Classification of Retinal Diseases

Gabriel J. Perin [**Universidade de São Paulo** | ***gabrieljp@usp.br***]
Diogo J. C. Alves [**Universidade de São Paulo** | ***diogo.alves@usp.br***]
Lucas M. S. Sousa [**Universidade de São Paulo** | ***lucas.medeiros.sousa@usp.br***]
Erik M. de Elias [**Universidade de São Paulo** | ***edeelias@acm.org***]
Nina S. T. Hirata [**Universidade de São Paulo** | ***nina@ime.usp.br***]

Instituto de Matemática, Estatística e Ciência da Computação, Universidade de São Paulo, Rua do Matão, 1010, Butantã, São Paulo, SP, 05508-090, Brazil.

Received: 12 April 2025 • **Accepted:** 12 December 2025 • **Published:** 16 June 2026

Abstract To address sample scarcity, a common challenge in medical imaging, we investigate the Reptile meta-learning algorithm for few-shot disease classification in fundus images. In a setup with eight training classes and four testing classes, we investigate different architectures, training strategies, varying N-way K-shot configurations, and different data augmentation techniques. Our results show that Reptile outperforms standard transfer learning approaches in several settings and that, when combined with data augmentation, especially during evaluation, correct predictions tend to receive higher confidence scores. The quantitative results are further supported by an analysis of prediction confidence levels and activation-map visualizations.

Keywords: Few-shot learning, meta-learning, Reptile, eye disease, retina image, image classification

1 Introduction

Eye fundus imaging can reveal signs of a variety of ocular diseases. Among these diseases, diabetic retinopathy (DR) is a well-known complication of diabetes and, in its advanced stage, can lead to severe visual impairment or even blindness. In addition to diseases that can affect vision, such as DR and macular degeneration, other systemic diseases, such as hypertension, stroke, cardiovascular and neurological diseases are often linked to retina anomalies, especially related to blood vessels. In fact, the eye is considered a window that enables *in vivo* non-invasive observation of blood vessels and neural tissues [Kawasaki and Grauslund, 2019].

Periodic examinations can support early detection and disease progression monitoring, and are crucial for interventions aimed at preventing severe outcomes. However, effectively implementing such monitoring routines in healthcare systems requires addressing the large volume of retinograms that need to be examined.

Thus, there is increasing demand for artificial intelligence models that can support disease detection in fundus images. Combining the visual information available in fundus images with recent advances in machine learning, automated detection of eye diseases has become both feasible and clinically relevant. Studies based on deep learning (DL) techniques have demonstrated strong performance in detecting diseases such as diabetic retinopathy, macular degeneration, glaucoma, and hemorrhage [Tazoar *et al*., 2024]. These approaches can help address shortages of medical resources by providing cost-effective and rapid analyses, and may support ophthalmologists in fundus screening [Kawasaki and Grauslund, 2019; Li *et al*., 2021]. In addition, they can provide quantitative information that complements expert assessment.

Nonetheless, DL models usually require large datasets for training and evaluation. Obtaining medical data emerges as a significant challenge in this domain, as it involves several concerns, such as patient privacy, rare diseases, data annotation complexity, class imbalance, variability in image acquisition conditions, ethical considerations, and regulatory restrictions.

Few-shot Learning (FSL) plays a crucial role in addressing this challenge, by enabling models to learn from only a few examples [Pachetti and Colantonio, 2023]. In particular, in the context of FSL, it has been shown that, through the idea of "learning to learn", meta-learning paradigm enables the creation of models that can quickly adapt to unseen classes and domains [Song *et al*., 2023].

In this work, we investigate the Reptile meta-learning algorithm [Nichol *et al*., 2018] for the few-shot retinal disease classification problem. Compared to many meta-learning algorithms and frameworks [Hospedales *et al*., 2022], Reptile is a well-established algorithm with several advantageous properties, such as low memory cost, ease of implementation, and model-agnostic design, making it a strong choice in the FSL context.

We use the Brazilian Multilabel Ophthalmological Dataset (BRSET) [Nakayama *et al*., 2024b] in our study. BRSET is a collection of retinal fundus images, collected from Brazilian patients. The images are annotated according to several classification dimensions, including anatomical classification, quality control parameters, pathological classification, and diabetic retinopathy classification. We focus on the pathological classification, which includes the following classes: diabetic retinopathy, diabetic macular edema, scar (toxoplasmosis), nevus, age-related macular degeneration (AMD), vascular occlusion, hypertensive retinopathy, drusens, nondiabetic retinal hemorrhage, retinal detachment, myopic fundus, increased cup disc ratio, and other.

Motivated by limited-label scenarios, which are particularly

relevant for eye fundus imaging in Latin America, we pose the following research question: *Can Reptile be leveraged to build a model capable of classifying unseen eye diseases from only a few labeled examples?*

To answer this question, several experiments were conducted, comparing Reptile to traditional transfer learning approaches, exploring different data augmentation strategies, and qualitatively evaluating the patterns learned by the models. The main contributions of this research are:

- To the best of our knowledge, this is the first study to evaluate the application of Reptile, coupled with different model architectures, to the retinal disease classification problem.
- A comparative analysis of advanced data augmentation techniques and their integration into the Reptile algorithm.
- A novel qualitative analysis comparing the patterns learned by models with and without the application of Reptile and data augmentation techniques.

Building upon our previous work [Perin and Hirata, 2024], we introduce a domain-specific pretraining strategy applied before the Reptile meta-learning phase. This strategy leverages a distinct ophthalmological dataset to improve final model performance. In addition, we present a new quantitative analysis of how different data augmentation settings affect the distribution of prediction confidence, offering further insight into the effects of augmentation on model uncertainty. All experimental protocols have been revised and updated to incorporate the newly introduced methods and analyses.

The next sections are organized as follows. Section 2 presents some background material on FSL, including the Reptile algorithm and its evaluation protocol, and overview of some related work. Section 3 presents the methodology used in this study, including a description of the datasets used and an outline of the main experiments designed to answer our questions. Section 4 presents details of the experiments, achieved results, and discussions. Section 5 presents our final thoughts and foreseen future studies.

2 Background

In this section, we first present representative work in the literature closely related to our study. We then recall the few-shot learning problem, describe the Reptile meta-learning algorithm used in this work, and explain how meta-learning algorithms are evaluated in few-shot learning scenarios.

2.1 Related work

In the medical imaging domain, data scarcity is a common challenge for learning-based approaches. Several works have applied the FSL paradigm to the medical domain [Pachetti and Colantonio, 2023]. For instance, Nurgazin and Tu [2023] compare the efficacy of different meta-learning algorithms, data augmentation techniques, and model architectures for the problem of skin tissue classification.

FSL in retinal disease classification has been addressed in some works [Huang *et al*., 2022; Mahapatra *et al*., 2022; Roychowdhury, 2021; Cai *et al*., 2022]. However, we are aware of only one work [Rajpoot and Seeja, 2023] that has specifically applied the meta-learning paradigm to the problem, by combining Siamese neural networks with Triplet loss and a K-nearest Neighbor (KNN) classifier. The method employed there is metric-based, and thus inherently different from Reptile, which is an optimization-based method.

Regarding the use of Reptile, we highlight MetaMed [Singh *et al*., 2021], a method that applies non-trivial data augmentation techniques to the inner loop of the Reptile algorithm to address the skin tissue classification problem. As for data augmentation in the context of meta-learning, Ni *et al*. [2021] studied the effect of different data augmentation techniques and application schemes on various meta-learning algorithms. Their study found that final performance is highly dependent on the meta-learning stage at which augmentation is applied. However, the study does not include the Reptile algorithm.

In summary, meta-learning in retinal disease classification is still an almost unexplored topic. We found no work that employs Reptile for this problem. We also identify opportunities to explore data augmentation in Reptile for this problem by combining ideas for inner-loop data augmentation in Reptile, as done in [Singh *et al*., 2021], with general data augmentation schemes not restricted to the inner loop of meta-learning approaches, as reported in Ni *et al*. [2021]. It is worth emphasizing that neither of these works considered retinal images.

2.2 Few-shot Learning and Meta-learning

Few-shot learning (FSL) refers to the problem of training models to generalize from only a few labeled examples [Wang *et al*., 2020].

A wide range of FSL algorithms have been proposed [Song *et al*., 2023]. Some approaches are optimization-based, leveraging training tasks to learn initialization parameters that facilitate generalization to unseen tasks. Notable examples include MAML [Finn *et al*., 2017], FOMAML [Nichol *et al*., 2018], iMAML [Rajeswaran *et al*., 2019], and Reptile [Nichol *et al*., 2018]. Other methods focus on explicitly optimizing network architectures to learn effective feature embeddings, such as Siamese Networks [Koch *et al*., 2015], Prototypical Networks [Snell *et al*., 2017], and Matching Networks [Vinyals *et al*., 2016], among others.

Many of these algorithms fall within the meta-learning approach [Hospedales *et al*., 2022]. While algorithms in standard learning are trained for specific tasks using input-output exemplars of the task, in meta-learning the algorithms are exposed to a distribution of tasks. Rather than optimizing the parameters for specific tasks, the training approach is modeled in such a way that parameters are optimized across a group of tasks. Thus, it is expected that meta-learning will generate models that are better suited to learn new tasks in few-shot scenarios. Due to this characteristic, meta-learning is also referred to as "learning to learn".

To better describe the core idea of meta-learning, we detail it for a multi-class classification problem. In the classification setting, FSL is typically formulated in terms of N-way K-shot tasks, where N denotes a number of classes out of the total classes, and each task consists of K examples from each of the N classes. Given a set $\mathcal{D} = \{(x_j, y_j)\}_{j=1}^{n}$ of instances

of the classification problem, the meta-learning classification problem considers a partition of $\mathcal{D}$ into two subsets: a meta-training set $\mathcal{D}_{\text{train}}$ and a meta-testing set $\mathcal{D}_{\text{test}}$, ensuring there is no class overlap between the two sets. To build a N-way K-shot task $\mathcal{T}(\mathcal{D}_i)$, $i \in \{\text{train}, \text{test}\}$, N distinct classes in $\mathcal{D}_i$ are sampled and then K examples for each of the N classes are sampled from $\mathcal{D}_i$. The general idea is to use tasks built from $\mathcal{D}_{\text{train}}$ to train the model and those built from $\mathcal{D}_{\text{test}}$ to evaluate the model regarding its ability to generalize to new few-shot tasks.

2.3 Reptile algorithm

Reptile [Nichol *et al*., 2018] is a well-established optimization-based meta-learning algorithm with several advantageous properties, including low memory requirements, ease of implementation, and architecture-agnostic design, making it a compelling choice for few-shot learning.

Optimization-based meta-learning algorithms such as MAML and Reptile are designed to leverage the set of training tasks, i.e., those sampled from $\mathcal{D}_{\text{train}}$, to learn an initialization that enables efficient adaptation to unseen test tasks sampled from $\mathcal{D}_{\text{test}}$. In the Reptile algorithm, this is achieved through an iterative optimization process in which the model parameters $\theta \in \mathbb{R}^d$ are updated using N-way K-shot tasks from $\mathcal{D}_{\text{train}}$. The optimization of the parameters is performed via a two-loop procedure: an *inner-loop* that aims adaptation to individual tasks and an *outer-loop* meta-update that refines the optimization across multiple tasks (see Algorithm 1).

Algorithm 1 Reptile

Require: $N, K, m \in \mathbb{N}, \theta \in \mathbb{R}^d, \mu \in \mathbb{R}$
  for iteration $= 1, 2, \ldots$ **do**
    Sample m N-Way K-Shot tasks $\{\mathcal{T}^i(\mathcal{D}_{\text{train}})\}_{i=1}^m$
    for $i = 1, \ldots, m$ **do**
      Train f_θ (using SGD or ADAM) on $\mathcal{T}^i(\mathcal{D}_{\text{train}})$ to obtain θ_i
    end for
    Update $\theta \leftarrow \theta + \mu \frac{1}{m} \sum_{i=1}^{m} (\theta_i - \theta)$
  end for

First, in the inner-loop, the algorithm samples m training tasks $\{\mathcal{T}^i(\mathcal{D}_{\text{train}})\}_{i=1}^m$ and independently trains f_θ on each of them, obtaining optimized parameters $\theta_1, \ldots, \theta_m$. Next, in the outer-loop, the algorithm updates θ by averaging the directions previously obtained in the inner loop trainings, as follows:

$$\theta \leftarrow \theta + \mu \frac{1}{m} \sum_{i=1}^{m} (\theta_i - \theta) \tag{1}$$

where $\mu \in \mathbb{R}$ is the outer-loop learning rate. This process continues until convergence is reached.

2.4 FSL Evaluation

After meta-training, it is expected that the meta-trained model f_θ is able to quickly adapt to new tasks. To evaluate how well f_θ generalizes to new tasks, multiple rounds of training and evaluation using N-way K-shot tasks from $\mathcal{D}_{\text{test}}$ are performed.

More specifically, k evaluation instances are created, each one used in one evaluation round. An evaluation instance consists of a pair $(\mathcal{T}^i(\mathcal{D}_{\text{test}}), T^i)$, where $\mathcal{T}^i(\mathcal{D}_{\text{test}})$ is a N-way K-shot test task and T^i is a set of N examples sampled from $\mathcal{D}_{\text{test}}$, one from each of the N classes, ensuring none of them is in $\mathcal{T}^i(\mathcal{D}_{\text{test}})$. Then, f_θ is trained on each one of the $\mathcal{T}^i(\mathcal{D}_{\text{test}})$ tasks, and the accuracy on their respective T^i are recorded. To obtain the final score, the accuracy over the k runs are averaged (see Algorithm 2).

Algorithm 2 Evaluation

Require: $N, K \in \mathbb{N}, \theta \in \mathbb{R}^d$
  for $i = 1, \ldots, k$ **do**
    Sample N-way K-shot test task $\mathcal{T}^i(\mathcal{D}_{\text{test}})$ and N evaluation examples T^i
    Train f_θ (using SGD or ADAM) on $\mathcal{T}^i(\mathcal{D}_{\text{test}})$ to obtain θ_i
    Use f_{θ_i} to compute accuracy a^i over T^i
  end for
  return $\frac{1}{k} \sum_{i=1}^{k} a^i$

Thus, the evaluation protocol provides an expected average performance of a trained model when it is further trained for a completely new few-shot task (in our case, new classes).

3 Methodology

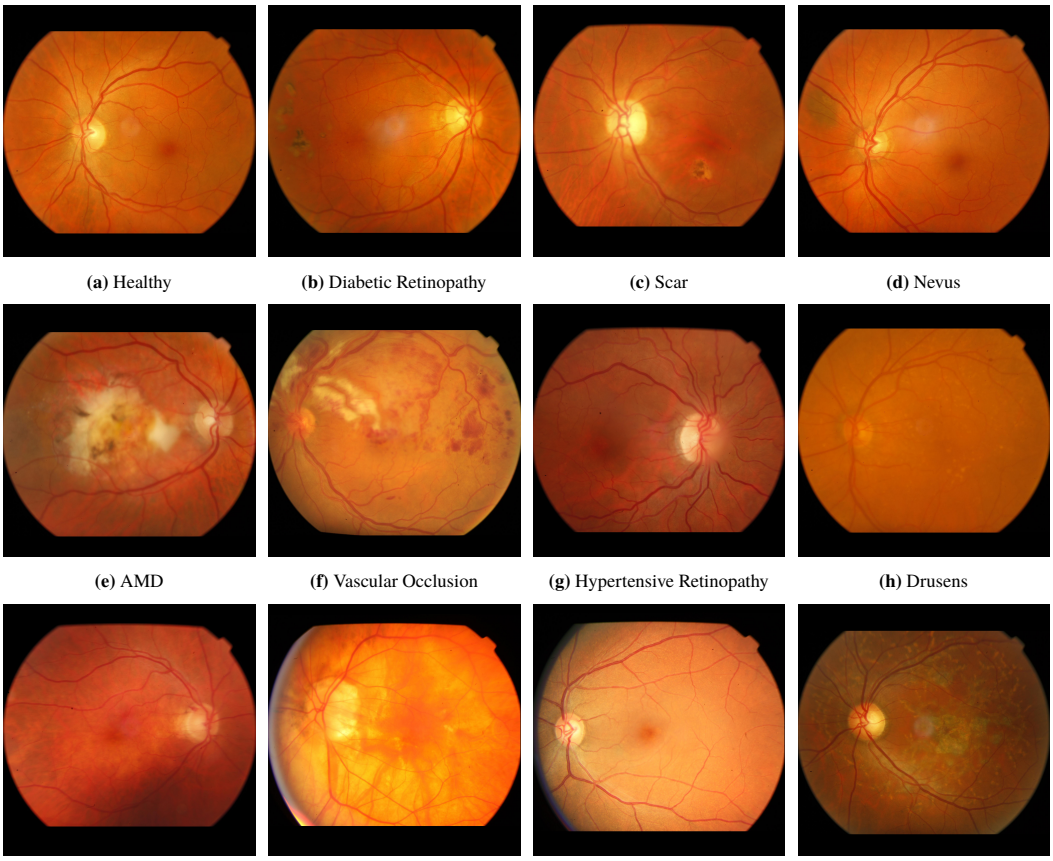
We propose a set of experiments designed to thoroughly assess the Reptile algorithm on the retinal disease classification problem. In this section, we detail the two datasets (BRSET and APTOS 2019) used in the experiments, and outline the rationale of each experiment. The second dataset, APTOS 2019, is used only for domain-specific pretraining of the models.

3.1 Datasets

BRSET: *Brazilian Multilabel Ophthalmological Dataset* [Nakayama *et al*., 2024b,a] is a public eye fundus image dataset collected from 8,524 patients at ophthalmological centers in São Paulo, Brazil. It consists of 16,226 images with annotations of 13 eye disease labels, image quality information, patient demographics, comorbidities, and other related data.

The eye disease labels, annotated by experts, are "diabetic retinopathy", "macular edema", "scar", "nevus", "AMD", "vascular occlusion", "hypertensive retinopathy", "drusen", "hemorrhage", "retinal detachment", "myopic fundus", "increased cup disc", and "other". A "healthy" image and one example from each of the classes, except "macular edema" and "retinal detachment", are shown in Fig. 1.

The BRSET dataset comprises multi-label images, where each image may be annotated with multiple eye disease labels. To align the dataset with the requirements of a meta-learning framework, we filtered out all images annotated with more than one disease label, retaining only single-label instances. Additionally, we excluded images deemed to be of "inadequate" quality based on the dataset's quality assessments. To address class imbalance, we limited the number of healthy images to 100.



(a) Healthy **(b)** Diabetic Retinopathy **(c)** Scar **(d)** Nevus

(e) AMD **(f)** Vascular Occlusion **(g)** Hypertensive Retinopathy **(h)** Drusens

(i) Hemorrhage **(j)** Myopic Fundus **(k)** Increased Cup Disc **(l)** Other

Figure 1. Images sampled from BRSET, except from the "macular edema" and "retinal detachment" classes (not used in this work). Images are centralized on a black background square frame for visualization purposes.

Table 1. Count of images per class before and after filtering. Underlined classes compose $\mathcal{D}_{\text{test}}$. The remaining classes (except "retinal detachment" and "macular edema") compose $\mathcal{D}_{\text{train}}$.

Disease class	Count before	Count after
increased cup disc	3202	2027
drusens	2807	1710
diabetic retinopathy	1046	480
other	758	446
myopic fundus	268	158
scar	290	150
AMD	366	132
healthy	8460	100
hypertensive retinopathy	283	93
nevus	134	63
vascular occlusion	103	41
hemorrhage	96	23
macular edema	402	16
retinal detachment	7	5

We discarded the two smallest classes after filtering, "retinal detachment" and "macular edema", due to the extremely reduced number of instances. Then, for $\mathcal{D}_{\text{test}}$, we chose the three least populated disease classes ("nevus", "hemorrhage", "vascular occlusion") plus "healthy". The remaining eight classes were included in $\mathcal{D}_{\text{train}}$. The number of examples of the dataset before and after selection is presented in Table 1.

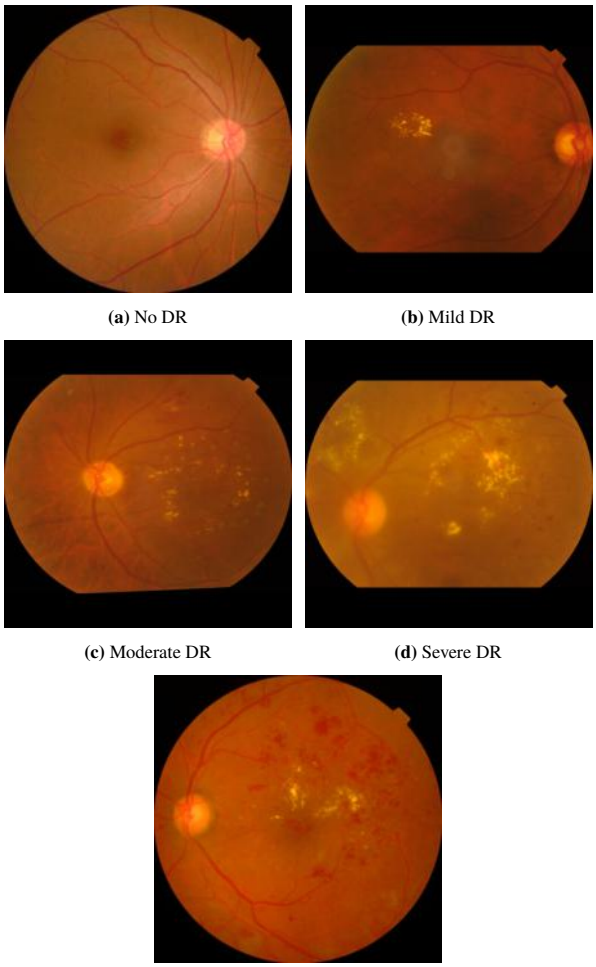
For dealing with varying image sizes and aspect ratios, we implemented the pre-processing technique used in Men *et al.* [2023]. Accordingly, we removed external black columns and rows of each image and then padded it as needed to achieve a square ratio. Finally, we resized the square image to 224×224 and then normalized it using the mean and standard deviation values specific to the chosen backbone model.

APTOS 2019: *Asia Pacific Tele-Ophthalmology Society 2019 Blindness Detection* dataset [Karthik *et al.*, 2019] was collected in rural India and organized by Aravind Eye Hospital. It consists of 3,662 images annotated with Diabetic Retinopathy (DR) grades following the *International Clinical Diabetic Retinopathy* (ICDR) severity scale with 5 classes: "no DR", "mild DR", "moderate DR", "severe DR", and "proliferative DR". Figure 2 shows some images of the positive class, with some degree of DR.

The APTOS 2019 dataset is utilized exclusively for model pretraining in this study, either as an initial training phase from scratch or as a preliminary fine-tuning step. This approach aims to introduce domain-specific knowledge into the models prior to meta-learning. The dataset is used in its entirety, without excluding any samples. The same preprocessing procedures applied to the BRSET dataset – specifically, image size standardization – were also applied to APTOS 2019 to ensure consistency across datasets.

3.2 Training Methods

Our primary research question is whether Reptile's meta-learning framework offers a performance advantage over con-



(a) No DR **(b)** Mild DR

(c) Moderate DR **(d)** Severe DR

(e) Proliferative DR

Figure 2. Images sampled from APTOS 2019 Dataset.

ventional transfer learning approaches in the context of eye disease classification, particularly under conditions of limited training data. To answer this question, we compare Reptile against baseline models that employ standard transfer learning approaches. To rigorously assess this hypothesis, we implement five distinct training methods and evaluate the resulting models on test tasks according to the protocol in Algorithm 2.

The training methods assume a common base architecture, and are illustrated in Figure 3. In Baseline 1, we apply the evaluation protocol directly to the ImageNet-pretrained model, without any further training. Baseline 2 is obtained by standard fine-tuning of the ImageNet-pretrained model on BRSET, using the training set $\mathcal{D}_{\text{train}}$. Reptile Meta-Learning (Reptile-plain) consists of training the ImageNet-pretrained model with the Reptile algorithm on $\mathcal{D}_{\text{train}}$.

Reptile-DS and Reptile-DSFT aim to evaluate the effect of domain-specific pretraining prior to Reptile training on BRSET[1]. In Reptile-DS, the model is trained from scratch using the APTOS 2019 dataset before applying Reptile training on $\mathcal{D}_{\text{train}}$. In Reptile-DSFT, the ImageNet-pretrained model is first fine-tuned on APTOS 2019, and then trained using the Reptile algorithm on BRSET.

[1]Reptile-DS and Reptile-DSFT: Here DS and FT stand for, respectively, "domain-specific" and "fine-tuning".

3.3 Data Augmentation Methods

As aforementioned, FSL literature indicates that data augmentation plays a crucial role in enhancing meta-learning performance [Ni *et al*., 2021; Singh *et al*., 2021]. Building on this established research, we aim to systematically investigate how different augmentation strategies affect Reptile's learning capabilities in our application context. We used the following augmentation methods.

1. **Basic Augmentation** Consists of a set of simple image transformations, composed of random horizontal flips and adjustments to contrast, saturation, and hue levels [Men *et al*., 2023]. These transformations create subtle variations in the training data while preserving the essential characteristics of the original images.
2. **MixUp** This technique creates synthetic training examples by computing weighted combinations of image pairs and their corresponding labels. By interpolating between different samples, MixUp helps the model learn smoother decision boundaries and develop more robust feature representations, ultimately leading to better generalization performance [Zhang *et al*., 2018].
3. **CutMix** This approach randomly replaces rectangular regions in one image with patches from another image, while proportionally mixing their labels. By creating these composite images, CutMix forces the model to learn from partial object features and diverse contextual information, leading to more robust and localization-aware feature representations [Yun *et al*., 2019].
4. **All Augmentations** Combines Basic Augmentation, MixUp, and CutMix to assess the cumulative effect of all techniques applied together.

Following insights from Ni *et al*. [2021], we also assess where these augmentations should be applied. Specifically, we test augmentation at different stages of Reptile training:

- Inner-Loop (I): Applied during meta-training.
- Evaluation (E): Applied only during the evaluation stage.
- Both (I+E): Applied in both training and evaluation stages.

We emphasize that MetaMed [Singh *et al*., 2021] evaluated application of augmentation only in the inner loop (I). On the other hand, although Ni *et al*. [2021] evaluated augmentation on both stages, they did not considered the Reptile algorithm. Our study extends these studies by testing the augmentation methods with Reptile for the above three stage combination cases.

3.4 Explainability

Given a classification model, one can use Gradient-weighted Class Activation Mapping (Grad-CAM) [Gildenblat and contributors, 2021] to visualize which regions of an input image contribute most to a specific output prediction. This technique involves selecting a layer and computing the gradient of the target class score with respect to the feature maps in that layer. These gradients are then spatially pooled to obtain weights for each channel, which are used to compute a weighted combination of the corresponding feature maps. The resulting

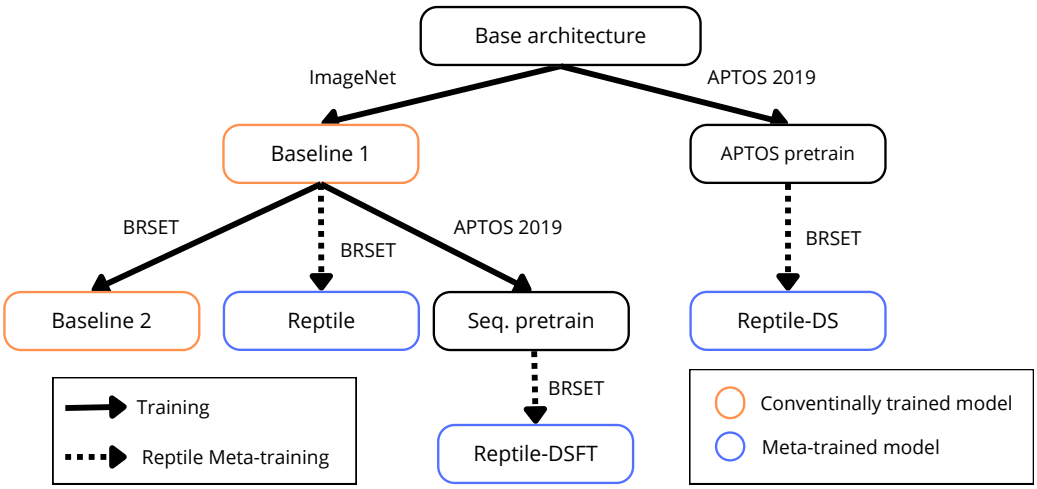


Figure 3. Overview of the training methods explored in this study. Models with orange borders represent conventionally trained models, while those with blue borders are meta-trained models. Solid arrows indicate training, and dotted arrows represent Reptile meta-training.

activation map is then up-sampled to the size of the input image and overlaid as a heatmap, highlighting the areas that most strongly influenced the model's decision.

To evaluate whether the meta-learned model is capable of identifying the relevant disease-specific features after fine-tuning, we propose a novel method for generating an aggregated heatmap by slightly modifying the evaluation procedure described in Algorithm 2. Let f_θ denote the model under consideration (e.g., Reptile or a baseline), and let A be a class of interest from the test set $\mathcal{D}_{\text{test}}$. We select a single image x belonging to class A and temporarily remove it from $\mathcal{D}_{\text{test}}$. The objective is to generate a heatmap for image x, which is performed following the steps detailed in Algorithm 3.

Algorithm 3 Heatmap generation

Require: $N, K \in \mathbb{N}$, $\theta \in \mathbb{R}^d$, x in class A, k
  Remove x from $\mathcal{D}_{\text{test}}$
  $j \leftarrow 0$
  while $j < k$ **do**
    Sample N-way K-shot test task $\mathcal{T}^i(\mathcal{D}_{\text{test}})$
    Train f_θ (using SGD or ADAM) on $\mathcal{T}^i(\mathcal{D}_{\text{test}})$ to obtain θ_i
    if f_{θ_i} correctly predicts x being of class A **then**
      $j \leftarrow j + 1$
      $\text{map}_j(x) \leftarrow \text{GradCAM}(f_{\theta_i}, x)$
    end if
  end while
  return $\frac{1}{k}\sum_{j=1}^{k} \text{map}_j(x)$

Each time the meta-trained model f_θ is trained using one N-way K-shot test task, we verify whether the updated model correctly classifies x. If so, a Grad-CAM heatmap for x is generated using the new model. Parameter k specifies the number of Grad-CAM heatmaps to be aggregated. By computing an aggregated heatmap, we aim to analyze if Reptile generates a model that is able to better recognize the patterns specific to images in class A.

4 Results and Discussion

The methods outlined in Sections 3.2, 3.3 and 3.4 comprise a comprehensive series of experiments to evaluate, using the BRSET dataset, how well Reptile performs in Few-Shot Learning scenarios.

Below, we present a detailed walkthrough of our methodology and findings, along with insights into the practical implications of our results. All experiments were conducted using models implemented in the PyTorch library.

4.1 Transfer-learning vs Reptile

In this section, we detail the experiments regarding the training methods listed in Section 3.2. We compare the baseline models with meta-learning (Reptile) models using three different base architectures: ResNet50 [He *et al.*, 2016], Swin-Tiny [Liu *et al.*, 2021], and ViT-Small [Dosovitskiy *et al.*, 2021]. These architectures are widely recognized in the literature and have an equivalent number of parameters, making them well-suited for a fair comparison in our experiments. To assess their performance, we conduct evaluations across multiple N-way K-shot task configurations, specifically for $N = 2, 3$ and $K = 5, 10, 20$. To ensure result comparability, we fix random seeds, making the evaluation process deterministic. The pretrained ImageNet weights were sourced from the PyTorch Image Models (timm) library [Wightman, 2019].

Hyperparameters: A thorough effort was made to tune the hyperparameters through a combination of grid search and manual inspection. For each architecture and training setting, we evaluated multiple values for learning rates, batch sizes, number of update steps, and data augmentation strategies. The

Table 2. Comparison of Reptile with the baselines. For each N-way K-shot setup, best result for each specific base architecture is underlined, and best overall result is in bold. (See Section 3.2 for the definition of the methods)

Architecture	Method	2-way			3-way		
		5-shot	10-shot	20-shot	5-shot	10-shot	20-shot
ResNet50	Baseline 1	59.63	62.75	68.00	41.92	50.83	52.00
	Baseline 2	64.75	69.63	76.25	50.58	56.75	59.92
	Reptile-plain	57.00	64.00	71.0	40.50	47.75	55.25
	Reptile-DS	55.13	57.63	64.63	36.08	41.50	47.75
	Reptile-DSFT	66.00	72.50	78.38	53.33	55.75	61.25
ViT	Baseline 1	61.88	66.88	71.25	49.33	54.83	55.67
	Baseline 2	60.13	59.63	64.50	39.83	46.42	48.33
	Reptile-plain	64.05	69.25	75.25	50.33	53.75	57.83
	Reptile-DS	48.75	51.63	54.63	31.75	33.25	34.67
	Reptile-DSFT	58.88	65.13	71.38	42.00	48.42	53.83
Swin	Baseline 1	65.88	68.75	76.88	50.42	54.00	61.67
	Baseline 2	66.38	70.25	74.75	**56.42**	62.58	59.67
	Reptile-plain	67.25	70.38	76.88	51.75	60.92	**69.25**
	Reptile-DS	48.05	51.88	58.50	30.50	37.25	40.92
	Reptile-DSFT	**71.88**	**76.50**	**79.00**	**56.42**	**63.83**	67.42

selection process was guided by validation performance, with particular attention paid to reducing overfitting and improving generalization across tasks. For the Reptile-based methods, we also explored different inner- and outer-loop learning rates, numbers of shots, and task batch sizes to ensure stable meta-optimization.

The Adam optimizer was used for all experiments, with a learning rate of 10^{-3} for the ResNet50 architecture and 10^{-4} for the Transformer-based architectures. For Reptile training, these values were used as the inner-loop learning rates.

Baseline 2 and the pretrained and fine-tuned models used in Reptile-DS and Reptile-DSFT were trained with a fixed batch size of 16 images for up to 100 epochs. To mitigate overfitting, early stopping was employed, selecting the final model based on the lowest validation loss. Additionally, class weights were incorporated to address data imbalance.

Pretraining and fine-tuning of the models used in Reptile-DS and Reptile-DSFT were conducted on domain-specific data from APTOS 2019, using CrossEntropyLoss to optimize classification of diabetic retinopathy severity. To further mitigate overfitting, we applied a combination of CutMix, MixUp, and standard data augmentation techniques.

For all Reptile training, the batch size was set to 10 for 2-way tasks and 15 for 3-way tasks. The number of inner-loop update steps was set to 2, 4, and 8 for the 5-shot, 10-shot, and 20-shot tasks, respectively. In the outer loop, we used a learning rate of 0.1 with linear decay, decreasing the learning rate from its initial value to zero over 1000 update steps, using a batch size of 5 tasks.

For the Evaluation stage, we used Adam with the previously mentioned learning rate, 52 update steps, 400 runs, and the same inner-loop batch size. To ensure a fair comparison, we applied the same hyperparameters when testing meta-learners (Reptile) and the baselines. Additionally, we set random seeds to guarantee the reproducibility and determinism of the evaluation task generation, minimizing variability across runs.

The five training methods were run for the three base architectures using data from $\mathcal{D}_{\text{train}}$ and evaluated on data from $\mathcal{D}_{\text{test}}$, as specified in Section 3.1.

Evaluation results are presented in Table 2. Overall, Swin-based models achieve the best performance. Next, we summarize the key findings from this experiment.

Reptile wins over transfer learning: Reptile-plain models consistently outperform Baselines 1 and 2 for both Swin and ViT architectures, with gains of up to 7.58 percentage points on the 3-Way 20-Shot tasks, highlighting the benefits of the meta-learning paradigm for this problem.

For the ViT architecture, comparing Baseline 1 and Baseline 2 reveals that additional standard training of the base model may degrade performance. This may be explained by a degree of overfitting during ViT's training phase, indicating that a significantly larger dataset would be necessary to mitigate this effect. Furthermore, these results hint that Reptile is better suited for few-shot learning scenarios.

In contrast, the ResNet50 architecture presents a different trend. Here, Reptile is outperformed by the baselines. Notably, prior studies often use a larger number of outer steps when training CNNs [Singh *et al*., 2021], which could potentially enhance Reptile's performance with ResNet50. However, to ensure a fair comparison across architectures within a similar training time, we maintained the same hyperparameter settings and leave further investigation for future work.

Fine-tuning with domain specific data helps: Reptile-DSFT consistently demonstrates superior performance compared to Reptile-plain, achieving significant improvements across both ResNet50 and Swin architectures. Notably, the gains reach up to 9 percentage points on 2-Way-5-Shot for the ResNet50 architecture. This highlights the considerable advantages of adding a domain-specific training stage prior to the Reptile training.

In contrast, the ViT architecture exhibits a different trend. In this case, Reptile-plain outperforms models that undergo additional pretraining. This may be, again, explained by ViT's training phase being more prone to overfitting, suggesting that a significantly larger dataset, along with stronger data aug-

Table 3. Augmentation techniques applied on different stages of Reptile-DSFT with Swin architecture. For each N-way K-shot setup, best result for each specific augmentation is underlined, and best overall result is in bold. ↑ indicates improvement over no augmentation.

Augmentation	Where	2-way			3-way		
		5-shot	10-shot	20-shot	5-shot	10-shot	20-shot
None	-	71.88	76.50	79.00	56.42	**63.83**	67.42
Basic	I	72.63↑	76.75↑	78.50	58.75↑	63.50	67.75↑
	E	72.63↑	76.75↑	78.38	58.58↑	63.50	67.75↑
	I+E	72.50↑	76.75↑	78.38	58.58↑	63.50	67.75↑
MixUp	I	72.63↑	75.25	79.75↑	**59.75**↑	60.83	69.83↑
	E	68.13	73.63	80.38↑	55.00	61.25	66.58
	I+E	68.00	74.38	79.25↑	54.58	62.67	**71.00**↑
CutMix	I	70.00	74.75	80.63↑	56.92↑	62.33	69.33↑
	E	66.75	75.13	79.75↑	55.58	62.92	68.50↑
	I+E	67.13	74.38	**82.38**↑	56.25	62.83	69.67↑
All	I	**73.88**↑	**78.25**↑	81.00↑	58.50↑	61.17	68.08↑
	E	71.25	78.13↑	82.13↑	56.67↑	63.08	69.08↑
	I+E	70.63	77.88↑	81.88↑	59.50↑	62.58	67.75↑

mentation strategies, may be required to mitigate this effect.

ImageNet pretraining wins over domain-specific pretraining: The approach of using Reptile after pretraining on ImageNet (Reptile-plain and Reptile-DSFT) consistently demonstrates superior performance when compared to using Reptile pretrained solely on a small, domain-specific dataset (Reptile-DS), across all tested architectures. The performance gains were particularly substantial, with an improvement of up to approximately 28.33 percentage points in the 3-Way-20-Shot setting when using the Swin architecture. This highlights the significant advantage of pretraining on a large and diverse dataset such as ImageNet, which appears to enhance the model's generalization capability and its ability to adapt to new, domain-specific tasks. This effect may be attributed to the stark contrast in dataset sizes: ImageNet contains roughly 14 million images, whereas APTOS comprises only 3,662.

The best overall performance is achieved by the Reptile-DSFT with Swin architecture. Therefore, this configuration was selected for the second experiment.

4.2 Effects of Data Augmentation

Using the same experimental setup of the selected model (Swin model with Reptile-DSFT), we apply the augmentation methods described in Section 3.3. These methods are tested across all combinations of $N = 2, 3$ and $K = 5, 10, 20$ N-way K-shot tasks, considering different application locations (I, E, or I+E). The augmentation techniques are implemented using the torchvision library [TorchVision Maintainers and contributors, 2016], with α set to 1.0.

Results: The results are presented in Table 3. Overall, we see that data augmentation, regardless of its type, results in improved evaluation accuracy, with the maximum improvement observed in our experiments being 3.58% (3-way 20-shot, with MixUp). The only exception is the 3-way 10-shot, where none of the augmentation strategies surpassed the no-augmentation baseline. We believe this is just an unfortunate coincidence, and we leave further investigation as future work.

Comparing the three strategies (I, E, I+E), best results shot-wise involve either I or I+E. This hints to the importance of data augmentation in the internal loop of the training stage (I), but also shows that when it is combined with augmentation in the evaluation stage (I+E) it may lead to further improvements. Nevertheless, there is no clear winner between the two.

Another noticeable characteristic is that data augmentation seems to be more effective for larger shot sizes. For the 5-shot cases, there are 13 improvement cases out of 24, while for the 20-shot cases, there are 20 improvement cases out of 24. This is reasonable since the larger the base dataset, more diverse augmentation is possible compared for instance to a very small base dataset of only 5 images.

Prediction confidence score analysis: To further investigate the differences between applying augmentations in the Inner-loop versus in the Evaluation stage, we examine the distribution of confidence scores across predictions to gain insights into the model's reliability and robustness.

To that end, given a model, let $\hat{p}_x$ denote the predicted score for the output class of a test input x during evaluation. It is of particular interest to investigate the model's behavior when making high-confidence predictions—specifically, when $\hat{p}_x \geq T$, for a chosen confidence threshold, such as $T = 0.8$.

To quantify this, we compute the proportion of correct predictions when the model is highly confident (say, $T = 0.8$), as follows:

$$\text{Acc}_{\geq T} = \frac{\#\text{ correct predictions with } \hat{p}_x \geq T}{\#\text{ total predictions with } \hat{p}_x \geq T}$$

This proportion can be interpreted as the accuracy computed with respect to the subset of examples for which a model makes a prediction with high confidence.

Table 4 shows $\text{Acc}_{\geq T}$, for $T = 0.8$ (confidence of 80%). As shown, regardless of the specific N-way K-shot configuration, higher overall $\text{Acc}_{\geq 0.8}$ scores are observed when data augmentation – specifically using methods such as CutMix or MixUp – is applied during the evaluation stage.

High values of $\text{Acc}_{\geq T}$ for a large confidence threshold T indicate that the model's predictions are more likely to be

Table 4. Recall $\text{Acc}_{\geq T}$ for the cases where Reptile-DSFT with Swin architecture model prediction confidence is above 80% ($T = 0.8$). For each N-way K-shot setup, best result for each specific augmentation is underlined, and best overall result is in bold.

Augmentation	Where	2-way			3-way		
		5-shot	10-shot	20-shot	5-shot	10-shot	20-shot
Basic	I+E	70.41	78.84	82.04	59.90	64.06	69.43
	I	71.90	78.48	79.01	61.30	68.38	73.88
	E	72.60	76.90	84.26	56.82	65.95	71.86
MixUp	I+E	80.81	79.48	84.67	67.90	73.58	75.86
	I	76.75	77.09	81.47	57.98	64.00	69.89
	E	79.89	82.96	88.47	72.97	68.18	83.47
CutMix	I+E	77.53	79.67	89.53	65.74	69.91	81.44
	I	72.29	77.31	79.91	62.61	59.91	66.01
	E	80.34	82.21	84.21	68.29	67.57	83.08
All	I+E	77.16	86.14	**90.75**	**75.00**	72.60	**88.64**
	I	70.51	75.84	81.84	60.00	64.02	69.51
	E	**81.91**	**87.21**	88.11	62.67	**79.49**	86.75

correct when it assigns high confidence. This is a highly desirable property, particularly in the medical domain, as it enhances the model's trustworthiness and reliability, which are essential when supporting physicians in disease diagnosis.

To further analyze confidence distributions, Figure 4 shows the confidence distribution plots for 100 correct and incorrect predictions, comparing augmentations applied during the Inner-loop (I) and during both the Inner-loop and Evaluation stages (I+E) for Basic and MixUp augmentations. Top two rows in the plot panel refer to basic augmentation, and the two bottom rows refer to MixUp augmentation. Columns in the plot panel refer respectively to 2-way 5-shot, 2-way 10-shot, and 2-way 20-shot setups. In each plot, in the x-axis is the prediction confidence ($\hat{p}_x$) and in the y-axis is the number of predictions made by the model with each confidence value. The number of correct predictions is shown in blue while the number of incorrect predictions is shown in orange. The curves are approximate probability density distributions.

The plots reveal that when Basic augmentation is employed (two top rows in the plot panel), the models tend to exhibit extreme confidence in their predictions regardless of being correct or incorrect. Regardless the augmentation type, when it is applied in the Inner-loop (I) only – rows 1 and 3 in the plot panel, the models also tend to exhibit extreme confidence in their predictions, whether they are correct or incorrect. On the other hand, when Mixup augmentation is applied in both the Inner-loop and Evaluation stages (I+E) – last row in the plot panel, the models display a more nuanced confidence pattern. In particular, it is interesting to note that they are less confident when making incorrect predictions and more confident when making correct ones.

Although not plotted in the panel, similar results are obtained when CutMix is used as the data augmentation method. In summary, the plots indicate that when augmentation (specifically MixUp or CutMix) is applied not only in the inner loop (I) but also in the Evaluation stage (I+E), models tend to exhibit higher confidence when they are actually correct.

To understand how the pattern observed in the plots of Figure 4 and quantified in Table 4 changes with varying confidence levels, we show in Figure 5 a set of plots of $\text{Acc}_{\geq T}$ for $T = 0.8, 0.85, 0.90, 0.95$. The top panel refers to Basic augmentation while the bottom panel refers to CutMix augmentation. Columns in the plot panel refers to the 2-way 5-shot, 2-way 10-shot, and 2-way 20-shot setups, respectively.

The plots show that, except for the 2-way 5-shot with Basic data augmentation, application of augmentation in the evaluation stage leads to improvements of $\text{Acc}_{\geq T}$. This improvement is particularly noticeable when CutMix is employed, reaching accuracy improvements of more than 10%. Moreover, we see a trend that indicates an increase in the gap with the increase in the confidence level.

4.3 Explainability Analysis

Besides the confidence level discussed in the previous section, explainability is a key element to enhance the trustworthiness of a prediction model. Thus, we generated heatmaps following the method described in Section 4.3. For each image, we averaged Grad-CAM heatmaps over 100 times the image was correctly classified in the evaluation process.

Figure 6 shows the generated heatmaps. They are all part of the 2-way K-shot setups, with the two classes considered including the "healthy" class and one second class among "hemorrhage", "nevus" and "vascular occlusion" classes. For each setup, a single K-shot and a single test image was selected. The first column in the panel corresponds to $K = 5$ and classes "healthy" and "hemorrhage"; second column to $K = 10$ and classes "healthy" and "nevus"; and third column to $K = 20$ and classes "healthy" and "vascular occlusion". Rows of the panel correspond to the training methods. The models, except Reptile-DSFT, are the ones based on the Swin architecture listed in Table 2. For Reptile-DSFT, for each 2-way K-shot setup, we considered models trained with data augmentation and chose the one that exhibited best performance according to Table 3.

In the first (hemorrhage class) and second (nevus class) columns, Baseline 2 and Reptile-DSFT appear to highlight the relevant image regions, whereas the other models emphasize less relevant areas. In the third column (vascular occlusion class), all models except Reptile-DS seem to highlight part of the correct region. However, the first four models generally produce more diffuse Grad-CAM maps, sometimes extending outside the circular fundus region. In contrast, the maps produced by Reptile-DSFT are sharper, with activations more

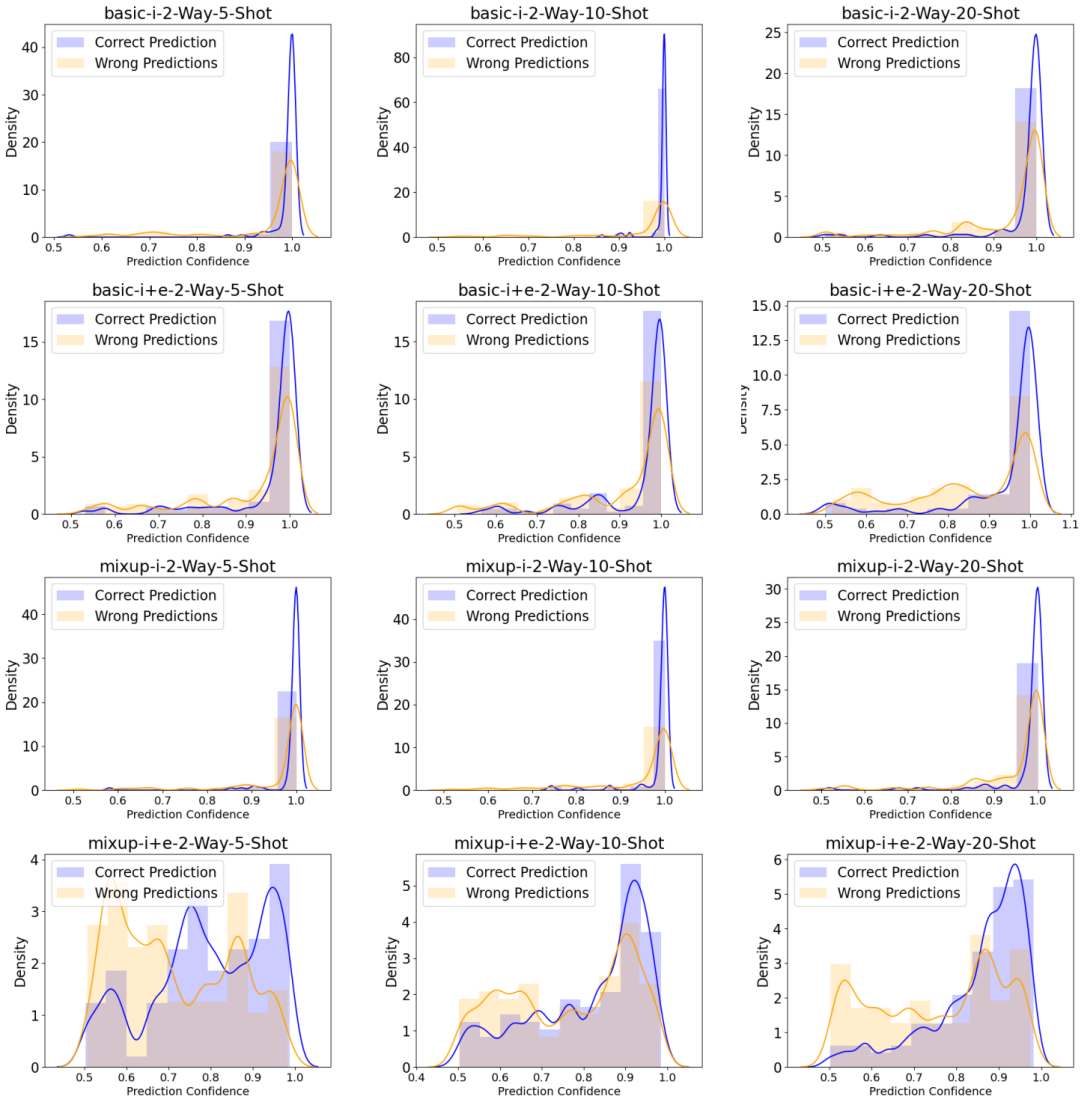


Figure 4. Prediction confidence distribution across N-way K-shot tasks of the Reptile-DSFT with Swin architecture. First two rows in the panel refer to Basic data augmentation and two bottom rows refer to MixUp data augmentation.

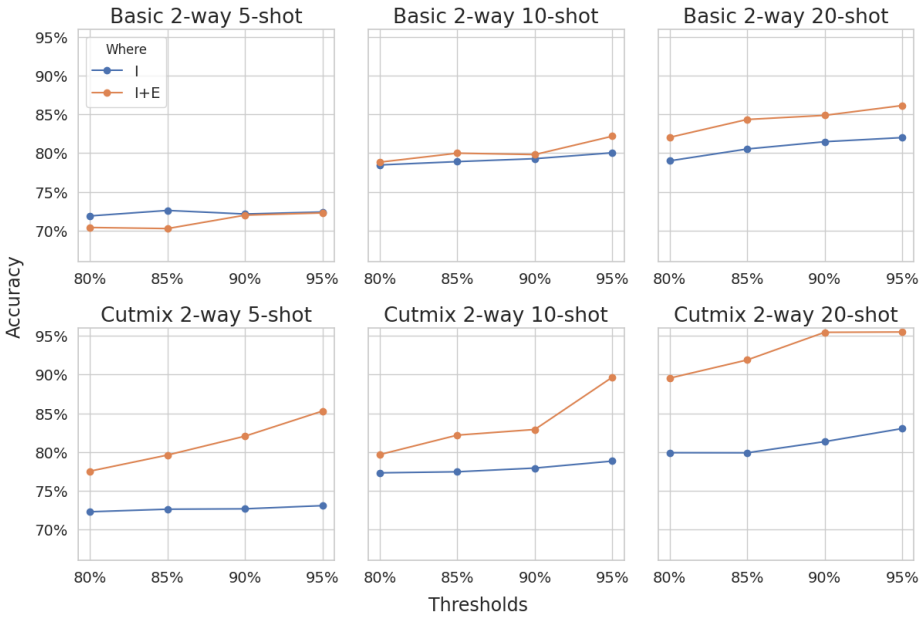


Figure 5. Reptile-DSFT with Swin architecture: x-axis shows confidence level T in percentage, and y-axis show $\text{Acc}_{\geq T}$.

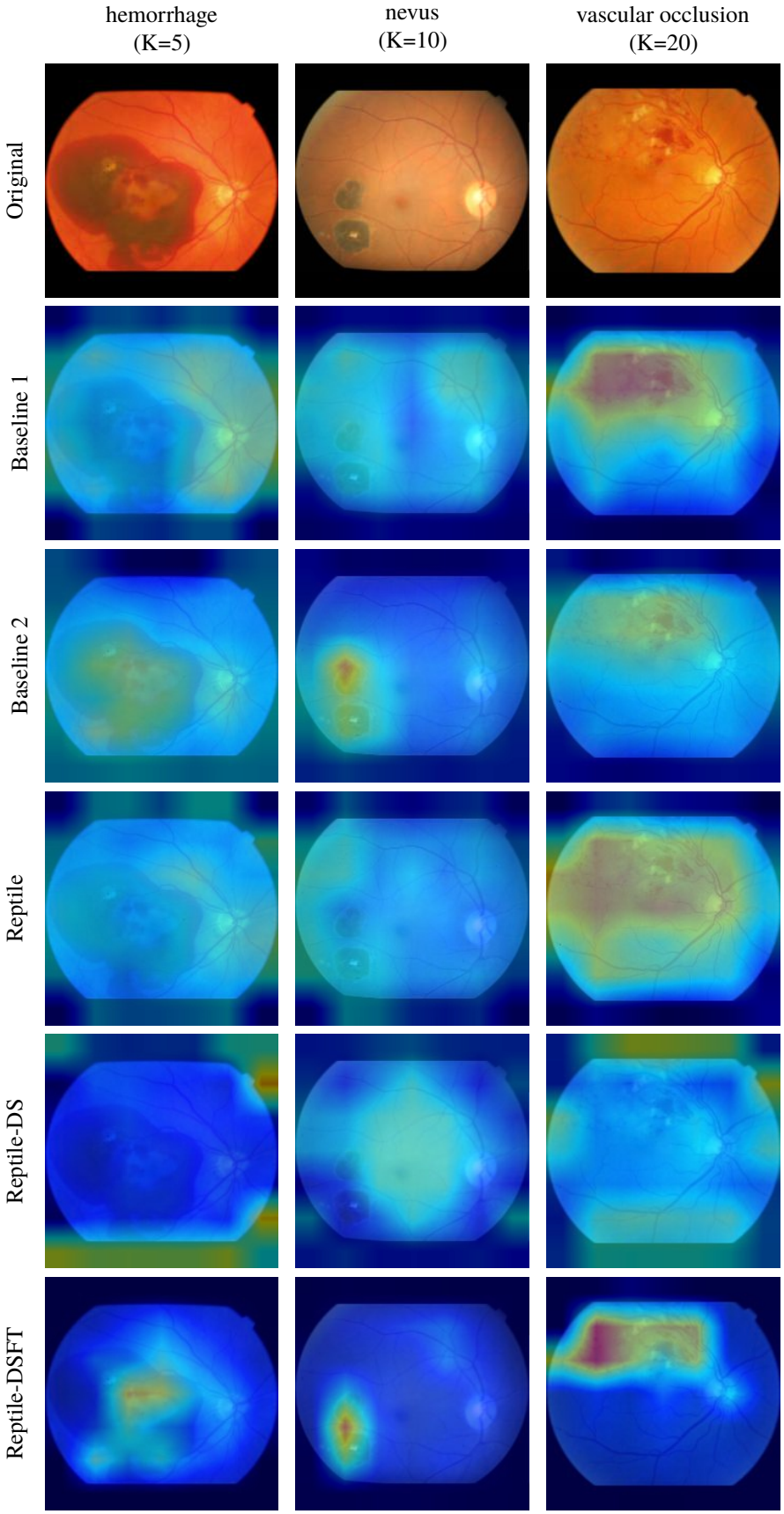


Figure 6. Heatmap examples: varying training methods for the 2-way K-shot setup (with $K = 5, 10, 20$) on "healthy" against "hemorrhage", "nevus" and "vascular occlusion" classes.

concentrated on the relevant regions.

Although the examples presented in Fig. 6 provide only a limited case analysis, they complement the quantitative analyses from the previous sections. In particular, the sharper and more localized Grad-CAM maps produced by Reptile-DSFT are consistent with the stronger performance observed in the previous experiments.

5 Conclusion

This study aimed to evaluate Reptile's ability to build a model capable of learning to classify new, previously unseen, eye diseases with just a few examples.

An extensive experimental setup was proposed, exploring three model architectures (ResNet50, ViT, Swin), two standard training and three Reptile based meta-training methods, and non-trivial data augmentation methods employed in the training and evaluation stages of the models.

Reptile presented superior performance than simple transfer learning. Among the architectures, Swin was clearly superior. Moreover, further training models pretrained on APTOS 2019 (a second eye fundus image dataset) reveals that final models present improved results.

Regarding data augmentation, although no specific augmentation type stood out, overall they lead to performance improvement. The gain is more noticeable for larger shots. A novelty in our study is the employment of data augmentation in the evaluation stage. The results indicate that, in this case, models generated higher confidence values when they are correct. Moreover, explainability analysis, based on a newly proposed Grad-CAM map aggregation method, indicates that the activations of the winning model is more focused on the anomaly region, reinforcing the observed quantitative results.

Based on the observed results, we conclude that Reptile has potential to learn eye diseases in retina images in few-shot scenarios. Thus, it can be employed to learn new disease classes or when adapting the model for a new dataset, such as to the newly released mBRSET, a Mobile Brazilian Retinal Dataset [Nakayama *et al*., 2024c].

Since using APTOS 2019 indicated that models benefit from additional training on domain specific data prior to meta-training, as a future work we would like to explore self-supervised approaches for the pretraining of the model architectures to take advantage of large volume of unlabeled images. We also would like to explore other few-shot learning approaches.

Declarations

Acknowledgements

The authors thank IME/USP Vision Lab where experiments were run, and thank L. F. Nakayama and F. Malerbi for the help to get access to BRSET.

Funding

This research was partly funded by São Paulo Research Foundation (FAPESP) [grants #2022/15304-4, #2022/11645-1], Ministério da Ciência, Tecnologia e Inovações (MCTI/Brazil) [grant PPI-Softex - TIC 13 - 01245.010222/2022-44, law 8.248], and National Council for Scientific and Technological Development (CNPq/Brazil) [fellowship grant #307701/2025-5].

Authors' Contributions

GJP was the main author of the preliminary study and contributed to the conception of this extended version, coordination of the experiment design and execution, result analysis, writing and final revision of the manuscript. DJCA contributed to experiment execution, result analysis, and writing of the manuscript. LMSS contributed to data preprocessing, experiment execution and writing of the manuscript. EME contributed to experiment execution, result analysis, and writing of the manuscript. NSTH contributed with supervision, conception of the extended version, result analysis, writing and final revision of the manuscript.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

The datasets used in this study are publicly available: BRSET is available at `https://doi.org/10.13026/1pht-2b69` upon accreditation, and APTOS 2019 is available at Kaggle. The codes developed for this study are available at `github.com/gabjp/FSL-retina`.

References

Cai, A., Chen, L., Chen, Y., Fang, J., Sun, M., and Chuan, Z. (2022). Pre-mocodiagnosis: Few-shot ophthalmic diseases recognition using contrastive learning. In *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 2059–2066. DOI: 10.1109/BIBM55620.2022.9994890.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *9th International Conference on Learning Representations (ICLR)*. DOI: 10.48550/arxiv.2010.11929.

Finn, C., Abbeel, P., and Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *34th International Conference on Machine Learning*, pages 1126–1135. DOI: 10.48550/arxiv.1703.03400.

Gildenblat, J. and contributors (2021). PyTorch library for CAM methods. Available at:`https://github.com/jacobgil/pytorch-grad-cam`.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778. DOI: 10.1109/CVPR.2016.90.

Hospedales, T., Antoniou, A., Micaelli, P., and Storkey, A. (2022). Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5149–5169. DOI: 10.1109/TPAMI.2021.3079209.

Huang, W., Huang, Y., and Tang, X. (2022). LesionPaste: One-Shot Anomaly Detection for Medical Images. *arXiv:2203.06354*. DOI: 10.48550/arxiv.2203.06354.

Karthik, Maggie, and Dane, S. (2019). APTOS 2019 Blindness Detection. Available at:`https://kaggle.com/competitions/aptos2019-blindness-detection`. Kaggle.

Kawasaki, R. and Grauslund, J. (2019). Chapter 2 - Clinical motivation and the needs for RIA in healthcare. In Trucco, E., MacGillivray, T., and Xu, Y., editors, *Computational Retinal Image Analysis*, The Elsevier and MICCAI Society Book Series, pages 5–17. Academic Press. DOI: 10.1016/B978-0-08-102816-2.00002-2.

Koch, G., Zemel, R., and Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*. Avaialble at:`https://www.semanticscholar.org/paper/Siamese-Neural-Networks-for-One-Shot-Image-Koch/f216444d4f2959b4520c61d20003fa30a199670a`.

Li, N., Li, T., Hu, C., Wang, K., and Kang, H. (2021). A Benchmark of Ocular Disease Intelligent Recognition: One Shot for Multi-disease Detection. In *Benchmarking, Measuring, and Optimizing*, volume 12614, pages 177–193. Cham. Series Title: Lecture Notes in Computer Science. DOI: 10.1007/978-3-030-71058-$3_1$1.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. (2021). Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In *International Conference on Computer Vision (ICCV)*, pages 9992–10002. DOI: 10.1109/ICCV48922.2021.00986.

Mahapatra, D., Ge, Z., and Reyes, M. (2022). Self-supervised generalized zero shot learning for medical image classification using novel interpretable saliency maps. *IEEE Transactions on Medical Imaging*, 41(9):2443–2456. DOI: 10.1109/TMI.2022.3163232.

Men, Y., Fhima, J., Celi, L. A., Ribeiro, L. Z., Nakayama, L. F., and Behar, J. A. (2023). DRStageNet: Deep Learning for Diabetic Retinopathy Staging from Fundus Images. *arXiv:2312.14891*. DOI: 10.48550/arxiv.2312.14891.

Nakayama, L. F., Goncalves, M., Zago Ribeiro, L., Santos, H., Ferraz, D., Malerbi, F., Celi, L. A., and Regatieri, C. (2024a). A Brazilian Multilabel Ophthalmological Dataset (BRSET). *PhysioNet*. Version 1.0.1. DOI: 10.13026/1pht-2b69.

Nakayama, L. F., Restrepo, D., Matos, J., Ribeiro, L. Z., Malerbi, F. K., Celi, L. A., and Regatieri, C. S. (2024b). BRSET: A Brazilian Multilabel Ophthalmological Dataset of Retina Fundus Photos. *PLOS Digital Health*, 3(7):1–16. DOI: 10.1371/journal.pdig.0000454.

Nakayama, L. F., Zago Ribeiro, L., Restrepo, D., Santos Barboza, N., Dias Fiterman, R., Vieira Sousa, M. l., Pereira, A. D. A., Regatieri, C., Malerbi, F. K., and Andrade, R. (2024c). mBRSET, a Mobile Brazilian Retinal Dataset. *PhysioNet*. Version 1.0. DOI: 10.13026/qxpd-1y65.

Ni, R., Goldblum, M., Sharaf, A., Kong, K., and Goldstein, T. (2021). Data augmentation for meta-learning. In *38th International Conference on Machine Learning*, volume 139, pages 8152–8161. DOI: 10.48550/arxiv.2010.07092.

Nichol, A., Achiam, J., and Schulman, J. (2018). On first-order meta-learning algorithms. *arXiv:1803.02999*. DOI: 10.48550/arxiv.1803.02999.

Nurgazin, M. and Tu, N. (2023). A comparative study of vision transformer encoders and few-shot learning for medical image classification. In *International Conference on Computer Vision Workshops (ICCVW)*, pages 2505–2513. DOI: 10.1109/ICCVW60793.2023.00265.

Pachetti, E. and Colantonio, S. (2023). A systematic review of few-shot learning in medical imaging. *arXiv:2309.11433*. DOI: 10.1016/j.artmed.2024.102949.

Perin, G. J. and Hirata, N. S. T. (2024). Few-shot Retinal Disease Classification on the Brazilian Multilabel Ophtalmological Dataset. In *2024 37th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 01–06, Manaus, Brazil. IEEE. DOI: 10.1109/SIBGRAPI62404.2024.10716320.

Rajeswaran, A., Finn, C., Kakade, S. M., and Levine, S. (2019). Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems*, volume 32. DOI: 10.48550/arxiv.1909.04630.

Rajpoot, A. and Seeja, K. R. (2023). Enhancing rare retinal disease classification: a few-shot meta-learning framework utilizing fundus images. *Multimedia Tools and Applications*, 83(18):55731–55749. DOI: 10.1007/s11042-023-17691-x.

Roychowdhury, S. (2021). Few shot learning framework to reduce inter-observer variability in medical images. In *25th International Conference on Pattern Recognition (ICPR)*, pages 4581–4588. DOI: 10.1109/ICPR48806.2021.9412620.

Singh, R., Bharti, V., Purohit, V., Kumar, A., Singh, A. K., and Singh, S. K. (2021). MetaMed: Few-shot medical image classification using gradient-based meta-learning. *Pattern Recognition*, 120:108111. DOI: 10.1016/j.patcog.2021.108111.

Snell, J., Swersky, K., and Zemel, R. (2017). Prototypical networks for few-shot learning. In *31st International Conference on Neural Information Processing Systems*, pages 4080–4090. DOI: 10.48550/arxiv.1703.05175.

Song, Y., Wang, T., Cai, P., Mondal, S. K., and Sahoo, J. P. (2023). A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. *ACM Comput. Surv.*, 55(13). DOI: 10.1145/3582688.

Tazoar, M. S., Jyoti, O., Goni, M. O. F., and Rahman, M. M. (2024). Multiple Major Ocular Disease Classification Using Deep Learning Methods: A Comprehensive Review. In *2024 3rd International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE)*, pages 1–6, Gazipur, Bangladesh. IEEE. DOI: 10.1109/ICAEEE62219.2024.10561679.

TorchVision Maintainers and contributors (2016). Torchvision: Pytorch's computer vision library. Available at:`https://github.com/pytorch/vision`.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. (2016). Matching networks for one shot learning. In *30th International Conference on Neural Information Processing Systems*, pages 3637–3645. DOI: 10.48550/arxiv.1606.04080.

Wang, Y., Yao, Q., Kwok, J. T., and Ni, L. M. (2020). General-

izing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, 53(3). DOI: 10.1145/3386252.

Wightman, R. (2019). PyTorch Image Models. `https://github.com/rwightman/pytorch-image-models`.

Yun, S., Han, D., Chun, S., Oh, S. J., Yoo, Y., and Choe, J. (2019). CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features. In *International Conference on Computer Vision (ICCV)*, pages 6022–6031. DOI: 10.1109/ICCV.2019.00612.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. (2018). mixup: Beyond empirical risk minimization. *arXiv:1710.09412*. DOI: 10.48550/arXiv.1710.09412.