Few-shot Medical Image Classification using Vision Transformers

by

Maxat Nurgazin

Submitted to the Department of Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Data Science

at the

NAZARBAYEV UNIVERSITY

April 2023

Author . . . .
Department of Computer Science
April 27, 2023

Certified by . . . .
Nguyen Anh Tu
Assistant Professor, School of Engineering and Digital Sciences
Thesis Supervisor

Certified by . . . .
Min-Ho Lee
Assistant Professor, School of Engineering and Digital Sciences
Thesis Co-Supervisor

Accepted by . . . .
Vassilios Tourassis
Dean, School of Engineering and Digital Sciences

Few-shot Medical Image Classification using Vision Transformers

by

Maxat Nurgazin

Submitted to the Department of Computer Science
on April 27, 2023, in partial fulfillment of the
requirements for the degree of
Master of Science in Data Science

Abstract

The analysis of medical imaging is crucial to improve and facilitate diagnosis of human diseases. Recently, Vision Transformers were successfully used for this task. However, lots of data is needed to train such a model to achieve satisfying results. It may be a problem in medical imaging as some diseases are rare and scarcely represented in datasets, while manual labeling is expensive as it requires professional expertise. For that, methods of few-shot learning can be used as they deal with learning from only few examples. Therefore, this research investigates the use of different Vision Transformer architectures for medical image classification in a few-shot learning scenario using two few-shot learning algorithms, ProtoNet and Reptile. This work also proposes a new ViT architecture which combines ConViT with Squeeze and Excitation block. In addition to the main experiments, we tested Cutout, Mixup, and Cutmix data augmentation techniques to evaluate their impact on performance. Our findings indicate that Vision Transformers used with ProtoNets consistently outperform similarly-sized CNNs in the tested scenarios. Additionally, ViT small outperformed PFEMed, a specialized model for few-shot learning, on ISIC 2018 dataset in all tasks and on BreakHis x100 dataset in 2-shot-10-way and all 3-way tasks, despite being significantly smaller. Our proposed model did not perform better than a standard ConVit. However, this is a preliminary result from pre-training on a small dataset. The advanced input augmentation techniques did not yield significant performance improvements over the standard approach. In fact, most of these techniques led to worse results, with the exception of Mixup, which demonstrated some positive effects on the performance of models.

Thesis Supervisor: Nguyen Anh Tu
Title: Assistant Professor, School of Engineering and Digital Sciences

Thesis Co-Supervisor: Min-Ho Lee
Title: Assistant Professor, School of Engineering and Digital Sciences

Contents

1 Introduction **11**

2 Related works **15**

2.1 Few-shot Learning with ViT 15

2.2 Medical Image Classification 16

2.2.1 Medical Image classification and Vision Transformers 16

2.2.2 Medical Image classification and FSL 18

3 Preliminaries **21**

3.1 Transformers 22

3.2 Vision Transformer 23

3.3 Meta Learning 25

3.3.1 Initialization-based methods 26

3.3.2 Metric-based methods 28

4 Methodology **31**

4.1 Problem Definition 32

4.2 Few-shot Learning 32

4.2.1 Reptile 33

4.2.2 Prototypical Networks 34

4.3 Custom ViT 35

4.4 Advanced Augmentation techniques 36

4.4.1 Cutout 36

4.4.2 Mixup 36

4.4.3 Cutmix 37

5 Results 39

5.1 Dataset Description 39

5.1.1 BreakHis 39

5.1.2 ISIC 2018 40

5.1.3 Pap Smear 41

5.2 Models 41

5.3 Implementation Details 42

5.4 Experimental Settings 42

5.4.1 Setup 42

5.4.2 Training 43

5.4.3 Evaluation 44

5.5 Analysis 44

5.5.1 Pretrained ViT without Meta-training 44

5.5.2 Meta-Training Results 45

5.5.3 Effect of Augmentations 47

5.5.4 Comparison with other works 48

5.5.5 Custom ViT Results 49

5.6 Discussion 50

6 Conclusion 53

List of Figures

3-1 Transformer architecture presented by Vaswani et al. 21

3-2 Architecture of a Vision Transformer 23

3-3 The diagram of MAML from the original paper 26

4-1 Overall System Pipeline 31

4-2 Few-shot classification using class prototypes in ProtoNet algorithm. . 34

4-3 The architecture of our custom ViT 35

5-1 BreakHis Dataset 39

5-2 ISIC 2018 Dataset 40

5-3 Pap smear Dataset 40

List of Tables

5.1 Models used in this thesis. 41

5.2 Results without meta-training for the ISIC 2018 44

5.3 Results without meta-training for the BreakHis 44

5.4 Results of FSL on ISIC 2018 dataset 45

5.5 Results of FSL on BreakHis dataset with 100x magnification 46

5.6 Results of FSL on Pap smear dataset 46

5.7 Effect of augmentations on FSL performance 47

5.8 Comparison with MetaMed and PFEMed: ISIC 2018 dataset 48

5.9 Comparison with MetaMed and PFEMed: BreakHis x100 dataset . . 48

5.10 Comparison with MetaMed and PFEMed: Pap smear 49

5.11 Custom ViT architecture results on ISIC 2018 dataset. 49

Chapter 1

Introduction

The importance of medical image analysis (MIA) cannot be underestimated, as it enables the diagnosis of various diseases and conditions from medical imaging. With the increasing amount of such data being produced, the development of accurate and efficient automated methods for MIA has become a critical area of research. Lately, machine learning has proven to be a promising technique for this area, with recent advances in deep learning yielding state-of-the-art performance on various tasks of MIA, including the main focus of this research - medical image classification (MIC).

Recently, Convolutional Neural Networks (CNNs) have been the state-of-the-art models in various fields of computer vision, including medical imaging. The basis of CNNs is a convolution operation that works locally and provides translational equivariance. The innate bias of locality allows them to grasp local spatial features and combine them into higher-order features, which in turn lead to great results. On the other hand, CNN-based models are limited when it comes to learning long-range pixel relationships. This shortcoming of CNNs is addressed in the Transformer architecture, which was first introduced in 2017 by Vaswani et al. [36] for the task of natural language processing and quickly became the state-of-the-art in that field. It was later adapted to computer vision under the name "Vision Transformer" (ViT) by Dosovitskiy et al. [10] showing impressive performance on various computer vision tasks.

Unlike traditional CNNs, which use convolutional layers to extract features from the input images, ViTs rely on a self-attention mechanism to learn global representations of the input image. ViTs can learn both short-range and long-range input relationships. This makes them highly effective at processing and classifying large images, such as medical images, which often contain fine-grained details and complex structures. It has been demonstrated that transformers can outperform CNNs when using larger datasets or self-supervised learning in medical imaging tasks [23]. Some versions of ViTs, like DeiT [35], which utilize self-supervision, have managed to outperform CNNs even without huge datasets. Additionally, Transformers have built-in saliency maps that allow to understand model's decisions such that the field experts verify the results of the model.

Classical supervised deep-learning shows excellent results when using huge annotated datasets. However, applying deep learning may not be successful or practical if data is scarce or if human annotation is expensive. This is the exact case in some subfields of medical imaging. In recent years, few-shot learning (FSL) has emerged as a promising technique for addressing the problem of limited labeled data in MIC. FSL involves training a model to learn how to recognize and classify new objects or concepts with only a few examples. At its core, FSL tries to imitate the human's way of learning new concepts from few examples. In the work of Hu et al. [16], it was shown that a few-shot learning pipeline based on a ViT can achieve great results on standard FSL benchmarks. To our knowledge, this idea has not been used for MIC. Considering the above, the aim of this thesis is to investigate the use of ViTs in a few-shot learning scenario for MIC. However, there are no standard FSL benchmarks or datasets for medical imaging. Therefore, in this paper, we follow the work by Singh et al., MetaMed [30], in which several standard medical datasets were adapted for FSL scenarios, specifically three common medical image datasets: ISIC 2018 [43], BreakHis [33], and Pap Smear [18].

Another way of tackling the problem of a limited number of labeled data is data augmentation - a technique of creating additional artificial training data by modifying existing data. There are numerous algorithms for data augmentation, but in this work, in line with MetaMed, Cutout [9], Mixup [42], and Cutmix [41] augmentation techniques were tested.

The main goal of this thesis is to explore the performance of ViTs in a few-shot learning scenario for MIC and compare it with traditional CNNs. Therefore, in this thesis, we will use some prominent ViT models for few-shot classification of medical imaging using FSL algorithms such as Prototypical Networks [31] and Reptile [25] and compare them with similarly-sized CNN models. Additionally, this work investigates the effects of advanced augmentation techniques, such as Cutout, Mixup, and Cutmix, on the performance of ViT for FSL.

This thesis work will answer the following research questions:

1. How effective are Vision Transformers for medical image classification under few-shot learning restrictions?

2. How will the performance of Vision Transformers used with different few-shot learning algorithms compare?

3. Will the effects of advanced data augmentation techniques be as noticeable for Vision Transformers as for smaller models used in few-shot learning?

The remainder of this thesis is organized as follows. In Chapter 2, we provide an overview of related works in few-shot learning with ViT, medical image classification using ViTs, and FSL. In Chapter 3, we present the overview of prominent Vision Transformer architectures, describe the original ViT architecture and its key components, and explain the concept of meta-learning and different approaches to it. In Chapter 4, we describe the methodology, including FSL algorithms, augmentation techniques, and the overall pipeline. In Chapter 5, we present descriptions of datasets, experimental setup, implementation details, and the results of our experiments, comparing the performance of ViTs with traditional CNNs and analyzing the impact of various factors. Finally, in Chapter 6, we draw conclusions and discuss future directions for research on medical image classification using ViTs in a few-shot learning scenario.

Chapter 2

Related works

This chapter provides an overview of the literature on few-shot learning with Vision Transformers (ViTs), medical image classification using Transformers, and the application of few-shot learning in medical image classification.

2.1 Few-shot Learning with ViT

This section reviews selected research papers on few-shot learning that use the ViT architecture. The general information about the state-of-the-art in FSL can be acquired from the following survey papers [32, 15, 39].

There is a limited number of papers that have used ViTs in a few-shot learning scenario. One example is a paper by Hu et al. [16], in which the authors investigated how a simple FSL pipeline compares with complex state-of-the-art FSL algorithms. This work considers two backbone models: ViT small and ResNet50. The pipeline consists of three stages: 1) self-supervised backbone pre-training using DINO [4], 2) meta-training on labeled few-shot tasks using ProtoNet [31], and 3) fine-tuning on the augmented support set of each task. The results show that this pipeline with the transformer backbone outperforms the state-of-the-art and also the pipeline with the CNN backbone.

A paper by Chen et al. [6] takes another direction and proposes an architecture that utilizes masking of irrelevant parts of images to perform few-shot learning. They reused a vanilla ViT and added a masking operation before the first transformer layer. The results showed that the proposed method outperformed a vanilla ViT across all tests.

These papers demonstrate the effectiveness of Vision Transformers in a few-shot learning scenario. This research follows the approach of the former work in the desire to utilize a simple pipeline based on ViT and applies it to a few-shot medical image classification task.

2.2 Medical Image Classification

Accurate classification of medical images can greatly aid in the accurate diagnosis of human diseases. The first part of this section covers works that have used ViT for the task of medical image classification, while the second part covers works on Few-Shot Learning (FSL) for medical image classification. However, upon conducting a literature search, no works were found that have used ViTs for medical image classification in a FSL scenario.

2.2.1 Medical Image classification and Vision Transformers

Krishnan and Krishnan used several off-the-shelf pre-trained models, both CNN and ViT, and fine-tuned them on chosen datasets for a new task [20]. Experimental results showed that ViT achieved the highest accuracy among the tested models.

Perera, Adhikari, and Yilmaz proposed a lightweight transformer architecture called POCFormer for detecting COVID-19 on portable devices [28]. In order to decrease the complexity of their architecture, they used a linear transformer model Linformer [38] which has linear complexity in contrast to ViT's quadratic complexity. Experiments on the POCUS dataset showed an overall accuracy of 93.9%, which was similar to or better than other networks with more parameters.

Liu and Yin presented a novel transformer architecture called VOLO with a new attention mechanism called outlooker attention for COVID-19 classification [21]. The authors used transfer learning with a pre-trained VOLO model on ImageNet-1K and fine-tuned it for the COVID-19 classification task, achieving 99.7% accuracy on their target dataset and outperforming the CNN state-of-the-art.

Duong et al. tried to combine CNN and ViT for the task of detection Tuberculosis in Chest X-ray images [11]. One of the aims of this paper was show good performance on a larger and diverse dataset in contrast to other paper of that time that mainly used smaller and not diverse datasets. In the proposed approach, EfficientNet learns feature maps from the input and its outputs are flattened and combined with positional encoding to be fed into a transformer encoder-decoder with additional layers for classification. Authors reported the maximum accuracy of 97.72% for one of their models.

Similarly, a proposed architecture by Park et al. uses a backbone model to extract low-level features that are later fed into a Vision Transformer [26]. By doing this, authors wanted to imitate how a human clinical expert looks at Chest X-rays and makes decision on a diagnose. The proposed model showed better performance when compared to the similar models for COVID-19, and other CNN and Transformer-based models.

Around the same time, Jiang and Lin presented their version of a solution by using ensemble learning which combines Swin-transformer and Transformer in Transformer (TNT) [19]. This approach uses both aforementioned models to retrieve their outputs, then performs weighted averaging which in turn gets fed into a linear classifier. Authors claim the accuracy result of 94.75% on the target dataset.

More recently, Behrendt et al. conducted a systematic comparison of ViTs and CNNs for multi-label medical image classification, and evaluated the performance of DeiT [2]. Their experiments showed that all models benefited from larger dataset sizes, and DeiT, which used DenseNet-201 as a teacher model for knowledge distillation, outperformed other models across all dataset sizes.

These papers demonstrate that ViTs can be successfully applied to the task of medical image classification, often outperforming CNNs.

2.2.2 Medical Image classification and FSL

Singh et al. [30] presented their solution to the problem of FSL and medical image classification called "MetaMed." In this study, the authors address the challenges posed by long-tailed distributions and the scarcity of high-quality annotated images in medical datasets by formulating a few-shot learning problem and presenting a meta-learning-based approach. The work utilizes the Reptile meta-learning algorithm and a simple CNN. The model was validated on three publicly available medical datasets: Pap smear, BreakHis, and ISIC 2018. To combat overfitting, advanced image augmentation techniques, such as Cutout, Mixup, and Cutmix, are employed. The proposed approach demonstrates promising results, achieving over 70% accuracy on all three datasets. The inclusion of advanced augmentation techniques improves the model's generalization capability by 2-5%. Furthermore, a comparative analysis showed that MetaMed consistently outperforms transfer learning for 3, 5, and 10-shot tasks in both 2-way and 3-way classification scenarios.

In another paper, Dai et al. proposed PFEMed, a novel few-shot classification method for medical images [8]. PFEMed employs a dual-encoder structure using one encoder with fixed weights pre-trained on public image classification datasets and another encoder trained on the target medical dataset. A prior-guided Variational Autoencoder module is introduced to enhance the target feature, which is the concatenation of general and specific features. The method matches target features extracted from support and query medical image samples to predict category attribution. Experiments on several publicly available medical image datasets show that PFEMed outperforms current state-of-the-art few-shot methods, surpassing MetaMed on the Pap smear dataset by over 2.63%. The authors demonstrate the effectiveness of using knowledge from publicly available datasets to solve few-shot classification problems in the medical field.

An article by Cherti and Jitsev [7] investigates the effect of pre-training scale in both in-domain and out-of-domain transfer settings using both natural image and medical X-Ray chest imaging datasets. Results showed that both intra- and inter-domain transfer benefited from larger pre-training scales, though the effects differed between scenarios and for full shot and few-shot regimes. It was discovered that large networks pre-trained on the very large generic natural ImageNet-21k performed as well or better than networks pre-trained on the largest available medical domain-specific X-Ray super-set data when transferring to large X-Ray targets. However, this effect was noticed in a full-shot scenario. For few-shot scenarios, pre-training on a medical image dataset showed better results when transferring to another medical dataset.

Chapter 3

Preliminaries

This chapter serves as an introduction to some of the key concepts and algorithms that are essential for understanding the thesis. It begins by providing an overview of Transformers and how the Vision Transformer architecture works, followed by an explanation of the concept of Meta Learning. Finally, the chapter delves into the details of two types of Meta learning algorithms, namely Initialization-based and Distance-based methods.

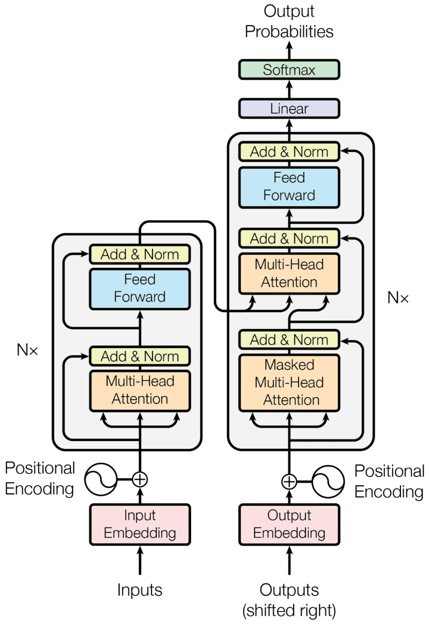

Figure 3-1: **Transformer architecture presented by Vaswani et al.** The image was acquired from the original paper [36].

3.1 Transformers

Transformer neural networks were first introduced in 2017 by Vaswani et al. [36] and relied solely on the attention mechanism. Initially, they were meant to be used in natural language processing (NLP). This model outdated other other models like RNN and CNN, and now it dominates the field of NLP. In 2020, Dosovitskiy et al. presented the former model's adaptation to a computer vision task, which they called "Vision Transformer" (ViT) [10] by representing input images as a sequence of patches. Despite the fact that this new architecture reuses the encoder block from the original transformer paper, this model showed excellent results comparable with or outperforming the state-of-the-art CNN models of that time. They used a proprietary dataset consisting of 300 million labeled images and stated that the ViT doesn't generalize well on smaller datasets. However, this is not practical for images, as all other available and known to me datasets are significantly smaller in size. To address this issue, Touvron et al. introduced their model named DeiT (Data efficient image Transformer) [35]. Their approach uses Teacher-Student knowledge distillation in which a student, transformer model, learns from a teacher, CNN model for example. This allows to train a transformer model with fewer samples. Around the same time, another interesting model, Swin-transformer by Liu et al. appeared, aiming to solve another ViT issue, scale and complexity [22]. In the original paper, ViT uses 16x16 patches, however this may be too large for some tasks. Swin-transformer's first layer has a patch size of 4x4 which is can be beneficial for some tasks of medical imaging where small pixel-level details are of big importance. Also, original approach has quadratic complexity with respect to input size, while Swin-transformer has only linear complexity due to leveraging shifted window approach in which self-attention is computed only inside a window. This can be important in medical imaging where images may have high resolution and down-scaling may remove crucial information. This paper experiments with the above-mentioned models to solve the few-shot learning problem of medical image classification.

3.2 Vision Transformer

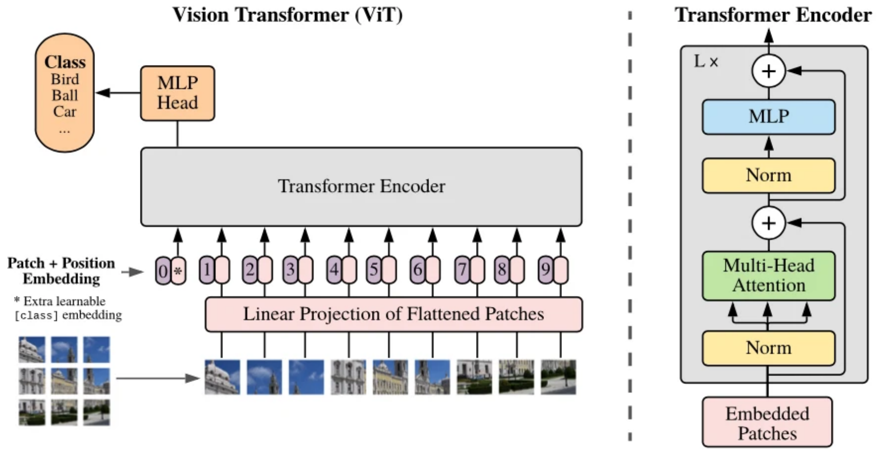

Figure 3-2: **Architecture of a Vision Transformer.** The image was acquired from the original paper [10].

As it was mentioned above, The Vision Transformer is a neural network architecture for computer vision tasks, introduced by Dosovitskiy et al. in their 2020 paper. The ViT architecture adapts the Transformer model (Figure 3-1), initially proposed by Vaswani et al. By doing so, ViT demonstrates competitive performance with state-of-the-art CNNs while offering a more scalable and flexible architecture. The overview of the model is presented in Figure 3-2.

The Vision Transformer architecture processes an input image by dividing it into non-overlapping patches and linearly embedding them into a sequence of flat vectors, which serve as input tokens for the Transformer model. The key components of the ViT architecture are:

1. **Image Patching:** The input image is divided into a fixed number of non-overlapping patches of equal size. For example, if an input image is of size 224×224 and the patch size is 16×16, the image is divided into 196 patches.

2. **Patch Embedding:** Each image patch is then reshaped into a 1D vector and linearly embedded using a fully connected (FC) layer, resulting in a sequence of patch embeddings. The dimensionality of these embeddings is referred to as the hidden size d.

3. **Position Embeddings:** To incorporate positional information, position embeddings are added to the patch embeddings. These position embeddings are learnable parameters initialized randomly and optimized during training.

4. **Transformer Layers:** The sequence of patch embeddings, along with the added positional information, serves as the input for a stack of Transformer layers which is just the encoder block of the model from the original paper. These layers consist of multi-head self-attention (MSA) and multi-layer perceptron (MLP) blocks, connected through residual connections and layer normalization. The self-attention mechanism allows the model to capture long-range dependencies and global context within the image.

Multi-Head Self-Attention

The multi-head self-attention mechanism is a key component of the Transformer architecture. It operates by computing attention scores for all pairs of input tokens and combines them using a weighted average, allowing the model to capture dependencies between tokens regardless of their positions in the sequence. It uses Scaled Dot-Product Attention which operates by calculating attention scores between pairs of input tokens, which are then used to compute a weighted average of the input embeddings. For each input token, query (Q), key (K), and value (V) matrices are computed by multiplying the input embeddings with learnable weight matrices. It is calculated as the following:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{3.1}$$

where d_k is the dimensionality of K. By employing multiple attention heads, the model can concurrently attend to different features or semantic aspects of the input tokens. Each attention head computes its own set of Q, K, and V matrices, and performs the scaled dot-product attention mechanism independently. The attended embeddings from all heads are then concatenated and projected back to the original hidden size using a linear layer.

Classification

For image classification tasks, the Vision Transformer uses a dedicated token for aggregating the global context information and producing the final output prediction. The classification token is prepended to the sequence of patch embeddings that represent the input image, allowing the model to process it alongside the other image patches and incorporate global context information.

3.3 Meta Learning

Meta learning, also known as "learning to learn," is a machine learning approach that aims to improve a model's ability to learn new tasks quickly and efficiently with minimal data. In contrast to traditional machine learning, where models are trained to perform a specific task with a large amount of data, meta learning focuses on enabling models to adapt to new tasks or variations of tasks using prior knowledge and experience. The idea of meta learning is to generalize learning strategies across a wide range of tasks. This is achieved by extracting common patterns and structures from a distribution of tasks, which can then be used to facilitate adaptation to previously unseen tasks. Meta learning aims to build models that are adapted to problems data scarcity and task diversity in real-world scenarios. Therefore, meta learning has been successfully applied to few-shot learning.

There are various approaches and algorithms for implementing meta learning, broadly categorized into initialization-based, metric-based, and memory-based approaches. These methods aim to solve different aspects of the meta learning problem, such as learning shared representations, discovering useful model initializations, or learning effective memory management techniques. In this paper, Reptile and Prototypical Networks (ProtoNet), one initialization-based and one metric-based algorithm, respectively, are used. Therefore, information on these two types of meta learning is provided below. The algorithms themselves will be discussed in the methodology chapter.

3.3.1 Initialization-based methods

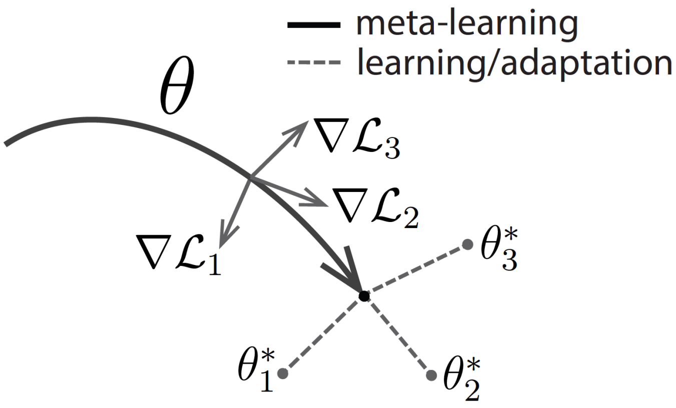

Figure 3-3: **The diagram of MAML from the original paper.**

Initialization-based methods are a category of meta-learning techniques that focus on finding an optimal initial set of model parameters. These parameters can be rapidly fine-tuned for a variety of new tasks with minimal data and training iterations. During the meta-training phase, the model is exposed to a wide range of tasks from a task distribution. The model learns to generalize across these tasks by optimizing the initial parameters, ensuring that they can be easily fine-tuned for new tasks encountered during the meta-testing phase. The following paragraphs describe the Model-Agnostic Meta-Learning (MAML) algorithm, as it is related to the Reptile algorithm.

Model-Agnostic Meta-Learning is a meta-learning algorithm proposed by Finn et al. [13] 2017. The MAML algorithm is designed to learn an optimal initialization of model parameters that facilitates rapid adaptation to new tasks using minimal data and training iterations. MAML is model-agnostic, as it can be applied to any model trained with gradient-based learning, encompassing a wide range of neural network architectures. Therefore, it can be used with both CNNs and ViTs.

The MAML algorithm comprises two nested loops: the outer loop for meta-training and the inner loop for task-specific training. During the meta-training phase, the model is exposed to a diverse set of tasks sampled from a task distribution. The primary goal is to learn a suitable initialization of model parameters (θ) that can be quickly fine-tuned for new tasks. The diagram how MAML works is depicted on Figure 3-3. The pseudo-code of MAML is presented below in Algorithm 1.

Algorithm 1 Model-Agnostic Meta-Learning (MAML)

```
Require: $p(\mathcal{T})$, distribution over tasks
Require: $\alpha, \beta$, step size hyperparameters
1: Randomly initialize $\theta$, model parameters
2: while not done do
3:     Sample a batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4:     for all $\mathcal{T}i$ do
5:         Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ respect to K examples
6:         Compute adapted parameters $\theta_i^{'} = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
7:     Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
```

Basically, this algorithm works as follow:

1. **Outer loop (meta-training) (line 2):** The outer loop of MAML focuses on learning the optimal initialization of model parameters (θ) by minimizing the expected loss across tasks after the inner loop updates. This is achieved by performing gradient-based updates on the initial model parameters (θ) with respect to the task-specific parameters (θ_i') obtained after inner loop updates.

2. **Sample tasks (line 3):** During the meta-training phase, the model is exposed to a diverse set of tasks sampled from a task distribution $p(\mathcal{T})$.

3. **Inner loop (task-specific training) (line 4):** For each task T_i in the meta-training set, the model undergoes task-specific training in the inner loop. This involves fine-tuning the model parameters (θ) on a small dataset (support set) specific to task T_i. The fine-tuning process generally involves the following steps:

    (a) **Compute gradients (line 5):** For each example in the support set, compute the gradients of the task-specific loss function with respect to the task-specific parameters (θ_i') by calculating the loss for the given example and backpropagating the error through the model.

(b) **Update task-specific parameters (line 6):** Perform a few gradient update steps (usually 1-5 steps) on the task-specific parameters (θ_i') using the computed gradients and a task-specific learning rate α.

4. **Meta-objective and parameter updates (line 7):** MAML aims to minimize the expected loss across tasks after the inner loop updates. This is achieved by updating the initial model parameters (θ) using the sum of the gradients of the task-specific losses concerning the task-specific parameters (θ_i'):

    (a) **Compute task-specific loss:** Evaluate the updated task-specific parameters (θ_i') on a separate dataset (query set) to compute the task-specific loss.

    (b) **Accumulate gradients:** Compute the gradients of the task-specific loss with respect to the task-specific parameters (θ_i') and accumulate them.

    (c) **Update initial model parameters:** After processing all tasks in the batch, update the initial model parameters (θ) using the accumulated gradients and a meta-learning rate β.

By iteratively performing these steps during the meta-training phase, MAML learns an optimal initialization of model parameters (θ) that allows for rapid adaptation to new tasks with minimal data and training iterations. Once the meta-training phase is complete, the learned initial parameters can be quickly adapted to new tasks during the meta-testing phase with only a few gradient updates and a small amount of data.

3.3.2 Metric-based methods

Metric-based methods are a category of meta-learning techniques that focus on learning a similarity metric or acquiring a good feature space, aiming to improve a model's ability to quickly and efficiently adapt to new tasks using minimal data. In contrast to other meta-learning approaches, such as initialization-based methods, metric-based methods do not rely on fine-tuning model parameters for new tasks. Instead, they learn a representation that enables effective comparisons between samples, facilitating few-shot learning by identifying similarities between instances in the context of the target task. More in-depth overview of this subject can be found in survey papers [15, 17] Description of some prominent metric-based methods is presented below:

- **Siamese Networks:** Siamese networks consist of two parallel neural networks that share weights and learn to differentiate between pairs of input data points, primarily used for tasks like one-shot learning and few-shot learning.

- **Matching Networks**: Matching Networks, introduced by Vinyals et al. [37], are designed to address few-shot learning problems by learning an attention mechanism over a support set. The support set consists of a small number of labeled examples from each class in the new task. Given a query data point, the model computes the similarity between the query and each example in the support set. The model then generates a weighted sum of the support set labels, with the weights determined by the attention mechanism. This process allows the model to make predictions for the query data point based on the most relevant examples in the support set.

- **Prototypical Networks:** Prototypical Networks, proposed by Snell et al. [31], learn class prototypes by computing the mean embedding of the data points belonging to each class within a support set. The embedding function is typically implemented as a neural network, and the mean embeddings represent the class prototypes in the embedding space. Given a query data point, the model assigns the data point to the class with the nearest prototype, as determined by a distance metric such as Euclidean distance.

- **Relation Networks**: Relation Networks, introduced by Sung et al. [34], combine the ideas of embedding learning and metric learning to address few-shot learning problems. The model consists of two sub-networks: a feature extractor and a relation module. The feature extractor generates embeddings for both the support set and query data points, while the relation module computes the pairwise relations between the query data point and the support set examples. The model then classifies the query data point based on the highest relation score. Relation Networks learn to compare data points effectively, allowing them to perform well in few-shot learning tasks.

In this paper, Prototypical Networks and Reptile were used in conjunction with CNNs and ViTs as few-shot learners. A more detailed explanation of these algorithms is presented in the Methodology chapter.

Chapter 4

Methodology

This chapter formulates the problem definition of few-shot medical image classification, presents the overall pipeline of the system, and describes the methodology.

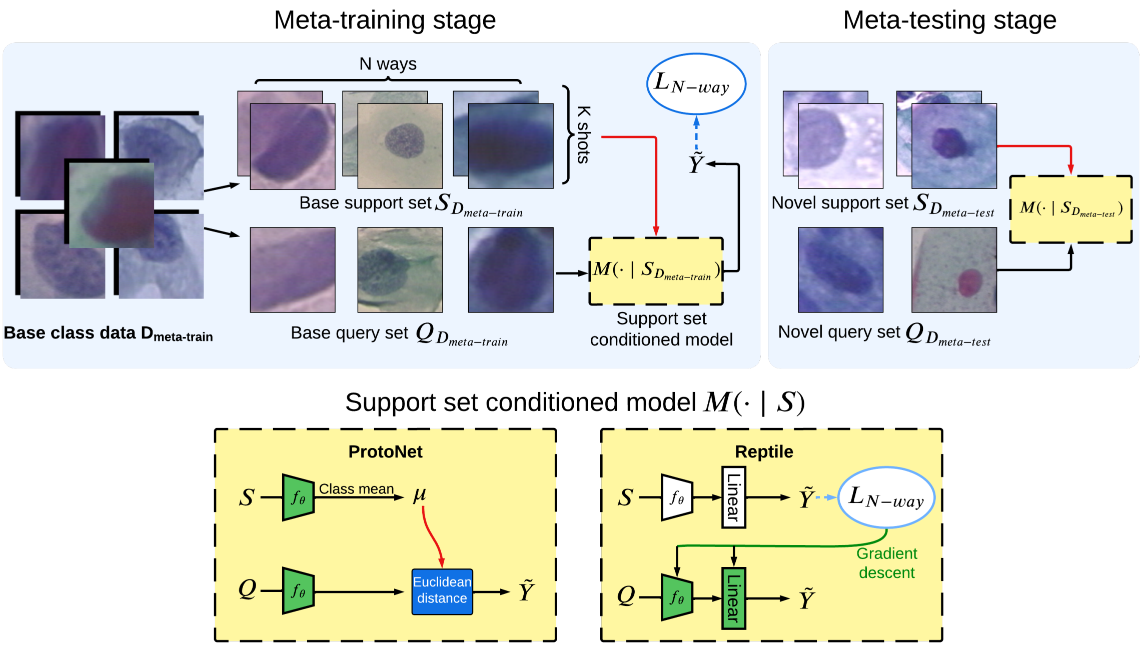
Meta-training stage
Meta-testing stage
N ways
K shots
L_{N-way}
$\tilde{Y}$
Base class data $D_{meta\text{-}train}$
Base support set $S_{D_{meta-train}}$
Base query set $Q_{D_{meta-train}}$
$M(\cdot \mid S_{D_{meta-train}})$
Support set conditioned model
Novel support set $S_{D_{meta-test}}$
Novel query set $Q_{D_{meta-test}}$
$M(\cdot \mid S_{D_{meta-test}})$
Support set conditioned model $M(\cdot \mid S)$
ProtoNet
S
f_θ
Class mean
μ
Q
f_θ
Euclidean distance
$\tilde{Y}$
Reptile
S
f_θ
Linear
$\tilde{Y}$
L_{N-way}
Gradient descent
Q
f_θ
Linear
$\tilde{Y}$

Figure 4-1: **Overall System Pipeline.**

4.1 Problem Definition

The problem definition is inline with the one presented in MetaMed [30]. Let $D = \{D_1, D_2, ..., D_n\}$ be a collection of n medical datasets, with each dataset D_k consisting of pairs $(x, y)_j$, where $(x, y)_j$ represents an image and its corresponding label (ground-truth). Each dataset is divided into a meta-test set ($D_{meta-test}$), which includes images of classes with fewer representative images (rare diseases), and a meta-train set ($D_{meta-train}$), which contains the remaining classes. The main idea is to utilize the abundant data available in $D_{meta-train}$ (base class data) to learn better initial weights when Reptile is used and then fine-tune the model on problems with limited data (novel class data). In the case of ProtoNet, the goal is to develop a model that can produce an effective embedding space, where the feature representation of a sample is close to its corresponding prototype and distant from prototypes of other classes. This facilitates the identification of similar items with ease. The overall pipeline of the system is presented in Figure 4-1, and in its context, both ProtoNet and Reptile are support set conditioned models. The bottom part of the figure shows the architectures of these models.

4.2 Few-shot Learning

The problem of few-shot learning is concerned with developing machine learning models that can generalize effectively to new tasks, given only a limited number of labeled examples from each class in the target domain. Generally, the difficulty of tasks in few-shot learning can be described as N-way-K-shot, where N represents the number of classes and K represents the number of samples from each class used for training. There are various approaches to few-shot learning, one of which is the meta-learning perspective. In this approach, the model learns to solve new few-shot tasks by drawing on the experience of solving other few-shot tasks, which is divided into meta-training and meta-testing phases. In each phase, data is presented episodically, with the support set serving as the training set and the query set as the test set. Transfer learning can also be considered as a few-shot learning approach, where the model is pre-trained on a large dataset and then fine-tuned on the limited support set. However, this approach is less effective when there is a large domain gap between the source and target datasets. Data augmentation is another technique for tackling few-shot learning, in which new samples are generated by augmenting the samples from a limited support set. This chapter describes two algorithms used in this thesis, namely Reptile and ProtoNet, in detail.

4.2.1 Reptile

Reptile [25] is a meta-learning algorithm designed for few-shot learning. It works by iteratively updating the model's weights through a two-level process: inner loop updates and outer loop updates. The inner loop focuses on learning from individual tasks, while the outer loop learns across tasks. Figure 3-3 can also be referenced in the context of Reptile. Pseudo-code is presented in Algorithm 2. Reptile's simplicity allows for faster training and easier implementation compared to MAML as it does not require the computation of second-order gradients. Cross entropy loss was used to update the weights of the model in both meta-training and meta-testing phases.For a task T_i, it is given by

$$\mathcal{L}_{T_i}(f_\phi) = - \sum_{x_i, y_i \sim T_i} y_i log(f_\phi(x_i) + (1 - y_i) log(1 - f_\phi(x_i)) \tag{4.1}$$

where (x_i, y_i) is a image and label pair.

Algorithm 2 Reptile

```
Initialize model weights $\boldsymbol{\theta}$
for iteration $= 1, 2, \dots, N$ do
    Sample task $T_i$ from task distribution $p(T)$
    Perform $K$ steps of SGD on $T_i$ to obtain updated weights $\boldsymbol{\theta}'$
    Update $\boldsymbol{\theta}$ using $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \epsilon(\boldsymbol{\theta}' - \boldsymbol{\theta})$
```

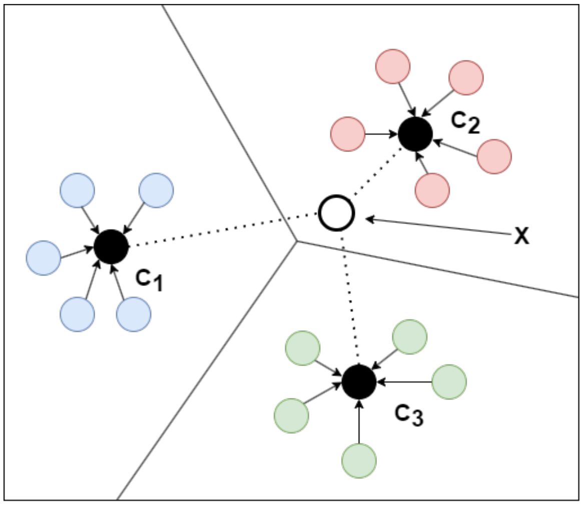

Figure 4-2: **Few-shot classification using class prototypes in ProtoNet algorithm.** Here, a sample x has been classified as class c_2, as its feature representation is closer to the class prototype of c_2.

4.2.2 Prototypical Networks

Prototypical Networks [31] aim to learn a prototype for each class in the embedding space. Given a set of support samples and their corresponding labels, the model learns an embedding function, $\phi : \mathcal{X} \to \mathbb{R}^d$, which maps input images to a d-dimensional space.

The prototype c_k for class k is computed as the mean of the embedded support samples belonging to that class:

$$c_k = \frac{1}{N_k} \sum_{(x_i, y_i) \in S, y_i = k} \phi(x_i) \tag{4.2}$$

Here, S represents the support set, and N_k is the number of support samples in class k.

For a given query sample x, the model computes its embedding $\phi(x)$ and determines the class by finding the prototype with the smallest Euclidean distance:

$$\hat{y} = \arg\min_k \|\phi(x) - c_k\|^2 \tag{4.3}$$

The classification procedure is demonstrated in Figure 4-2.

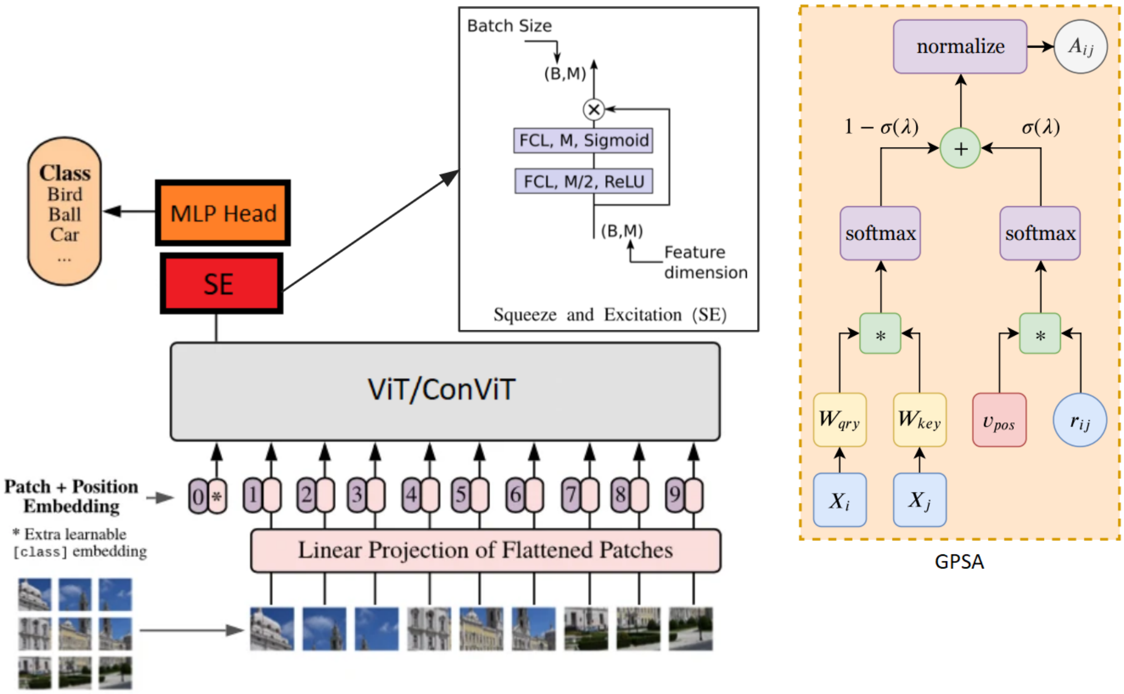

Figure 4-3: **The architecture of our custom ViT.** Image was adapted from [12] and [1].

4.3 Custom ViT

Aouayeb et al. [1] adapted Squeeze and Excitation (SE) block from CNNs to ViTs and achieved improved results on the Facial Expression Recognition task when compared to vanilla ViTs. When used with ViTs, there is no need in squeeze operation, therefore SE block serves as an additional attention mechanism on top of a ViT encoder for learning more global features from a class token. In this thesis, we propose to use the SE block with ConViT architecture [12]. Proposed by d'Ascoli et al., this architecture differs from the original ViT by introducing a new attention mechanism called gated positional self-attention (GPSA) as an attempt to introduce convolution-like locality bias. This bias should allow the model to train using less data in comparison with the original ViT. The architecture of ViTs, SE block, and GPSA are depicted in Figure 4-3.

4.4 Advanced Augmentation techniques

Generally, ViTs rely on various regularization techniques including data augmentation in order to achieve high performance on small datasets. This may be even more important in FSL, as the amount of data is smaller. Augmentation techniques encourage models to learn more generalized representations, as it has to predict the correct label proportions based on the augmented inputs, ultimately reducing overfitting. Such techniques were used in MetaMed paper, where it improved accuracy of FSL significantly. However, as we are using much bigger models, it is important to see if the results would be as promising. It should be stated that only the Cutout technique is compatible with ProtoNet algorithm, as the other 2 methods modify labels of the support set.

4.4.1 Cutout

Cutout is a data augmentation technique that randomly removes rectangular regions from input images during training. Given an input image x with dimensions $H \times W$, Cutout selects a rectangular region with dimensions $h \times w$, where $h \leq H$ and $w \leq W$. The center (c_x, c_y) of the selected region is chosen uniformly at random, ensuring that the rectangle lies within the image boundaries. The pixel values within this rectangular region are then set to a predefined constant value (e.g., zero).

4.4.2 Mixup

Mixup is a method that involves generating new training examples by taking a linear combination of two randomly chosen input images and their corresponding labels. Specifically, given two images x_1 and x_2 with their respective labels y_1 and y_2, and a mixing coefficient λ sampled from a Beta distribution, the new mixed image x_{mixup} and its label y_{mixup} are obtained by computing $x_{\text{mixup}} = \lambda x_1 + (1 - \lambda) x_2$ and $y_{\text{mixup}} = \lambda y_1 + (1 - \lambda) y_2$.

4.4.3 Cutmix

Cutmix is a method that combines the strengths of both Cutout and Mixup. The idea behind Cutmix is to replace a portion of an input image x_1 with another image x_2, while also adjusting the corresponding labels accordingly. To achieve this, a random bounding box is selected within the original image x_1, and its content is replaced with the corresponding region from the second image x_2. The new mixed image x_{cutmix} is then formed, and its label y_{cutmix} is computed as $y_{\text{cutmix}} = \lambda y_1 + (1 - \lambda) y_2$, where λ is the proportion of the area of the replaced region in relation to the entire image.

Chapter 5

Results

5.1 Dataset Description

Three publicly available medical imaging datasets were selected for this research. Each datasets contains at least six classes such that both 2- and 3-way n-shot learning can be performed. In contrast with other works, images were downsampled to 224x224 (standard is 84x84) in order to utilize pre-trained models.

5.1.1 BreakHis

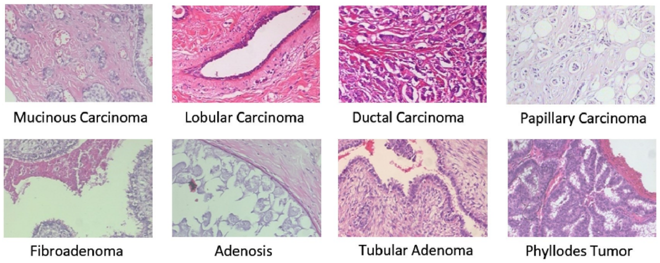

Figure 5-1: **BreakHis Dataset.**

The BreakHis dataset consists of 9109 microscopic images of breast tumor tissues, collected from 82 patients and captured at magnification levels of 40, 100, 200, and 400. Each image has a height of 700 and a width of 460. This dataset is divided into eight classes. Five classes with the most samples were selected as meta-train

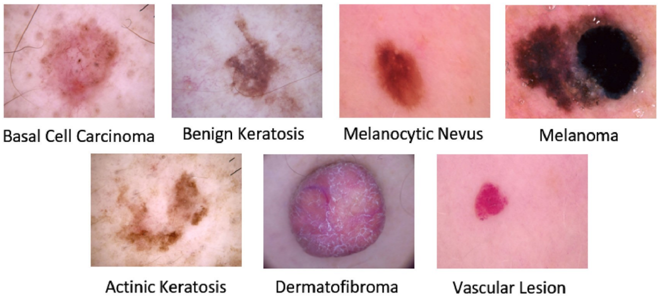

Figure 5-2: **ISIC 2018 Dataset.**

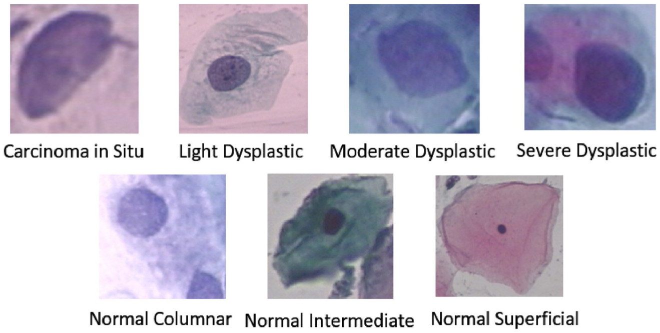

Figure 5-3: **Pap smear Dataset.**

classes, while the rest as meta-test classes. Fig. 5-1 shows examples of images from the dataset representing all classes.

5.1.2 ISIC 2018

The ISIC 2018 Skin Lesion dataset comprises a total of 10,015 dermoscopic images spanning seven classes. The dataset's distribution of diseases reflects real-world prevalence, with a higher number of images for benign lesions compared to malignant ones. The images have dimensions of 600 pixels in height and 450 pixels in width. Similarly, four classes with the most samples were selected as meta-train classes. The remaining three classes are designated for meta-testing. Fig. 5-2 shows examples of images from the dataset representing all classes.

5.1.3 Pap Smear

The benchmark dataset for Pap-smear consists of microscopic images of cervical smears taken at Herlev University Hospital. The dataset contains a total of 917 images, unevenly distributed across seven distinct classes, which were annotated by experienced cyto-technicians and doctors. Fig. 5(a) and (b) display the representative images and class distribution, respectively. Four classes with the most samples were selected as meta-train classes, while the remaining three classes are selected for meta-testing. Fig. 5-3 shows examples of images from the dataset representing all classes.

Table 5.1: Models used in this thesis.

Model	Dim	Parameters
ViT_tiny [10]	192	5.5m
MViT_v2_0.5 [24]	384	1.4m
ViT_small [10]	384	22m
ViT_base [10]	768	85m
DeIT_base [35]	768	85m
Swin_base [22]	1024	86m
ResNet50 [14]	2048	23.5m
VGG16 [29]	4096	134m

5.2 Models

This section discusses various models used for testing in the thesis. Table 5.1 summarizes the feature dimensionality and the number of parameters for each model. The models tested can be grouped into three categories:

- ViT family: Three models from the Vision Transformer (ViT) family [10] were used, including ViT_tiny, ViT_small, and ViT_base. These models differ in the number of parameters and feature dimensionality, with ViT_tiny having the least parameters (5.5 million) and ViT_base having the most (85 million).

- Other Vis architectures: Mobile_ViT (MViT_v2_0.5) [24], DeiT_base [35], and Swin_base [22] models were selected to investigate the performance of alternative ViT architectures. MViT_v2_0.5 has fewer parameters (1.4 million) but a higher feature dimensionality (384) than ViT_tiny. DeiT_base and Swin_base have similar feature dimensionality (768) and number of parameters (85 million and 86 million, respectively) as ViT_base.

- CNN models: ResNet50 [14] and VGG16 [29] are two well-known Convolutional Neural Network (CNN) models used for comparison with the ViT models. ResNet50 has 23.5 million parameters and a feature dimensionality of 2048, while VGG16 has 134 million parameters and a feature dimensionality of 4096.

All models were pre-trained on the ImageNet1k dataset, which is a widely used dataset for training computer vision models.

5.3 Implementation Details

The overall implementation was done in Python using the PyTorch framework [27]. Pre-trained models were obtained from the timm (PyTorch Image Models) library [40]. The ProtoNet experiments were conducted using the easyfsl library [3]

5.4 Experimental Settings

5.4.1 Setup

The hardware specifications for the PC and Google Colab Pro Platform used for the experiments were as follows:

PC:

- NVIDIA RTX 3060 Ti with 8GB of VRAM
- Intel i5-10400 CPU
- 16GB of RAM

Google Colab Pro Platform:

- NVIDIA Tesla T4 with 16GB of VRAM or A100 with 40GB of VRAM

The models were trained and tested on the PC, while some experiments were also performed on the Google Colab Pro Platform for additional computational resources.

5.4.2 Training

We utilized pre-trained model checkpoints during the training phase and employed data augmentation techniques. We believe that the effect of class imbalance is mitigated by the nature of episodic task sampling in FSL, as few-shot learners see an equal amount samples from each class.

For ProtoNet, we trained the model for 20 epochs and found that further training epochs caused the model to overfit. Each epoch included 500 episodes or tasks. Stochastic gradient descent (SGD) optimizer was used with a learning rate of 10^{-5} or 10^{-6} on the model, with a momentum of 0.9, depending on the dataset. We also utilized a cosine annealing learning rate schedule.

For Reptile, we used the SGD optimizer with a learning rate of 10^{-3} for the inner optimization problem and SGD with a learning rate (step size) of 10^{-1} for the outer meta-update step. During meta-training, a backbone was trained for 1000 meta-iterations with a batch size of 10 tasks. In both training and testing, batch size was 10 tasks per meta-iteration. For the inner problem, we experimented with 5 and 50 adaptation steps for each task.

5.4.3 Evaluation

In line with MetaMed [30], we utilized accuracy (%) as the evaluation metric in our experiments, a common performance indicator for few-shot classification tasks. Similar to the previous section, class imbalance does not impact accuracy scores in this context because task sampling ensures an equal number of query samples from each class are selected from the meta-test dataset. To assess performance on the BreakHis, ISIC 2018, and Pap smear datasets, we randomly select 400 episodes from the novel categories in the test set each time and compute the average accuracy rate for image classification. We tested 2- and 3-way 2-, 5-, and 10-shot few shot learning scenarios.

5.5 Analysis

5.5.1 Pretrained ViT without Meta-training

Table 5.2: Few-shot classification results without meta-training for the ISIC 2018 dataset.

Model	2-way			3-way		
	3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
ViT_tiny	70.25	74.60	76.15	54.83	59.51	65.41
MViT_v2_0.5	59.30	63.10	67.80	46.13	48.27	49.90
ViT_small	77.40	81.89	85.95	63.67	69.84	75.28
ViT_base	74.75	77.70	82.45	60.73	65.73	69.97
DeIT_base	71.75	79.40	81.75	58.33	61.87	69.47
Swin_base	75.10	80.15	82.00	62.27	67.67	71.50
ResNet50	72.66	76.17	79.15	56.69	62.31	65.81
VGG16	72.45	78.60	81.30	60.00	65.87	68.20

Table 5.3: Few-shot classification results without meta-training for the BreakHis dataset.

Model	2-way			3-way		
	3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
ViT_tiny	71.25	77.35	78.50	57.27	61.53	67.87
ViT_small	74.71	79.42	83.24	63.22	69.25	73.91
ViT_base	74.50	80.70	84.90	63.90	69.17	75.50
Swin_base	77.95	83.20	85.37	72.77	80.30	82.3
ResNet50	79.62	83.31	85.72	68.75	73.09	77.61
VGG16	70.40	79.15	81.75	60.70	65.40	71.67

This section examines the results of models pre-trained on ImageNet1k for few-shot classification tasks without meta-training on ISIC 2018 and BreakHis x100 datasets. Model checkpoints pretrained on ImageNet1k are used directly as ProtoNet backbones without fine-tuning or performing meta-training. Tables 5.2-5.3 present the findings. These findings reflect the differentiability of feature representations of samples from various classes generated by models pre-trained solely on a natural dataset.

In the following chapter, these results are compared with those after meta-training to demonstrate the improvement or deterioration of feature spaces generated by the models. Generally, from the tables, we see that models with more parameters tend to show higher results. On the contrary, Mobile ViT (MViT_v2_0.5) with 1.4 million parameters has the lowest score, and ViT_tiny has the second lowest. In terms of comparison between ViT and CNN, both show comparable results in general. However, these results only serve as an initial baseline for ProtoNet and should not be used for judging the overall performance of models in few-shot learning.

5.5.2 Meta-Training Results

Table 5.4: Performance of models using different meta-learning algorithms for ISIC 2018 dataset.

Algorithm	Model	2-way			3-way		
		3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
Protonet	MViT_v2_0.5	74.64	76.94	81.50	60.60	64.23	69.23
	ViT_tiny	81.03	83.61	86.52	67.84	71.82	77.68
	ViT_small	84.35	86.70	89.72	72.10	76.18	81.45
	ViT_base	83.94	86.02	90.26	72.75	77.69	81.99
	Swin_base	82.49	84.17	89.12	70.75	74.67	79.92
	ResNet50	66.62	68.65	72.81	51.43	53.83	58.34
	VGG16	72.32	76.04	80.69	57.81	61.86	66.92
Protonet w/o Pre-traning	ViT_small	56.19	57.55	60.17	39.87	41.08	41.88
Reptile 5 steps	ViT_small	71.23	76.65	81.38	66.20	72.23	78.10
	ResNet50	59.50	62.80	65.78	42.62	43.22	44.13
Reptile 50 steps	ViT_small	76.05	80.3	85.55	67.5	73.15	76.27
	ResNet50	66.68	72.13	77.03	53.63	57.03	60.18

This section analyzes the test results of few-shot classification models using ProtoNet and Reptile meta-learning algorithms. The results are presented in Tables 5.4 - 5.6. By comparing these tables with those from the previous section, we can observe

Table 5.5: Performance of models using different meta-learning algorithms for BreakHis with X100 magnification dataset.

Algorithm	Model	2-way			3-way		
		3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
Protonet	MViT_v2_0.5	76.89	79.60	84.65	64.51	71.43	77.05
	ViT_tiny	75.34	79.44	83.53	62.64	69.88	75.18
	ViT_small	80.64	83.80	87.62	69.39	75.91	81.47
	ViT_base	79.33	81.65	84.62	68.52	73.27	76.38
	Swin_base	79.46	82.86	86.26	68.34	74.28	80.51
	ResNet50	68.62	72.12	73.31	55.80	60.28	61.88
	VGG16	67.06	69.70	74.74	52.89	57.94	61.15
Reptile 5 steps	ViT_small	66.90	74.20	81.80	47.37	57.17	68.47
	ResNet50	64.90	67.60	73.25	34.70	36.33	38.23
Reptile 50 steps	ViT_small	73.45	77.9	86.18	55.05	63.38	75.92
	ResNet50	72.15	76.63	80.33	60.33	63.45	68.47

Table 5.6: Performance of models using different meta-learning algorithms for Pap Smear dataset.

Algorithm	Model	2-way			3-way		
		3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
Protonet	MViT_v2_0.5	80.84	84.36	86.88	68.04	73.24	78.37
	ViT_tiny	84.65	86.96	88.86	74.33	77.92	81.17
	ViT_small	92.40	94.05	94.90	86.38	89.09	90.62
	ViT_base	92.05	93.26	93.94	85.21	88.48	89.47
	Swin_base	85.42	87.56	89.78	75.73	79.88	82.46
	ResNet50	70.49	71.75	69.61	57.74	58.48	59.60
	VGG16	87.95	90.11	91.45	78.21	81.81	84.32
Reptile 5 steps	ViT_small	83.35	87.05	91.96	72.52	81.13	87.94
	ResNet50	71.44	74.59	78.39	48.00	49.86	50.44
Reptile 50 steps	ViT_small	85.85	88.33	92.55	76.75	82.58	86.92
	ResNet50	86.60	90.38	90.85	65.73	67.75	73.83

that ViTs paired with ProtoNet demonstrated noticeable performance gains across all datasets and FSL tasks. Mobile ViT and ViT, being the smallest and second smallest models, showed correspondingly lower results when compared with larger ViTs. However, ViT_small demonstrated the highest results in most cases, often outperforming bigger models. When it comes to ProtoNet and CNN, it can be stated that both ResNet50 and VGG16 performed worse after meta-training; therefore, in comparisons, we would consider their pre-meta-train ProtoNet and Reptile scores. Similar behaviour of ResNet50 was demonstrated in a paper by Chen et al. [5], where ResNet50 scores were significantly lower after meta-training with ProtoNet.

Additionally, the importance of pretraining can be highlighted by comparing ProtoNet with and without pretraining ("ProtoNet w/o Pre-training" in Table 5.4). ProtoNet with a ViT_small backbone pretrained on ImageNet1k has accuracy scores up to 30% higher when meta-trained. This indicates that the model is learning a more discriminative feature representation space. This result coincides with the observations of [16] and highlights the importance of pre-training.

As for the Reptile algorithm, its performance is highly dependent on proper hyperparameter selection. All tables contain results with 5 and 50 inner adaptation steps. Both models see a noticeable performance increase when task-adapted for more steps. However, this increase depends on the dataset and model used. For example, for ViT_small, the average increase from 5 to 50 steps is 2.35% for ISIC 2018, 5.99% for BreakHis, and only 1.51% for the Pap smear dataset. For ResNet50, these increases are 11.44%, 17.73%, and 17.07% respectively. This may imply that ViTs can adapt to novel classes faster than CNNs (at least for ResNet50). In general, ViT_small outperforms ResNet50 in an overwhelming majority of tasks across datasets. Despite the increase, the performance of ViTs with Reptile is still lower when compared with ProtoNets. On the contrary, ResNet50 showed much better results with Reptile.

Considering the ease of use and training, generally better performance across datasets and FSL tasks, and lower complexity of the algorithm, ProtoNet with a ViT backbone seems to be a better option than a CNN or a ViT paired with Reptile.

5.5.3 Effect of Augmentations

Table 5.7: Effect of different Augmentation techniques on Few-shot classification for ISIC 2018 Dataset

Algo.	Model	FSL	2-way				3-way			
			Standart	CutOut	MixUp	CutMix	Standart	CutOut	MixUp	CutMix
PN	ViT_s	3 shot	84.35	81.73	-	-	72.10	70.55	-	-
		5 shot	86.70	85.89	-	-	76.18	76.23	-	-
		10 shot	89.72	89.22	-	-	81.45	81.13	-	-
	RN50	3 shot	66.62	65.52	-	-	51.43	49.32	-	-
		5 shot	68.65	68.75	-	-	53.83	53.81	-	-
		10 shot	72.81	72.18	-	-	58.34	57.74	-	-
Rept.	ViT_s	3 shot	76.05	75.30	77.50	74.85	67.50	64.87	66.20	67.40
		5 shot	80.30	80.35	79.40	77.75	73.15	69.97	71.33	72.57
		10 shot	85.55	83.95	85.75	85.65	77.37	76.53	77.87	79.63
	RN50	3 shot	70.28	68.73	70.75	70.10	54.47	55.70	55.00	53.70
		5 shot	75.78	73.60	74.15	74.60	58.22	59.90	60.65	58.92
		10 shot	78.83	76.58	78.03	77.95	61.58	64.67	64.95	63.62

Table 5.8: Comparison with MetaMed and PFEMed: ISIC 2018 dataset

Algorithm	Model	2-way			3-way		
		3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
ProtoNet	ViT_small	**84.35**	**86.70**	**89.72**	**72.10**	**76.18**	**81.45**
	ResNet50	66.62	68.65	72.81	51.43	53.83	58.34
Reptile	ViT_small	76.05	80.30	85.55	67.50	73.15	77.37
	ResNet50	70.28	75.78	78.83	54.47	58.22	61.58
	MetaMed	72.75	75.62	81.37	54.83	59.33	69.75
-	PFEMed	81.69	83.87	85.14	66.94	69.78	73.81

Table 5.9: Comparison with MetaMed and PFEMed: BreakHis x100 dataset

Algorithm	Model	2-way			3-way		
		3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
ProtoNet	ViT_small	80.64	83.80	**87.62**	**69.39**	**75.91**	**81.47**
	ResNet50	68.62	72.12	73.31	55.80	60.28	61.88
Reptile	ViT_small	73.45	77.90	86.18	55.05	63.38	75.92
	ResNet50	72.15	76.63	80.33	60.33	63.45	68.47
	MetaMed	78.75	81.38	83.88	63.08	66.42	74.08
-	PFEMed	**82.16**	**85.28**	86.90	69.21	75.04	78.93

Cutout, Mixup, and Cutmix augmentation techniques were tested on the ISIC 2018 dataset in this paper. Results are summarized in Table 5.7. For ProtoNet, the only applicable method, Cutout, resulted in lower scores for most tasks for both ViT_small and ResNet50. For Reptile, the situation is better. The use of Cutout led to lower performance in most cases, except for 3-way k-shot tasks of ResNet50. A similar situation is observed with CutMix, where results are generally lower for the majority of tasks. On the contrary, when input data was augmented using Mixup, there was an uplift in accuracy scores in 4 and 3 tasks out of 6 for ResNet50 and ViT_small, respectively. In general, Mixup performs better than the other techniques and can be recommended as a good data augmentation technique.

5.5.4 Comparison with other works

In this section, we compare the results of our models with those presented in the MetaMed [30] and PFEMed papers [8]. We focused on ViT_small and ResNet50 models, which were used in both ProtoNet and Reptile. The results are presented in Tables 5.8 - 5.10. It should be noted that all models were meta-trained without the use of advanced augmentation techniques. However, it should also be noted

Table 5.10: Comparison with MetaMed and PFEMed: Pap smear

Algorithm	Model	2-way			3-way		
		3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
ProtoNet	ViT_small	92.40	94.05	94.90	86.38	89.09	90.62
	ResNet50	70.49	71.75	69.61	57.74	58.48	59.60
Reptile	ViT_small	83.35	87.05	91.96	72.52	81.13	87.94
	ResNet50	71.44	74.59	78.39	48.00	49.86	50.44
	MetaMed	85.37	86.50	89.37	70.58	72.42	83.00
-	PFEMed	**95.53**	**95.87**	**96.00**	**92.42**	**92.48**	**92.68**

Table 5.11: Custom ViT architecture results on ISIC 2018 dataset.

Algorithm	Model	2-way			3-way		
		3-shot	5-shot	10-shot	3-shot	5-shot	10-shot
Protonet	ViT_small	**84.35**	**86.70**	**89.72**	**72.10**	**76.18**	**81.45**
	ViT_small_SE	77.84	80.66	84.36	64.30	68.24	74.66
	ConViT	**76.33**	**78.89**	**82.94**	**63.17**	**67.07**	**71.96**
	ConViT_SE	75.21	77.18	81.71	60.94	65.11	69.60

that MetaMed used a simple CNN model (with only 3840 parameters), which is a standard in FSL, while PFEMed utilizes a model with 72.95m parameters which is significantly higher than 22m and 23.5m parameters of ViT_small and ResNet50 respectively. Therefore, it may not be a fair comparison.

Upon analyzing the results across all datasets, we observed that ViT_small outperformed other models in all tasks when used with ProtoNet on ISIC 2018 dataset. On BreakHis x100, it showed the highest accuracy in 2-way-10-shot and all 3-way tasks. However, on Pap smear dataset, PFEMed showed higher results across all tasks. Generally, It can be observed that the ViT results scales better with the increasing number of shots in comparison with PFEMed. At the same time, ResNet50 failed to catch up with the performance of other models, including those presented in the MetaMed paper.

5.5.5 Custom ViT Results

For custom ViTs, we used ImageNet1k pretrained checkpoints and attached SE block on top of an encoder block. Then these modified models were trained on CIFAR 100 dataset in a supervised manner to train the SE block. After that the models were meta-trained using ProtoNet algorithm on ISIC 2018 dataset. The Table 5.11

demonstrates our findings. Generally, it is clear that unmodified models performed better than those with SE block attached. However, these are only preliminary results as the training was performed on CIFAR 100 dataset which is too small for proper training.

5.6 Discussion

From the results presented above, we can conclude that ViTs can be effectively used for few-shot medical image classification, especially when combined with ProtoNet, as they outperformed comparable CNNs in the majority of tasks. When compared with a bigger model which specializes on FSL, PFEMed, ViT_small managed to outperform it on ISIC 2018 in all tasks and on BreakHis x100 in 2-way-10-shot and all 3-way tasks. For Pap smear dataset, PFEMed showerd better results across all tasks. When comparing our results to those reported in the MetaMed paper [30], a ProtoNet with a ViT_small backbone demonstrated superior performance in all cases. Nonetheless, it is essential to take into account the differences in model size when interpreting these findings. The ViT_small model has considerably more parameters compared to the simpler CNN model used in MetaMed, which may contribute to the performance disparity observed. Additionally, it should be noted that the performance highly depends on the FSL algorithm used. It was observed that ViTs paired with ProtoNet showed much stronger results compared to the Reptile algorithm. Nevertheless, better hyperparameter selection may improve Reptile's performance, as demonstrated by changing the inner adaptation steps from 5 to 50. This may be a direction for further study.

Finally, the effect of advanced augmentation techniques is mostly negative, except for Mixup. When used with our best FSL algorithm - ProtoNet, ViT_small showed lower results with Cutout augmentation, suggesting that it should not be used during the meta-training phase. As for Reptile, Mixup improved the accuracy scores of the model in most cases, while other techniques showed positive results in less than 50% of tasks.

As for future study directions, there are several options, including designing a new ViT architecture, investigating the use of synthetic data augmentation techniques such as Variational Autoencoders or Generative Adversarial Networks. Additionally, we are still experimenting with our custom ViT architectures and plan to assess their performance after performing training with bigger datasets.

Chapter 6

Conclusion

In this study, we explored the application of vision transformer (ViT) architecture in medical image classification within a few-shot learning scenario. We evaluated several well-known ViT and CNN architectures, employing two few-shot learning algorithms, ProtoNet and Reptile, on three publicly available medical datasets: ISIC 2018, BreakHis, and Pap smear. Our results revealed that a ViT serving as a backbone for ProtoNet outperforms other configurations, including those with ResNet50. Additionally, we compared it with other notable works from the field. ViT_small paired with ProtoNet outperformed results presented in the MetaMed paper in all cases. When compared with PFEMed few-shot learner, our configuration showed better results in all task on ISIC 2018 dataset and 2-way-10-shot and all 3-way tasks on BreakHis x100 dataset, only falling short on Pap smear dataset. From this, we can state that ViTs, when paired with ProtoNets, can be effectively utilized for few-shot medical image classification tasks.

Furthermore, we assessed the effectiveness of Cutout, Mixup, and Cutmix data augmentation techniques when applied to ViT small and ResNet50 on the ISIC 2018 dataset. These augmentation techniques generally did not yield positive effects on the performance of models, except for the Mixup method. It improved test scores in 4 and 3 tasks out of 6 for ResNet50 and ViT_small models respectively when used with the Reptile algorithm. For ProtoNet scores, the only compatible augmentation method Cutout deteriorated the model performance in most of the cases.

We also proposed a new ViT architecture which combines ConViT and Squeeze&Excitation block. Preliminary results demonstrated that this architecture is performing worse than a standard ConViT when paired with ProtoNets. However, these are only preliminary scores from pretraining on a small dataset. Later, we plan to perform training on much bigger dataset to draw final conclusion on the performance of the proposed architecture.

For a future direction, we plan to use data generation techniques like Variational Autoencoders or GANs for input augmentation. Additionally, we may try to develop a ViT based architecture designed for FSL tasks.

Bibliography

[1] Mouath Aouayeb, Wassim Hamidouche, Catherine Soladie, Kidiyo Kpalma, and Renaud Seguier. Learning vision transformer with squeeze and excitation for facial expression recognition. *arXiv preprint arXiv:2107.03107*, 2021.

[2] Finn Behrendt, Debayan Bhattacharya, Julia Krüger, Roland Opfer, and Alexander Schlaefer. Data-efficient vision transformers for multi-label disease classification on chest radiographs. *Current Directions in Biomedical Engineering*, 8(1):34–37, 2022.

[3] Etienne Bennequin. easyfsl. `https://github.com/sicara/easy-few-shot-learning`, 2021.

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[5] Yudong Chen, Chaoyu Guan, Zhikun Wei, Xin Wang, and Wenwu Zhu. Metadelta: A meta-learning system for few-shot image classification. In *AAAI Workshop on Meta-Learning and MetaDL Challenge*, pages 17–28. PMLR, 2021.

[6] Yuzhong Chen, Zhenxiang Xiao, Lin Zhao, Lu Zhang, Haixing Dai, David Weizhong Liu, Zihao Wu, Changhe Li, Tuo Zhang, Changying Li, et al. Mask-guided vision transformer (mg-vit) for few-shot learning. *arXiv preprint arXiv:2205.09995*, 2022.

[7] Mehdi Cherti and Jenia Jitsev. Effect of pre-training scale on intra-and inter-domain full and few-shot transfer learning for natural and medical x-ray chest images. *arXiv preprint arXiv:2106.00116*, 2021.

[8] Zhiyong Dai, Jianjun Yi, Lei Yan, Qingwen Xu, Liang Hu, Qi Zhang, Jiahui Li, and Guoqiang Wang. Pfemed: Few-shot medical image classification using prior guided feature enhancement. *Pattern Recognition*, 134:109108, 2023.

[9] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] Linh T Duong, Nhi H Le, Toan B Tran, Vuong M Ngo, and Phuong T Nguyen. Detection of tuberculosis from chest x-ray images: boosting the performance with vision transformer and transfer learning. *Expert Systems with Applications*, 184:115519, 2021.

[12] Stéphane d'Ascoli, Hugo Touvron, Matthew L Leavitt, Ari S Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. In *International Conference on Machine Learning*, pages 2286–2296. PMLR, 2021.

[13] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[15] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5149–5169, 2021.

[16] Shell Xu Hu, Da Li, Jan Stühmer, Minyoung Kim, and Timothy M Hospedales. Pushing the limits of simple pipelines for few-shot learning: External data and fine-tuning make a difference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9068–9077, 2022.

[17] Mike Huisman, Jan N Van Rijn, and Aske Plaat. A survey of deep meta-learning. *Artificial Intelligence Review*, 54(6):4483–4541, 2021.

[18] Jan Jantzen, Jonas Norup, Georgios Dounias, and Beth Bjerregaard. Pap-smear benchmark data for pattern classification. *Nature inspired Smart Information Systems (NiSIS 2005)*, pages 1–9, 2005.

[19] Juntao Jiang and Shuyi Lin. Covid-19 detection in chest x-ray images using swin-transformer and transformer in transformer. *arXiv preprint arXiv:2110.08427*, 2021.

[20] Koushik Sivarama Krishnan and Karthik Sivarama Krishnan. Vision transformer based covid-19 detection using chest x-rays. In *2021 6th International Conference on Signal Processing, Computing and Control (ISPCC)*, pages 644–648. IEEE, 2021.

[21] Chengeng Liu and Qingshan Yin. Automatic diagnosis of covid-19 using a tailored transformer-like network. In *Journal of Physics: Conference Series*, volume 2010, page 012175. IOP Publishing, 2021.

[22] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[23] Christos Matsoukas, Johan Fredin Haslum, Magnus Söderberg, and Kevin Smith. Is it time to replace cnns with transformers for medical images? *arXiv preprint arXiv:2108.09038*, 2021.

[24] Sachin Mehta and Mohammad Rastegari. Separable self-attention for mobile vision transformers, 2022.

[25] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[26] Sangjoon Park, Gwanghyun Kim, Yujin Oh, Joon Beom Seo, Sang Min Lee, Jin Hwan Kim, Sungjun Moon, Jae-Kwang Lim, and Jong Chul Ye. Multitask vision transformer using low-level chest x-ray feature corpus for covid-19 diagnosis and severity quantification. *Medical Image Analysis*, 75:102299, 2022.

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[28] Shehan Perera, Srikar Adhikari, and Alper Yilmaz. Pocformer: A lightweight transformer architecture for detection of covid-19 using point of care ultrasound. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 195–199. IEEE, 2021.

[29] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[30] Rishav Singh, Vandana Bharti, Vishal Purohit, Abhinav Kumar, Amit Kumar Singh, and Sanjay Kumar Singh. Metamed: Few-shot medical image classification using gradient-based meta-learning. *Pattern Recognition*, 120:108111, 2021.

[31] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

[32] Yisheng Song, Ting Wang, Subrota K Mondal, and Jyoti Prakash Sahoo. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. *arXiv preprint arXiv:2205.06743*, 2022.

[33] Fabio A Spanhol, Luiz S Oliveira, Caroline Petitjean, and Laurent Heutte. A dataset for breast cancer histopathological image classification. *Ieee transactions on biomedical engineering*, 63(7):1455–1462, 2015.

[34] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018.

[35] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[37] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

[38] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[39] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020.

[40] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

[41] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

[42] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[43] Jinyi Zou, Xiao Ma, Cheng Zhong, and Yao Zhang. Dermoscopic image analysis for isic challenge 2018. *arXiv preprint arXiv:1807.08948*, 2018.