Journal of *Imaging*

MDPI

Article

An Optimization-Based Meta-Learning Model for MRI Reconstruction with Diverse Dataset

Wanyu Bian [1,*,†], Yunmei Chen [1], Xiaojing Ye [2] and Qingchao Zhang [1]

[1] Department of Mathematics, University of Florida, Gainesville, FL 32611, USA; yun@ufl.edu (Y.C.); qingchaozhang@ufl.edu (Q.Z.)
[2] Department of Mathematics and Statistics, Georgia State University, Atlanta, GA 30303, USA; xye@gsu.edu
* Correspondence: wanyu.bian@ufl.edu
† Author list in alphabetical order.

Abstract: This work aims at developing a generalizable Magnetic Resonance Imaging (MRI) reconstruction method in the meta-learning framework. Specifically, we develop a deep reconstruction network induced by a learnable optimization algorithm (LOA) to solve the nonconvex nonsmooth variational model of MRI image reconstruction. In this model, the nonconvex nonsmooth regularization term is parameterized as a structured deep network where the network parameters can be learned from data. We partition these network parameters into two parts: a task-invariant part for the common feature encoder component of the regularization, and a task-specific part to account for the variations in the heterogeneous training and testing data. We train the regularization parameters in a bilevel optimization framework which significantly improves the robustness of the training process and the generalization ability of the network. We conduct a series of numerical experiments using heterogeneous MRI data sets with various undersampling patterns, ratios, and acquisition settings. The experimental results show that our network yields greatly improved reconstruction quality over existing methods and can generalize well to new reconstruction problems whose undersampling patterns/trajectories are not present during training.

Keywords: MRI reconstruction; meta-learning; domain generalization

Citation: Bian, W.; Chen, Y.; Ye, X.; Zhang, Q. An Optimization-Based Meta-Learning Model for MRI Reconstruction with Diverse Dataset. *J. Imaging* **2021**, *7*, 231. https://doi.org/10.3390/jimaging7110231

Academic Editors: Fabiana Zama and Elena Loli Piccolomini

Received: 23 September 2021
Accepted: 28 October 2021
Published: 31 October 2021

1. Introduction

Deep learning methods have demonstrated promising performance in a variety of image reconstruction problems. However, deep learning models are often trained for specific tasks and require the training samples to follow the corresponding distribution. In particular, the source-domain/training samples and target-domain/testing samples need to be drawn from the same distribution [1–4]. In practice, these data sets are often collected at different sources and exhibit substantial heterogeneity, and thus the samples may follow related but different distributions in real-world applications [5,6]. Therefore, the robust and efficient training of deep neural networks using such data sets is theoretically important and practically relevant in the application of deep learning-based methods.

Meta-learning provides a unique paradigm to achieve robust and efficient neural network training [1,4,7–10]. Meta-learning is known as *learning-to-learn* and aims to quickly learn unseen tasks from the experience of learning episodes that cover the distribution of relevant tasks. In a multiple-task scenario, given a family of tasks, meta-learning has been proven to be a useful tool for extracting task-agnostic knowledge and improving the learning performance of new tasks from that family [11,12]. We leverage this feature of meta-learning for network training where the MRI training data are acquired by using different under-sampling patterns (e.g., Cartesian mask, Radial mask, Poisson mask), under-sampling ratios, and different settings of the scanning parameters, which result in different levels of contrast (e.g., T1-weighted, T2-weighted, proton-density (PD), and Flair). These data are vastly heterogeneous and can be considered as being from various tasks.

Thus, our goal is to develop a robust and generalizable image reconstruction method in the meta-learning framework to leverage such large-scale heterogeneous data for MRI reconstruction.

Our approach can be outlined as follows. First, we introduce a variational model rendering a nonconvex nonsmooth optimization problem for image reconstruction. In our variational model, the regularization term is parameterized as a structured deep neural network where the network parameters can be learned during the training process. We then propose a learnable optimization algorithm (LOA) with rigorous convergence guarantees to solve this optimization problem. Then, we construct a deep reconstruction network by following this LOA exactly; namely, each phase of this LOA-induced network is exactly one iteration of the LOA. This approach is inspired by [13], but the LOA developed in the present work is computationally more efficient than that in [13]: the safeguard variable in this work is updated only if necessary, which can significantly reduce computational cost while retaining the convergence guarantee.

Second, to improve network robustness and mitigate the overfitting issue, we explicitly partition the network parameters of the regularization into two parts: a task-invariant part and a task-specific part. The former extracts common prior information of images from different tasks in the training data and learns the task-invariant component of the regularization. The latter, on the other hand, is obtained from another network which exploits proper task-specific components (also called meta-knowledge) of the regularization for different tasks. The hyperparameters of this network are also learned during training. Furthermore, we split the available data into two sets: the training set and the validation set. Then, we introduce a bilevel optimization model for learning network parameters. Specifically, the lower-level problem (also known as inner problem) finds the task-invariant part of the regularization with the fixed task-specific part on the training dataset, whereas the upper-level (outer) problem seeks for the optimal task-specific part of the regularization parameter using the validation dataset. This approach greatly increases the robustness of the learned regularization, meaning that the trained LOA-induced deep reconstruction network can generalize well to unseen tasks.

As demonstrated by our numerical experiments in Section 5, our proposed framework yields much improved image qualities using diverse data sets of various undersampling trajectories and ratios for MRI image reconstruction. The reason is that effective regularization can integrate common features and prior information from a variety of training samples from diverse data sets, but they need to be properly weighed against the data fidelity term obtained in specific tasks (i.e., undersampling trajectory and ratios). Our contributions can be summarized as follows:

1. An LOA inspired network architecture—our network architecture exactly follows a proposed LOA with guaranteed convergence. Thus, the network is more interpretable, parameter-efficient, and stable than existing unrolling networks.
2. Adaptive design of regularization—our adaptive regularizer consists of a task-invariant part and a task-specific part, both of which can be appropriately trained from data.
3. Improved network robustness and generalization ability—we improve the robustness of the network parameter training process by posing it as a bilevel optimization using training data in the lower-level and validation data in the upper-level. This approach also improves the generalization ability of the trained network so that it can be quickly adapted to image reconstruction with new unseen sampling trajectories and produces high-quality reconstructions.

The remainder of the paper is organized as follows. In Section 2, we discuss related work for both optimization-based meta-learning and deep unrolled networks for MRI reconstructions. We propose our meta-learning model and the neural network in Section 3 and describe the implementation details in Section 4. Section 5 provides the numerical results of the proposed method. Section 6 concludes the paper.

2. Related Work

In recent years, meta-learning methods have demonstrated promising results in various fields with different techniques [12]. Meta-learning techniques can be categorized into three groups [14–16]: metric-based methods [17–19], model-based methods [20–23], and optimization-based methods [8,24,25]. Optimization-based methods are often cast as a bilevel optimization problem and exhibit relatively better generalizability for wider task distributions. We mainly focus on optimization-based meta-learning in this paper. For more comprehensive literature reviews and developments of meta-learning, we refer the readers to the recent surveys [12,16].

Optimization-based meta-learning methods have been widely used in a variety of deep learning applications [8,24–31]. The network training problems in these meta-learning methods are often cast as the bilevel optimization of a leader variable and a follower variable. The constraint of the bilevel optimization is that the follower variable is optimal for the lower-level problem for each fixed leader variable, and the ultimate goal of bilevel optimization is to find the optimal leader variable (often, the corresponding optimal follower variable as well) that minimizes the upper-level objective function under the constraint. The lower-level problem is approximated by one or a few gradient descent steps in many existing optimization-based meta learning applications, such as Model-Agnostic Meta-Learning (MAML) [8], and a large number of followup works of MAML proposed to improve generalization using similar strategy [9,15,27,29,30,32–34]. Deep bilevel learning [35] seeks to obtain better generalization than when trained on one task and generalize well to another task. The model is used to optimize a regularized loss function to find network parameters from the training set and identify hyperparameters so that the network performs well on the validation dataset.

When the unseen tasks lie in inconsistent domains with the meta-training tasks, as revealed in [36], the generalization behavior of the meta-learner will be compromised. This phenomenon partially arises from the meta-overfitting on the already seen meta-training tasks, which is identified as a memorization problem in [34]. A meta-regularizer forked with information theory is proposed in [34] to handle the memorization problem by regulating the information dependency during the task adaption. MetaReg [4] decouples the entire network into the feature network and task network, where the meta-regularization term is only applied to the task network. They first update the parameters of the task network with a meta-train set to obtain the domain-aligned task network and then update the parameters of the meta-regularization term on the meta-test set to learn the cross-domain generalization. In contrast to MetaReg, Feature-Critic Networks [37] exploit the meta-regularization term to pursue a domain-invariant feature extraction network. The meta-regularization is designed as a feature-critic network that takes the extracted feature as an input. The parameters of the feature extraction network are updated by minimizing the new meta-regularized loss. The auxiliary parameters in the feature-critic network are learned by maximizing the performance gain over the non-meta case. To effectively evaluate the performance of the meta-learner, several new benchmarks [38–40] were developed under more realistic settings that operate well on diverse visual domains. As mentioned in [39], the generalization to unseen tasks within multimodal or heterogeneous datasets remains a challenge to the existing meta-learning methods.

The aforementioned methods pursue domain generalization for the classification networks that learned a regularization function to learn cross-domain generalization. Our proposed method was developed to solve the inverse problem, and we construct an adaptive regularization that not only learns the universal parameters among tasks but also the task-aware parameters. The designated adaptive regularizer assists the generalization ability of the deep model so that the well-trained model can perform well on heterogeneous datasets of both seen and unseen tasks.

3. Proposed Method

3.1. Preliminaries

We first provide the background of compressed sensing MRI (CS-MRI), the image reconstruction problem, and the learned optimization algorithm to solve the image reconstruction problem. CS-MRI accelerates MRI data acquisition by under-sampling the k-space (Fourier space) measurements. The under-sampled k-space measurement are related to the image by the following formula [41]:

$$\mathbf{y} = \mathbf{P}\mathcal{F}\mathbf{x} + \mathbf{n}, \tag{1}$$

where $\mathbf{y} \in \mathbb{C}^p$ represents the measurements in k-space with a total of p sampled data points, $\mathbf{x} \in \mathbb{C}^{N\times 1}$ is the MR image to be reconstructed with N pixels, $\mathcal{F} \in \mathbb{C}^{N\times N}$ is the 2D discrete Fourier transform (DFT) matrix, and $\mathbf{P} \in \mathbb{R}^{p\times N}$ $(p < N)$ is the binary matrix representing the sampling trajectory in k-space. $\mathbf{n}$ is the acquisition noise in k-space.

Solving $\mathbf{x}$ from (noisy) under-sampled data $\mathbf{y}$ according to (1) is an ill-posed problem. An effective strategy to elevate the ill-posedness issue is to incorporate prior information to the reconstruction. The variational method is one of the most effective ways to achieve this. The general framework of this method is to minimize an objective function that consists of a data fidelity term and a regularization term as follows:

$$\bar{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{P}\mathcal{F}\mathbf{x} - \mathbf{y}\|^2 + R(\mathbf{x}), \tag{2}$$

where the first term is data fidelity, which ensures consistency between the reconstructed $\mathbf{x}$ and the measured data $\mathbf{y}$, and the second term $R(\mathbf{x})$ is the regularization term, which introduces prior knowledge to the image to be reconstructed. In traditional variational methods, $R(\mathbf{x})$ is a hand-crafted function such as Total Variation (TV) [42]. The advances of the optimization techniques allowed more effective algorithms to solve the variational models with theoretical justifications. However, hand-crafted regularizers may be too simple to capture subtle details and satisfy clinic diagnostic quality.

In recent years, we have witnessed the tremendous success of deep learning in solving a variety of inverse problems, but the interpretation and generalization of these deep-learning-based methods still remain the main concerns. As an improvement over generic black-box-type deep neural networks (DNNs), several classes of learnable optimization algorithms (LOAs) inspired neural networks, known as unrolling networks, which unfold iterative algorithms to multi-phase networks and have demonstrated promising solution accuracy and efficiency empirically [43–50]. However, many of them are only specious imitations of the iterative algorithms and hence lack the backbone of the variational model and any convergence guarantee.

In light of the substantial success of deep learning and the massive amount of training data now available, we can parameterize the regularization term as a deep convolutional neural network (CNN) that learns from training samples. LOA-induced reconstruction methods have been successfully applied to CS-MRI to solve inverse problems with a learnable regularizer:

$$\arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{P}\mathcal{F}\mathbf{x} - \mathbf{y}\|^2 + R(\mathbf{x};\Theta). \tag{3}$$

where $R(\mathbf{x};\Theta)$ is the regularization parameterized as a deep network with parameter Θ. Depending on the specific parametric form of $R(\mathbf{x};\Theta)$ and the optimization scheme used for unrolling, several unrolling networks have been proposed in recent years. For example, the variational network (VN) [51] was introduced to unroll the gradient descent algorithm and parametrize the regularization as a combination of linear filters and nonlinear CNNs. MoDL [52] proposed a weight sharing strategy in a recursive network to learn the regularization parameters by unrolling the conjugate gradient method. ADMM-Net [53] mimics the celebrated alternating direction method of multipliers; the regularizer is designed to be L_1-norm replaced by a piecewise linear function. ISTA-Net [54] considers the regularizer

as the L_1-norm of a convolutional network. The network unrolls several phases iteratively, and each phase mimics one iteration of iterative shrinkage thresholding algorithm (ISTA) [55,56]. However, these networks only superficially mimic the corresponding optimization schemes, but they lack direct relations to the original optimization method or variational model and do not retain any convergence guarantee. In this work, we first develop a learnable optimization algorithm (LOA) for (3) with comprehensive convergence analysis and obtain an LOA-induced network by following the iterative scheme of the LOA exactly.

3.2. LOA-Induced Reconstruction Network

In this section, we first introduce a learned optimization algorithm (LOA) to solve (3) where the regularization network parameter Θ is fixed. As Θ is fixed in (3), we temporarily omit this in the derivation of the LOA below and write $R(\mathbf{x};\Theta)$ as $R(\mathbf{x})$ for notation simplicity.

In this work, to incorporate sparsity along with the learned features, we parameterize the function $R(\mathbf{x}) = \kappa \cdot r(\mathbf{x})$, where $\kappa > 0$ is a weight parameter that needs be chosen properly depending on the specific task (e.g., noise level, undersampling ratio, etc.), and r is a regularizer parameterized as a composition of neural networks and can be adapted to a broad range of imaging applications. Specifically, we parameterize r as the composition of the $l_{2,1}$ norm and a learnable feature extraction operator $\mathbf{g}(\mathbf{x})$. That is, we set r in (11) to be

$$r(\mathbf{x}) := \|\mathbf{g}(\mathbf{x})\|_{2,1} = \sum_{j=1}^{m} \|\mathbf{g}_j(\mathbf{x})\|. \tag{4}$$

Here, ":=" stands for "defined as". $\mathbf{g}(\cdot) = (\mathbf{g}_1(\cdot), \ldots, \mathbf{g}_m(\cdot))$, $\mathbf{g}_j(\cdot) = \mathbf{g}_j(\cdot;\theta)$ is parametrized as a convolutional neural network (CNN) for $j = 1, \cdots, m$, and θ is the learned and fixed network parameter in $r(\cdot;\theta)$, as mentioned above. We also consider κ to be learned and fixed as θ for now, and we discuss how to learn both of them in the next subsection. We use a smooth activation function in $\mathbf{g}$ as formulated in (20), which renders $\mathbf{g}$ a smooth but nonconvex function. Due to the nonsmooth $\|\cdot\|_{2,1}$, r is therefore a nonsmooth nonconvex function.

Since the minimization problem in (11) is nonconvex and nonsmooth, we need to derive an efficient LOA to solve it. Here, we first consider smoothing the $l_{2,1}$ norm that for any fixed $\mathbf{g}(\mathbf{x})$ is

$$r_\varepsilon(\mathbf{x}) = \textstyle\sum_{j=1}^{m} \sqrt{\|\mathbf{g}_j(\mathbf{x})\|^2 + \varepsilon^2} - \varepsilon. \tag{5}$$

We denote $R_\varepsilon = \kappa \cdot r_\varepsilon$. The LOA derived here is inspired by the proximal gradient descent algorithm and iterates the following steps to solve the smoothed problem:

$$\mathbf{z}_{t+1} = \mathbf{x}_t - \alpha_t \nabla f(\mathbf{x}_t) \tag{6a}$$
$$\mathbf{x}_{t+1} = \mathrm{prox}_{\alpha_t R_{\varepsilon_t}}(\mathbf{z}_{t+1}), \tag{6b}$$

where ε_t denotes the smoothing parameter ε at the specific iteration t, and the proximal operator is defined as $\mathrm{prox}_{\alpha g}(\mathbf{b}) := \arg\min_{\mathbf{x}} \|\mathbf{x} - \mathbf{b}\| + \alpha g(\mathbf{x})$ in (6b). A quick observation from (5) is that $R_\varepsilon \to R$ as ε diminishes, so later we intentionally push $\varepsilon_t \to 0$ at Line 16 in Algorithm 1. Then, one can readily show that $R_\varepsilon(x) \le R(x) \le R_\varepsilon(x) + \varepsilon$ for all x and $\varepsilon > 0$. From Algorithm 1, line 16 automatically reduces ε, and the iterates will converge to the solution of the original nonsmooth nonconvex problem (11)—this is clarified precisely in the convergence analysis in Appendix A.

Since R_{ε_t} is a complex function involving a deep neural network, its proximal operator does not have a closed form and cannot be computed easily in the subproblem in (6b). To overcome this difficulty, we consider to approximate R_{ε_t} by

$$\hat{R}_{\varepsilon_t}(\mathbf{z}_{t+1}) = R_{\varepsilon_t}(\mathbf{z}_{t+1}) + \langle \nabla R_{\varepsilon_t}(\mathbf{z}_{t+1}), \mathbf{x} - \mathbf{z}_{t+1} \rangle + \frac{1}{2\beta_t} \|\mathbf{x} - \mathbf{z}_{t+1}\|^2. \tag{7}$$

Then, we update $\mathbf{u}_{t+1} = \text{prox}_{\alpha_t \hat{R}_{\varepsilon_t}}(\mathbf{z}_{t+1})$ to replace (6b); therefore, we obtain

$$\mathbf{u}_{t+1} = \mathbf{z}_{t+1} - \tau_t \nabla R_{\varepsilon_t}(\mathbf{z}_{t+1}), \text{ where } \tau_t = \frac{\alpha_t \beta_t}{\alpha_t + \beta_t}. \tag{8}$$

In order to guarantee the convergence of the algorithm, we introduce the standard gradient descent of ϕ_{ε_t} (where $\phi_{\varepsilon_t} := f + R_{\varepsilon_t}$) at $\mathbf{x}$:

$$\mathbf{v}_{t+1} = \arg\min_{\mathbf{x}} \langle \nabla f(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle + \langle \nabla R_{\varepsilon}(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle + \frac{1}{2\alpha_t} \|\mathbf{x} - \mathbf{x}_t\|^2, \tag{9}$$

which yields

$$\mathbf{v}_{t+1} = \mathbf{x}_t - \alpha_t \nabla \phi_{\varepsilon_t}(\mathbf{x}_t), \tag{10}$$

to serve as a safeguard for the convergence. Specifically, we set $\mathbf{x}_{t+1} = \mathbf{u}_{t+1}$ if $\phi_{\varepsilon_t}(\mathbf{u}_{t+1}) \leq \phi_{\varepsilon_t}(\mathbf{v}_{t+1})$; otherwise, we set $\mathbf{x}_{t+1} = \mathbf{v}_{t+1}$. Then, we repeat this process.

Our algorithm is summarized in Algorithm 1. The prior term with unknown parameters has the exact residual update itself which improves the learning and training process [57]. The condition checking in Line 5 is introduced to make sure that it is in the energy descending direction. Once the condition in Line 5 fails, the process moves to $\mathbf{v}_{t+1}$, and the line search in Line 12 guarantees that the appropriate step size can be achieved within finite steps to make the function value decrease. From Line 3 to Line 14, we consider that it solves a problem of minimizing ϕ_{ε_t} with ε_t fixed. Line 15 is used to update the value of ε_t depending on a reduction criterion. The detailed analysis of this mechanism and in-depth convergence justification is shown in Appendix A. The corresponding unrolling network exactly follows Algorithm 1 and thus shares the same convergence property. Compared to LDA [13], which computes both candidates $\mathbf{u}_{t+1}$, $\mathbf{v}_{t+1}$ at every iteration and then chooses the one that achieves a smaller function value, we propose the criteria above in Line 5 for updating $\mathbf{x}_{t+1}$, which potentially saves extra computational time for calculating the candidate $\mathbf{v}_{t+1}$ and potentially mitigates the frequent alternations between the two candidates. Besides, the smoothing method proposed in this work is more straightforward than smoothing in dual space [13] while still keeping provable convergence, as shown in Theorem A5.

The proposed LOA-induced network is a multi-phase network whose architecture exactly follows the proposed LOA (Algoirthm 1) in the way that each phase corresponds to one iteration of the algorithm. Specifically, we construct a deep network, denoted by F_Θ, that follows Algorithm 1 exactly for a user-specified number of T iterations. Here, Θ denotes the set of learnable parameters in F_Θ, which includes the regularization network parameter θ, weight κ, and other algorithmic parameters of Algorithm 1. Therefore, for any input under-sampled k-space measurement $\mathbf{y}$, $F_\Theta(\mathbf{y})$ executes the LOA (Algorithm 1) for T iterations and generates an approximate solution of the minimization problem (11):

$$F_\Theta(\mathbf{y}) \approx \arg\min_{\mathbf{x}} \left\{ \phi_\Theta(\mathbf{x}, \mathbf{y}) := f(\mathbf{x}, \mathbf{y}) + R(\mathbf{x}; \Theta) \right\}. \tag{11}$$

where we use "$\approx$" since F_Θ follows only finitely many steps of the optimization algorithm to approximate the solution. It is worth emphasizing that this approach can be readily applied to a much broader class of image reconstruction problems as long as f is (possibly nonconvex and) continuously differentiable with the Lipschitz continuous gradient. In the next subsection, we develop a meta-learning based approach for the robust training of the network parameter Θ.

3.3. Bilevel Optimization Algorithm for Network Training

In this section, we consider the parameter training problem of the LOA-induced network F_Θ. Specifically, we develop a bilevel optimization algorithm to train our network parameters Θ from diverse data sets to improve network robustness and generalization ability.

Algorithm 1: Algorithmic Unrolling Method with Provable Convergence

Input: Initial $\mathbf{x}_0$, $0 < \rho, \gamma < 1$, and $\varepsilon_0, a, \sigma > 0$. Max total phases T or tolerance $\epsilon_{\text{tol}} > 0$.
for $t = 0, 1, 2, \ldots, T-1$ **do**
  $\mathbf{z}_{t+1} = \mathbf{x}_t - \alpha_t \nabla f(\mathbf{x}_t)$
  $\mathbf{u}_{t+1} = \mathbf{z}_{t+1} - \tau_t \nabla R_{\varepsilon_t}(\mathbf{z}_{t+1})$,
  if $\|\nabla \phi_{\varepsilon_t}(\mathbf{x}_t)\| \le a\|\mathbf{u}_{t+1} - \mathbf{x}_t\|$ and $\phi_{\varepsilon_t}(\mathbf{u}_{t+1}) - \phi_{\varepsilon_t}(\mathbf{x}_t) \le -\frac{1}{a}\|\mathbf{u}_{t+1} - \mathbf{x}_t\|^2$ **then**
    set $\mathbf{x}_{t+1} = \mathbf{u}_{t+1}$,
  else
    $\mathbf{v}_{t+1} = \mathbf{x}_t - \alpha_t \nabla \phi_{\varepsilon_t}(\mathbf{x}_t)$,
    if $\phi_{\varepsilon_t}(\mathbf{v}_{t+1}) - \phi_{\varepsilon_t}(\mathbf{x}_t) \le -\frac{1}{a}\|\mathbf{v}_{t+1} - \mathbf{x}_t\|^2$ holds **then**
      set $\mathbf{x}_{t+1} = \mathbf{v}_{t+1}$,
    else
      update $\alpha_t \leftarrow \rho\alpha_t$, then **go to** 8,
    end if
  end if
  if $\|\nabla \phi_{\varepsilon_t}(\mathbf{x}_{t+1})\| < \sigma\gamma\varepsilon_t$, set $\varepsilon_{t+1} = \gamma\varepsilon_t$; **otherwise**, set $\varepsilon_{t+1} = \varepsilon_t$.
  if $\sigma\varepsilon_t < \epsilon_{\text{tol}}$, terminate.
end for
Output: $\mathbf{x}_t$.

Recall that the LOA-induced network F_Θ exactly follows Algorithm 1, which is designed to solve the variational model (11) containing learnable regularization $R(\mathbf{x}; \Theta)$. As shown in Section 3.2, we design $R(\mathbf{x}; \Theta) = \kappa \cdot r(\mathbf{x}; \Theta)$, where r is learned to capture the intrinsic property of the underlying common features across all different tasks. To account for the large variations in the diverse training/validation data sets, we introduce a task-specific parameter ω_i to approximate the proper κ for the ith task. Specifically, for the ith task, the weight κ is set to $\sigma(\omega_i) \in (0, 1)$, where $\sigma(\cdot)$ is the sigmoid function. Therefore, $\kappa = \sigma(\omega_i)$ finds the proper weight of r for the i-th task according to its specific sampling ratio or pattern. The parameters ω_i are to be optimized in conjunction with Θ through the hyperparameter tuning process below.

Suppose that we are given $\mathcal{M}$ data pairs $\{(\mathbf{y}_m, \mathbf{x}_m^*)\}_{m=1}^{\mathcal{M}}$ for the use of training and validation, where $\mathbf{y}_m$ is the observation, which is the partial k-space data in our setting, and $\mathbf{x}_m^*$ is the corresponding ground truth image. The data pairs are then sampled into $\mathcal{N}$ tasks $\{\mathcal{D}_{\tau_i}\}_{i=1}^{\mathcal{N}}$, where each $\mathcal{D}_{\tau_i}$ represents the collection of data pairs in the specific task τ_i. In each task τ_i, we further divide the data into the task-specific training set $\mathcal{D}_{\tau_i}^{tr}$ and validation set $\mathcal{D}_{\tau_i}^{val}$. The architecture of our base network exactly follows the LOA (Algorithm 1) developed in the previous section with learnable parameters θ and a task-specific parameter ω_i for the ith task. More precisely, for one data sample denoted by $(\mathbf{y}_j^{(i)}, \mathbf{x}*_j^{(i)})$ in task τ_i with index j, we propose the algorithmic unrolling network for task τ_i as

$$F_{\theta,\omega_i}(\mathbf{y}_j^{(i)}) \approx \arg\min_{\mathbf{x}} f(\mathbf{x}, \mathbf{y}_j^{(i)}) + \sigma(\omega_i) r(\mathbf{x}; \theta), \tag{12}$$

where θ denotes the learnable common parameters across different tasks with task-invariant representation, whereas ω_i is a task-specific parameter for task τ_i. The weight $\sigma(\omega_i)$ represents the weight of r associated with the specific task τ_i. In our proposed network, Θ is the collection of (θ, ω_i) for task $i = 1, \cdots \mathcal{N}$. We denote ω to be the set $\{\omega_i\}_{i=1}^{\mathcal{N}}$. The detailed architecture of this network is illustrated in Section 3.2. We define the task-specific loss

$$\ell_{\tau_i}(\theta, \omega_i; \mathcal{D}_{\tau_i}) := \sum_{j=1}^{|\mathcal{D}_{\tau_i}|} \ell\big(F_{\theta,\omega_i}(\mathbf{y}_j^{(i)}), \mathbf{x}*_j^{(i)}\big), \tag{13}$$

where $|\mathcal{D}_{\tau_i}|$ represents the cardinality of $\mathcal{D}_{\tau_i}$ and

$$\ell\big(F_{\theta,\omega_i}(\mathbf{y}_j^{(i)}), \mathbf{x}*_j^{(i)}\big) := \frac{1}{2}\|F_{\theta,\omega_i}(\mathbf{y}_j^{(i)}) - \mathbf{x}*_j^{(i)}\|^2. \tag{14}$$

For the sake of preventing the proposed model from overfitting the training data, we introduce a novel learning framework by formulating the network training as a bilevel optimization problem to learn ω and θ in (12) as

$$\min_{\theta,\ \omega=\{\omega_i: i\in[N]\}} \quad \sum_{i=1}^{\mathcal{N}} \ell_{\tau_i}(\theta(\omega), \omega_i; \mathcal{D}_{\tau_i}^{val}) \tag{15a}$$

$$\text{s.t.} \qquad \theta(\omega) = \arg\min_{\theta} \sum_{i=1}^{\mathcal{N}} \ell_{\tau_i}(\theta, \omega_i; \mathcal{D}_{\tau_i}^{tr}). \tag{15b}$$

In (15), the lower-level optimization learns the task-invariant parameters θ of the feature encoder with the fixed task-specific parameter ω_i on the training dataset, and the upper-level adjusts the task-specific parameters $\{\omega_i\}$ so that the task-invariant parameters θ can perform robustly on the validation dataset as well. For simplicity, we omit the summation and redefine $\mathcal{L}(\theta, \omega; \mathcal{D}) := \sum_{i=1}^{\mathcal{N}} \ell_{\tau_i}(\theta, \omega; \mathcal{D})$ and then briefly rewrite (15) as

$$\min_{\theta,\omega} \mathcal{L}(\theta(\omega), \omega; \mathcal{D}^{val}) \quad \text{s.t. } \theta(\omega) = \arg\min_{\theta} \mathcal{L}(\theta, \omega; \mathcal{D}^{tr}). \tag{16}$$

Then, we relax (16) into a single-level constrained optimization where the lower-level problem is replaced with its first-order necessary condition following [58]

$$\min_{\theta,\omega} \mathcal{L}(\theta, \omega; \mathcal{D}^{val}) \quad \text{s.t. } \nabla_\theta \mathcal{L}(\theta, \omega; \mathcal{D}^{tr}) = 0. \tag{17}$$

which can be further approximated by an unconstrained problem by a penalty term as

$$\min_{\theta,\omega} \big\{\widetilde{\mathcal{L}}(\theta, \omega; \mathcal{D}^{tr}, \mathcal{D}^{val}) := \mathcal{L}(\theta, \omega; \mathcal{D}^{val}) + \frac{\lambda}{2}\|\nabla_\theta \mathcal{L}(\theta, \omega; \mathcal{D}^{tr})\|^2\big\}. \tag{18}$$

We adopt the stochastic gradients of the loss functions on mini-batch data sets in each iteration. In our model, we need to include the data pairs of multiple tasks in one batch; therefore, we propose the cross-task mini-batches when training. At each training iteration, we randomly sample the training data pairs $\mathcal{B}_{\tau_i}^{tr} = \{(\mathbf{y}_j^{(i)}, \mathbf{x}*_j^{(i)}) \in \mathcal{D}_{\tau_i}^{tr}\}_{j=1}^{\mathcal{J}^{tr}}$ and the validation pairs $\mathcal{B}_{\tau_i}^{val} = \{(\mathbf{y}_j^{(i)}, \mathbf{x}*_j^{(i)}) \in \mathcal{D}_{\tau_i}^{val}\}_{j=1}^{\mathcal{J}^{val}}$ on each task τ_i. Then, the overall training and validation mini-batches $\mathcal{B}^{tr}$ and $\mathcal{B}^{val}$ used in every training iteration are composed of the sampled data pairs from the entire set of tasks; i.e., $\mathcal{B}^{tr} = \bigcup_{i=1}^{\mathcal{N}}\{\mathcal{B}_{\tau_i}^{tr}\}$ and $\mathcal{B}^{val} = \bigcup_{i=1}^{\mathcal{N}}\{\mathcal{B}_{\tau_i}^{val}\}$. Thus in each iteration, we have $\mathcal{N}\cdot\mathcal{J}^{tr}$ and $\mathcal{N}\cdot\mathcal{J}^{val}$ data pairs used for training and validation, respectively. To solve the minimization problem (17), we utilize the stochastic mini-batch alternating direction method summarized in Algorithm 2, which is modified from [58].

As analyzed in [58], this penalty-type method has linear time complexity without computing the Hessian of the low level and only requires a constant space since we only need to store the intermediate θ, ω at each training iteration, which is suitable for solving the large-scale bilevel optimization problem. Algorithm 2 relaxes the bi-level optimization problem to a single-level constrained optimization problem by using the first-order necessary condition, which is not equivalent to the original problem but is much easier and efficient to solve. In the inner-loop (Line 5–9) of Algorithm 2, we continue minimizing the converted single-level optimization function (18) with respect to θ for K steps and then ω once alternatively until the condition with tolerance δ in Line 5 fails. The basic idea behind the condition in Line 5 arises from the first-order necessary condition as we would like to push the gradient of $\widetilde{\mathcal{L}}$ toward 0. Furthermore, at Line 11 of the outer

loop (Line 2–11), we decrease the tolerance δ. Combining Line 5 and 11 guarantees that each time the inner loop terminates, the gradients of $\widetilde{\mathcal{L}}$ with respect to θ and ω become increasingly close to 0. The parameter δ_{tol} is used to control the accuracy of the entire algorithm, and the outer-loop will terminate when δ is sufficiently small (i.e., $\delta \leq \delta_{tol}$). In addition, λ is the weight for the second constraint term of (18); in the beginning, we set λ to be small to achieve a quick starting convergence, then gradually increase its value to emphasize the constraint.

Algorithm 2: Stochastic mini-batch alternating direction penalty method to solve problem (17)

1: **Input** $\mathcal{D}_{\tau_i}^{tr}$, $\mathcal{D}_{\tau_i}^{val}$, $\delta_{tol} > 0$.
2: **Initialize** $\theta, \omega, \delta, \lambda > 0$ and $\nu_\delta \in (0,1),\ \nu_\lambda > 1$.
3: **while** $\delta > \delta_{tol}$ **do**
4: Sample cross-task training batch $\mathcal{B}^{tr} = \bigcup_{i=1}^{\mathcal{N}} \{(\mathbf{y}_j^{(i)}, \mathbf{x}*_j^{(i)}) \in \mathcal{D}_{\tau_i}^{tr}\}_{j=1:\mathcal{J}^{tr}}$
5: Sample cross-task validation batch $\mathcal{B}^{val} = \bigcup_{i=1}^{\mathcal{N}} \{(\mathbf{y}_j^{(i)}, \mathbf{x}*_j^{(i)}) \in \mathcal{D}_{\tau_i}^{val}\}_{j=1:\mathcal{J}^{val}}$
6: **while** $\|\nabla_\theta \widetilde{\mathcal{L}}(\theta, \omega; \mathcal{B}^{tr}, \mathcal{B}^{val})\|^2 + \|\nabla_\omega \widetilde{\mathcal{L}}(\theta, \omega; \mathcal{B}^{tr}, \mathcal{B}^{val})\|^2 > \delta$ **do**
7: **for** $k = 1, 2, \ldots, K$ (inner loop) **do**
8: $\theta \leftarrow \theta - \rho_\theta^k \nabla_\theta \widetilde{\mathcal{L}}(\theta, \omega; \mathcal{B}^{tr}, \mathcal{B}^{val})$
9: **end for**
10: $\omega \leftarrow \omega - \rho_\omega \nabla_\omega \widetilde{\mathcal{L}}(\theta, \omega; \mathcal{B}^{tr}, \mathcal{B}^{val})$
11: **end while**
12: **update** $\delta \leftarrow \nu_\delta \delta,\ \lambda \leftarrow \nu_\lambda \lambda$
13: **end while**
14: **output:** θ, ω.

4. Implementation

4.1. Feature Extraction Operator

We set the feature extraction operator $\mathbf{g}$ to be a vanilla l-layer CNN with the component-wise nonlinear activation function φ and no bias, as follows:

$$\mathbf{g}(x) = \mathbf{w}_l * \varphi \cdots \varphi(\mathbf{w}_3 * \varphi(\mathbf{w}_2 * \varphi(\mathbf{w}_1 * x))), \tag{19}$$

where $\{\mathbf{w}_q\}_{q=1}^{l}$ denote the convolution weights consisting of d kernels with identical spatial kernel size, and $*$ denotes the convolution operation. Here, φ is constructed to be the smoothed rectified linear unit as defined below:

$$\varphi(x) = \begin{cases} 0, & \text{if } x \leq -\delta, \\ \frac{1}{4\delta}x^2 + \frac{1}{2}x + \frac{\delta}{4}, & \text{if } -\delta < x < \delta, \\ x, & \text{if } x \geq \delta, \end{cases} \tag{20}$$

where the prefixed parameter δ is set to be 0.001 in our experiment. The default configuration of the feature extraction operator is set as follows: the feature extraction operator $\mathbf{g}$ consists of $l = 3$ convolution layers and all convolutions are with 4 kernels of a spatial size of 3×3.

4.2. Setups

As our method introduces an algorithmic unrolling network, there exists a one-to-one correspondence between the algorithm iterations and the neural network phases (or blocks). Each phase of the forward propagation can be viewed as one algorithm iteration, which motivates us to imitate the iterating of the optimization algorithm and use a stair training strategy [13]. At the first stage, we start training the network parameters using one phase, then after the the loss converges, we add more phases (one phase each time) then continue the training process. We repeat this procedure and stop it when the loss does not decrease

any further when we add more blocks. We minimize the loss for 100 epochs/iterations each time using the SGD-based optimizer Adam [59] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and the initial learning rate set to 10^{-3} as well as a mini-batch size of 8. The Xavier Initializer [60] is used to initialize the weights of all convolutions. The initial smoothing parameter ε_0 is set to be 0.001 and then learned together with other network parameters. The input $\mathbf{x}_0$ of the unrolling network is obtained by the zero-filling strategy [61]. The deep unrolling network was implemented using the Tensorflow toolbox [62] in the Python programming language.

5. Numerical Experiments

5.1. Dataset

To validate the performance of the proposed method, the data we used were from Multimodal Brain Tumor Segmentation Challenge 2018 [63], in which the training dataset contains four modalities (T1, $T1_c$, T2 and FLAIR)scanned from 285 patients and the validation dataset contains images from 66 patients, each with a volume size of $240 \times 240 \times 155$. Each modality consists of two types of gliomas: 75 volumes of low-grade gliomas (LGG) and 210 volumes of high-grade gliomas (HGG). Our implementation involved HGG MRI in two modalities—T1 and T2 images—and we chose 30 patients from each modality in the training dataset to train our network. In the validation dataset, we randomly picked 15 patients as our validation data and 6 patients in the training dataset as testing data, which were distinct from our training set and validation set. We cropped the 2D image size to be 160×180 in the center region and picked 10 adjacent slices in the center of each volume, resulting in a total of 300 images as our training data, 150 images as our validation data, and a total of 60 images as testing data. The amount of data mentioned here is for a single task, but since we employedy multi-task training, the total number of images in each dataset should be multiplied by the number of tasks. For each 2D slice, we normalized the spatial intensity by dividing the maximum pixel value.

5.2. Experiment Settings

All the experiments were implemented on a Windows workstation with an Intel Core i9 CPU at 3.3GHz and an Nvidia GTX-1080Ti GPU with 11 GB of graphics card memory via TensorFlow [64]. The parameters in the proposed network were initialized by using Xavier initialization [65]. We trained the meta-learning network with four tasks synergistically associated with four different CS ratios—10%, 20%, 30%, and 40%—and tested the well-trained model on the testing dataset with the same masks of these four ratios. We used 300 training data for each CS ratio, amounting to a total of 1200 images in the training dataset. The results for T1 and T2 MR reconstructions are shown in Tables 1 and 2, respectively. The associated reconstructed images are displayed in Figures 1 and 2. We also tested the well-trained meta-learning model on unseen tasks with radial masks for unseen ratios of 15%, 25%, and 35% and random Cartesian masks with ratios of 10%, 20%, 30%, and 40%. The task-specific parameters for the unseen tasks were retrained for different masks with different sampling ratios individually with fixed task-invariant parameters θ. In this experiments, we only needed to learn ω_i for three unseen CS ratios with radial mask and four regular CS ratios with Cartesian masks. The experimental training proceeded with fewer data and iterations, where we used 100 MR images with 50 epochs. For example, to reconstruct MR images with a CS ratio of 15% from the radial mask, we fixed the parameter θ and retrained the task-specific parameter ω on 100 raw data points with 50 epochs, then tested with renewed ω on our testing data set with raw measurements sampled from the radial mask with a CS radial of 15%. The results associated with radial masks are shown in Tables 3 and 4, Figures 3 and 4 for T1 and T2 images, respectively. The results associated with Cartesian masks are listed in Table 5 and reconstructed images are displayed in Figure 5.

We compared our proposed meta-learning method with conventional supervised learning, which was trained with one task at each time and only learned the task-invariant parameter θ without the task-specific parameter ω_i. The forward network of conventional

learning unrolled Algorithm 1 with 11 phases, which was the same as meta-learning. We merged the training set and validation set, resulting in 450 images for the training of the conventional supervised learning. The training batch size was set as 25 and we applied a total of 2000 epochs, while in meta-learning, we applied 100 epochs with a batch size of 8. The same testing set was used in both meta-learning and conventional learning to evaluate the performance of these two methods.

We made comparisons between meta-learning and the conventional network on the seven different CS ratios (10%, 20%, 30%, 40%, 15%, 25%, and 35%) in terms of two types of random under-sampling patterns: radial sampling mask and Cartesian sampling mask. The parameters for both meta-learning and conventional learning networks were trained via the Adam optimizer [59], and they both learned the forward unrolled task-invariant parameter θ. The network training of the conventional method used the same network configuration as the meta-learning network in terms of the number of convolutions, depth and size of CNN kernels, phase numbers and parameter initializer, etc. The major difference in the training process between these two methods is that meta-learning is performed for multi-tasks by leveraging the task-specific parameter ω_i learned from Algorithm 2, and the common features among tasks are learned from the feed-forward network that unrolls Algorithm 1, while conventional learning solves the task-specific problem by simply unrolling the forward network via Algorithm 1, where both training and testing are implemented on the same task. To investigate the generalizability of meta-learning, we tested the well-trained meta-learning model on MR images in different distributions in terms of two types of sampling masks with various trajectories. The training and testing of conventional learning were applied with the same CS ratios; that is, if the conventional method was trained with a CS ratio 10%, then it was also tested on a dataset with a CS ratio of 10%, etc.

Because MR images are represented as complex values, we applied complex convolutions [66] for each CNN; that is, every kernel consisted of a real part and imaginary part. Three convolutions were used in $\mathbf{g}$, where each convolution contained four filters with a spatial kernel size of 3×3. In Algorithm 1, a total of 11 phases can be achieved if we set the termination condition $\epsilon_{\text{tol}} = 1 \times 10^{-3}$, and the parameters of each phase are shared except for the step sizes. For the hyperparameters in Algorithm 1, we chose an initial learnable step size $\alpha_0 = 0.01, \tau_0 = 0.01, \varepsilon_0 = 0.001$, and we set prefixed values of $a = 10^5, \sigma = 10^3, \rho = 0.9$, and $\gamma = 0.9$. The principle behind the choices of those parameters is based on the convergence of the algorithm and effectiveness of the computation. The parameter $0 < \rho < 1$ is the reduction rate of the step size during the line search used to guarantee the convergence. The parameter $0 < \gamma < 1$ at step 15 is the reduction rate for ε. In Algorithm 1, from step 2 to step 14, the smoothing level ε is fixed. When the gradient of the smoothed function is small enough, we reduce ε by a fraction factor γ to find an approximate accumulation point of the original nonsmooth nonconvex problem. We chose a larger a in order to have more iterations k for which u_{k+1} satisfies the conditions in step 5, so that there would be fewer iterations requiring the computation of v_{k+1}. Moreover, the scheme for computing u_{k+1} is in accordance with the residual learning architecture that has been proven effective for reducing training error.

In Algorithm 2, we set $\nu_\delta = 0.95$ and the parameter δ was initialized as $\delta_0 = 1 \times 10^{-3}$ and stopped at value $\delta_{tol} = 4.35 \times 10^{-6}$, and a total of 100 epochs were performed. To train the conventional method, we set 2000 epochs with the same number of phases, convolutions, and kernel sizes as used to train the meta-learning approach. The initial λ was set as 1×10^{-5} and $\nu_\lambda = 1.001$.

We evaluated our reconstruction results on the testing data sets using three metrics: peak signal-to-noise ratio (PSNR) [67], structural similarity (SSIM) [68], and normalized mean squared error (NMSE) [69]. The following formulations compute the PSNR, SSIM, and NMSE between the reconstructed image $\mathbf{x}$ and ground truth $\mathbf{x}^*$. PSNR can be induced by the mean square error (MSE) where

$$PSNR(\mathbf{x}, \mathbf{x}^*) = 20 \log_{10} \left(\frac{\max(|\mathbf{x}^*|)}{\sqrt{MSE(\mathbf{x}, \mathbf{x}^*)}} \right), \tag{21}$$

where N is the total number of pixels of the ground truth and MSE is defined by $MSE(\mathbf{x}, \mathbf{x}^*) = \frac{1}{N}\|\mathbf{x}^* - \mathbf{x}\|^2$.

$$SSIM(\mathbf{x}, \mathbf{x}^*) = \frac{(2\mu_{\mathbf{x}}\mu_{\mathbf{x}^*} + C_1)(2\sigma_{\mathbf{x}\mathbf{x}^*} + C_2)}{(\mu_{\mathbf{x}}^2 + \mu_{\mathbf{x}^*}^2 + C_1)(\sigma_{\mathbf{x}}^2 + \sigma_{\mathbf{x}^*}^2 + C_2)}, \tag{22}$$

where $\mu_{\mathbf{x}}, \mu_{\mathbf{x}^*}$ represent local means, $\sigma_{\mathbf{x}}, \sigma_{\mathbf{x}^*}$ denote standard deviations, $\sigma_{\mathbf{x}\mathbf{x}^*}$ represents the covariance between $\mathbf{x}$ and $\mathbf{x}^*$, $C_1 = (k_1 L)^2, C_2 = (k_2 L)^2$ are two constants which avoid the zero denominator, and $k_1 = 0.01, k_2 = 0.03$. L is the largest pixel value of MR image.

$$NMSE(\mathbf{x}, \mathbf{x}^*) = \frac{\|\mathbf{x} - \mathbf{x}^*\|_2^2}{\|\mathbf{x}\|_2^2}, \tag{23}$$

where NMSE is used to measure the mean relative error. For detailed information of these three metrics mentioned above, please refer to [67–69].

5.3. Experimental Results with Different CS Ratios in Radial Mask

In this section, we evaluate the performance of well-trained meta-learning and conventional learning approaches. Tables 1, 2 and 5 report the quantitative results of averaged numerical performance with standard deviations and associated descaled task-specific meta-knowledge $\sigma(\omega_i)$. From the experiments implemented with radial masks, we observe that the average PSNR value of meta-learning improved by 1.54 dB in the T1 brain image for all four CS ratios compared with the conventional method, and for the T2 brain image, the average PSNR of meta-learning improved by 1.46 dB. Since the general setting of meta-learning aims to take advantage of the information provided from each individual task, with each task associated with an individual sampling mask that may have complemented sampled points, the performance of the reconstruction from each task benefits from other tasks. Smaller CS ratios will inhibit the reconstruction accuracy, due to the sparse undersampled trajectory in raw measurement, while meta-learning exhibits a favorable potential ability to solve this issue even in the situation of insufficient amounts of training data.

In general supervised learning, training data need to be in the same or a similar distribution; heterogeneous data exhibit different structural variations of features, which hinder CNNs from extracting features efficiently. In our experiments, raw measurements sampled from different ratios of compressed sensing display different levels of incompleteness; these undersampled measurements do not fall in the same distribution but they are related. Different sampling masks are shown at the bottom of Figures 1 and 3, and these may have complemented sampled points, in the sense that some of the points which a 40% sampling ratio mask did not sample were captured by other masks. In our experiment, different sampling masks provided their own information from their sampled points, meaning that four reconstruction tasks helped each other to achieve an efficient performance. Therefore, this explains why meta-learning is still superior to conventional learning when the sampling ratio is large.

Meta-learning expands a new paradigm for supervised learning—the purpose is to quickly learn multiple tasks. Meta-learning only learns task-invariant parameters once for a common feature that can be shared with four different tasks, and each $\sigma(\omega_i)$ provides task-specific weighting parameters according to the principle of "learning to learn". In conventional learning, the network parameter needs to be trained four times with four different masks since the task-invariant parameter cannot be generalized to other tasks, which is time-intensive. From Tables 1 and 2, we observe that a small CS ratio needs a higher value of $\sigma(\omega_i)$. In fact, in our model (11), the task-specific parameters behave as weighted constraints for task-specific regularizers, and the tables indicate that lower CS ratios require larger weights to be applied for the regularization.

A qualitative comparison between conventional and meta-learning methods is shown in Figures 1 and 2, displaying the reconstructed MR images of the same slice for T1 and T2, respectively. We label the zoomed-in details of HGG in the red boxes. We observe evidence

that conventional learning is more blurry and loses sharp edges, especially with lower CS ratios. From the point-wise error map, we find that meta-learning has the ability to reduce noises, especially in some detailed and complicated regions, compared to conventional learning.

We also tested the performance of meta-learning with two-thirds of the training and validation data for the T1-weighted image used in the previous experiment, denoted as "meta-learning". For conventional learning, the network was also trained by using two-thirds of the training samples in the previous experiment. The testing dataset remained the same as before. These results are displayed in Table 1, where we denote the reduced data experiments as "meta-learning*" and "conventional*". These experiments reveal that the accuracy of test data decreases when we reduce the training data size, but it is not a surprise that meta-learnining* still outperforms conventional learning*, and even conventional learning.

To verify the reconstruction performance of the proposed LOA 1, we compared the proposed conventional learning with ISTA-Net$^+$ [54], which is a state-of-the-art deep unfolded network for MRI reconstruction. We retrained ISTA-Net$^+$ with the same training dataset and testing dataset as conventional learning on the T1-weighted image. For a fair comparison, we used the same number of convolution kernels, the same dimension of kernels for each convolution during training, and the same phase numbers as conventional learning. The testing numerical results are listed in Table 1 and the MRI reconstructions are displayed in Figure 1. We can observe that the conventional learning which unrolls Algorithm 1 outperforms ISTA-Net$^+$ in any of the CS ratios. From the corresponding point-wise absolute error, the conventional learning attains a much lower error and much better reconstruction quality.

5.4. Experimental Results with Different Unseen CS Ratios in Different Sampling Patterns

In this section, we test the generalizability of the proposed model for unseen tasks. We fixed the well-trained task-invariant parameter θ and only trained ω_i for sampling ratios of 15%, 25%, and 35% with radial masks and sampling ratios of 10%, 20%, 30%, and 40% with Cartesian masks. In this experiment, we only used 100 training data points for each CS ratio and applied a total of 50 epochs. The averaged evaluation values and standard deviations are listed in Tables 3 and 4 for reconstructed T1 and T2 brain images, respectively, with radial masks, and Table 5 shows the qualitative performance for the reconstructed T2 brain image with random Cartesian sampling masks applied. In the T1 image reconstruction results, meta-learning showed an improvement of 1.6921 dB in PSNR for the 15% CS ratio, 1.6608 dB for the 25% CS ratio, and 0.5764 dB for the 35% ratio compared to the conventional method, showing the tendency that the level of reconstruction quality for lower CS ratios improved more than higher CS ratios. A similar trend was found for T2 reconstruction results with different sampling masks. The qualitative comparisons are illustrated in Figures 3–5 for T1 and T2 images tested with unseen CS ratios in radial masks and T2 images tested with Cartesian masks with regular CS ratios, respectively. In the experiments conducted with radial masks, meta-learning was superior to conventional learning, especially at a CS ratio of 15%—one can observe that the detailed regions in red boxes maintained their edges and were closer to the true image, while the conventional method reconstructions are hazier and lost details in some complicated tissues. The point-wise error map also indicates that meta-learning has the ability to suppress noises.

Training with Cartesian masks is more difficult than radial masks, especially for conventional learning, where the network is not very deep since the network only applies three convolutions each with four kernels. Table 5 indicates that the average performance of Meta-learning improved about 1.87 dB compared to conventional methods with T2 brain images. These results further demonstrate that meta-learning has the benefit of parameter efficiency, and the performance is much better than conventional learning even if we apply a shallow network with a small amount of training data.

The numerical experimental results discussed above show that meta-learning is capable of fast adaption to new tasks and has more robust generalizability for a broad range of tasks with heterogeneous, diverse data. Meta-learning can be considered as an efficient technique for solving difficult tasks by leveraging the features extracted from easier tasks.

Table 1. Quantitative evaluations of the reconstructions of T1 brain image associated with various sampling ratios of **radial** masks. Conventional* and meta-learning* are trained with two-thirds of the dataset used in training conventional and meta-learning approaches, respectively.

CS Ratio	Methods	PSNR	SSIM	NMSE	$\sigma(\omega_i)$
10%	ISTA-Net+ [54]	21.2633 ± 1.0317	0.5487 ± 0.0440	0.1676 ± 0.0253	
	Conventional*	21.6947 ± 1.0264	0.5689 ± 0.0404	0.1595 ± 0.0240	
	Conventional	21.7570 ± 1.0677	0.5650 ± 0.0412	0.0259 ± 0.0082	
	Meta-learning*	22.9633 ± 1.0969	0.5962 ± 0.0415	0.0194 ± 0.0065	0.9339
	Meta-learning	23.2672 ± 1.1229	0.6101 ± 0.0436	0.0184 ± 0.0067	0.9218
20%	ISTA-Net+ [54]	26.2734 ± 1.0115	0.7068 ± 0.0364	0.0944 ± 0.0155	
	Conventional*	26.4639 ± 1.0233	0.7107 ± 0.0357	0.0924 ± 0.0154	
	Conventional	26.6202 ± 1.1662	0.7121 ± 0.0397	0.0910 ± 0.0169	
	Meta-learning*	27.9381 ± 1.1121	0.7541 ± 0.0360	0.0063 ± 0.0023	0.8150
	Meta-learning	28.2944 ± 1.2119	0.7640 ± 0.0377	0.0058 ± 0.0022	0.7756
30%	ISTA-Net+ [54]	28.8309 ± 1.3137	0.7492 ± 0.0407	0.0708 ± 0.0142	
	Conventional*	29.2923 ± 1.3194	0.7522 ± 0.0399	0.0671 ± 0.0136	
	Conventional	29.5034 ± 1.4446	0.7557 ± 0.0408	0.0657 ± 0.0143	
	Meta-learning*	30.8691 ± 1.5897	0.8310 ± 0.0394	0.0033 ± 0.0015	0.6359
	Meta-learning	31.1417 ± 1.5866	0.8363 ± 0.0385	0.0031 ± 0.0014	0.6501
40%	ISTA-Net+ [54]	30.7282 ± 1.5482	0.8008 ± 0.0428	0.0572 ± 0.0127	
	Conventional*	31.3761 ± 1.5892	0.8035 ± 0.0420	0.0532 ± 0.0121	
	Conventional	31.4672 ± 1.6390	0.8111 ± 0.0422	0.0029 ± 0.0014	
	Meta-learning*	32.7330 ± 1.6386	0.8623 ± 0.0358	0.0022 ± 0.0010	0.6639
	Meta-learning	32.8238 ± 1.7039	0.8659 ± 0.0370	0.0022 ± 0.0010	0.6447

Table 2. Quantitative evaluations of the reconstructions of T2 brain image associated with various sampling ratios of **radial** masks.

CS Ratio	Methods	PSNR	SSIM	NMSE	$\sigma(\omega_i)$
10%	Conventional	23.0706 ± 1.2469	0.5963 ± 0.0349	0.2158 ± 0.0347	
	Meta-learning	24.0842 ± 1.3863	0.6187 ± 0.0380	0.0112 ± 0.0117	0.9013
20%	Conventional	27.0437 ± 1.0613	0.6867 ± 0.0261	0.1364 ± 0.0213	
	Meta-learning	28.9118 ± 1.0717	0.7843 ± 0.0240	0.0122 ± 0.0030	0.8742
30%	Conventional	29.5533 ± 1.0927	0.7565 ± 0.0265	0.1023 ± 0.0166	
	Meta-learning	31.4096 ± 0.9814	0.8488 ± 0.0217	0.0069 ± 0.0019	0.8029
40%	Conventional	32.0153 ± 0.9402	0.8139 ± 0.0238	0.0770 ± 0.0128	
	Meta-learning	33.1114 ± 1.0189	0.8802 ± 0.0210	0.0047 ± 0.0015	0.7151

Table 3. Quantitative evaluations of the reconstructions of T1 brain image associated with various sampling ratios of **radial** masks. Meta-learning was trained with CS ratios of 10%, 20%, 30%, and 40% and tested with unseen ratios of 15%, 25%, and 35%. The conventional method was subjected to regular training and testing with the same CS ratios of 15%, 25%, and 35%.

CS Ratio	Methods	PSNR	SSIM	NMSE	$\sigma(\omega_i)$
15%	Conventional	24.6573 ± 1.0244	0.6339 ± 0.0382	0.1136 ± 0.0186	
	Meta-learning	26.3494 ± 1.0102	0.7088 ± 0.0352	0.0090 ± 0.0030	0.9429
25%	Conventional	28.4156 ± 1.2361	0.7533 ± 0.0368	0.0741 ± 0.0141	
	Meta-learning	30.0764 ± 1.4645	0.8135 ± 0.0380	0.0040 ± 0.0017	0.8482
35%	Conventional	31.5320 ± 1.5242	0.7923 ± 0.0420	0.0521 ± 0.0119	
	Meta-learning	32.1084 ± 1.6481	0.8553 ± 0.0379	0.0025 ± 0.0011	0.6552

Table 4. Quantitative evaluations of the reconstructions of T2 brain image associated with various sampling ratios of **radial** masks. Meta-learning was trained with CS ratios of 10%, 20%, 30%, and 40% and tested with unseen ratios of 15%, 25%, and 35%. Conventional method was subjected to regular training and testing with the same CS ratios of 15%, 25%, and 35%.

CS Ratio	Methods	PSNR	SSIM	NMSE	$\sigma(\omega_i)$
15%	Conventional	24.8921 ± 1.2356	0.6259 ± 0.0285	0.1749 ± 0.0280	
	Meta-learning	26.7031 ± 1.2553	0.7104 ± 0.0318	0.0205 ± 0.0052	0.9532
25%	Conventional	29.0545 ± 1.1980	0.7945 ± 0.0292	0.1083 ± 0.0173	
	Meta-learning	30.0698 ± 0.9969	0.8164 ± 0.0235	0.0093 ± 0.0022	0.8595
35%	Conventional	31.5201 ± 1.0021	0.7978 ± 0.0236	0.0815 ± 0.0129	
	Meta-learning	32.0683 ± 0.9204	0.8615 ± 0.0209	0.0059 ± 0.0014	0.7388

Table 5. Quantitative evaluations of the reconstructions of T2 brain image associated with various sampling ratios of random **Cartesian** masks.

CS Ratio	Methods	PSNR	SSIM	NMSE	$\sigma(\omega_i)$
10%	Conventional	20.8867 ± 1.2999	0.5082 ± 0.0475	0.0796 ± 0.0242	
	Meta-learning	22.0434 ± 1.3555	0.6279 ± 0.0444	0.0611 ± 0.0188	0.9361
20%	Conventional	22.7954 ± 1.2819	0.6057 ± 0.0412	0.0513 ± 0.0157	
	Meta-learning	24.7162 ± 1.3919	0.6971 ± 0.0380	0.0329 ± 0.0101	0.8320
30%	Conventional	24.2170 ± 1.2396	0.6537 ± 0.0360	0.0371 ± 0.0117	
	Meta-learning	26.4537 ± 1.3471	0.7353 ± 0.0340	0.0221 ± 0.0068	0.6771
40%	Conventional	25.3668 ± 1.3279	0.6991 ± 0.0288	0.1657 ± 0.0265	
	Meta-learning	27.5367 ± 1.4107	0.7726 ± 0.0297	0.0171 ± 0.0050	0.6498

J. Imaging **2021**, *1*, 0 16 of 29

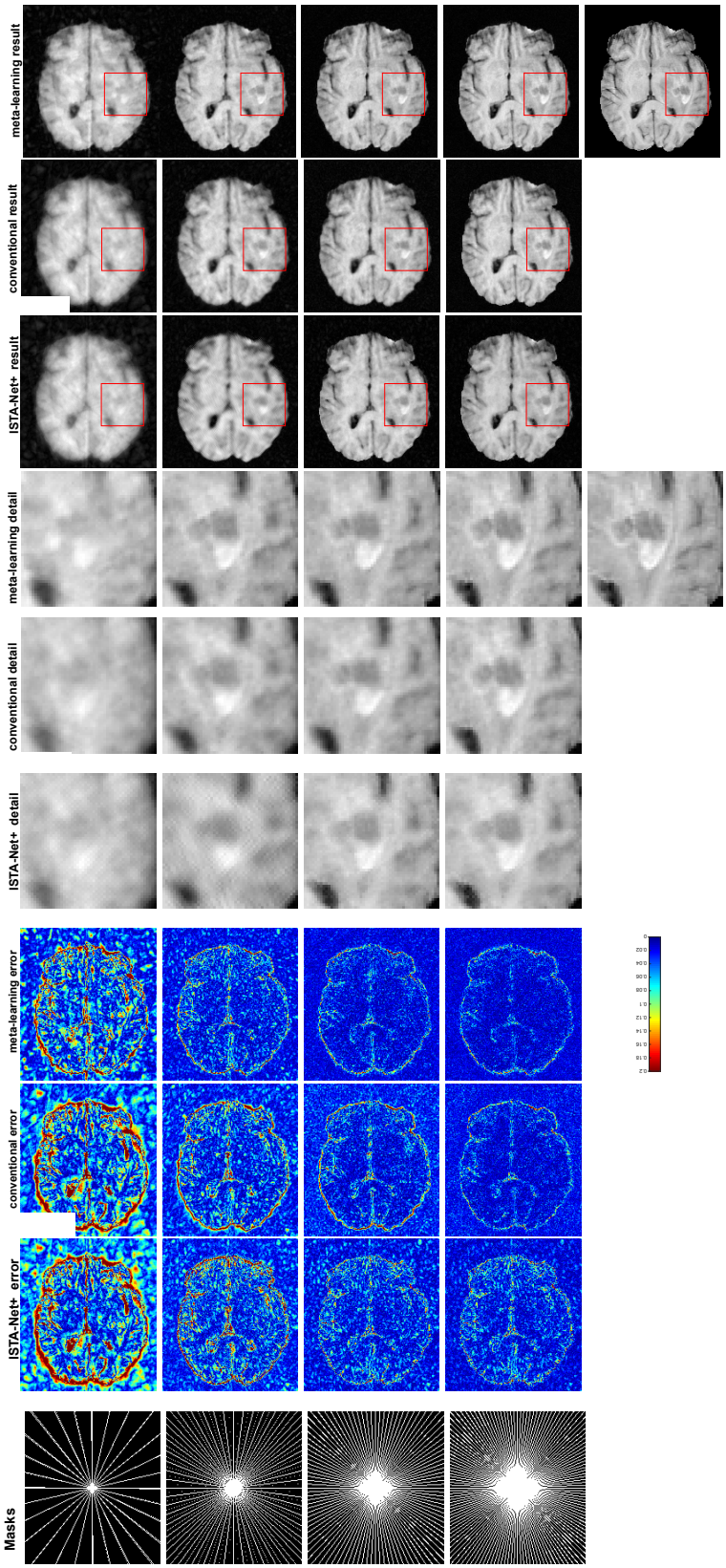

Figure 1. The pictures (from top to bottom) display the reconstruction results, zoomed-in details, point-wise errors with a color bar, and associated **radial** masks for meta-learning, conventional learning, and ISTA-Net$^+$ with four different CS ratios of 10%, 20%, 30%, 40% (from left to right). The top-right image is the ground truth fully-sampled image.

Figure 1. The pictures (from top to bottom) display the reconstruction results, zoomed-in details, point-wise errors with a color bar, and associated **radial** masks for meta-learning, conventional learning, and ISTA-Net$^+$ with four different CS ratios of 10%, 20%, 30%, 40% (from left to right). The top-right image is the ground truth fully-sampled image.

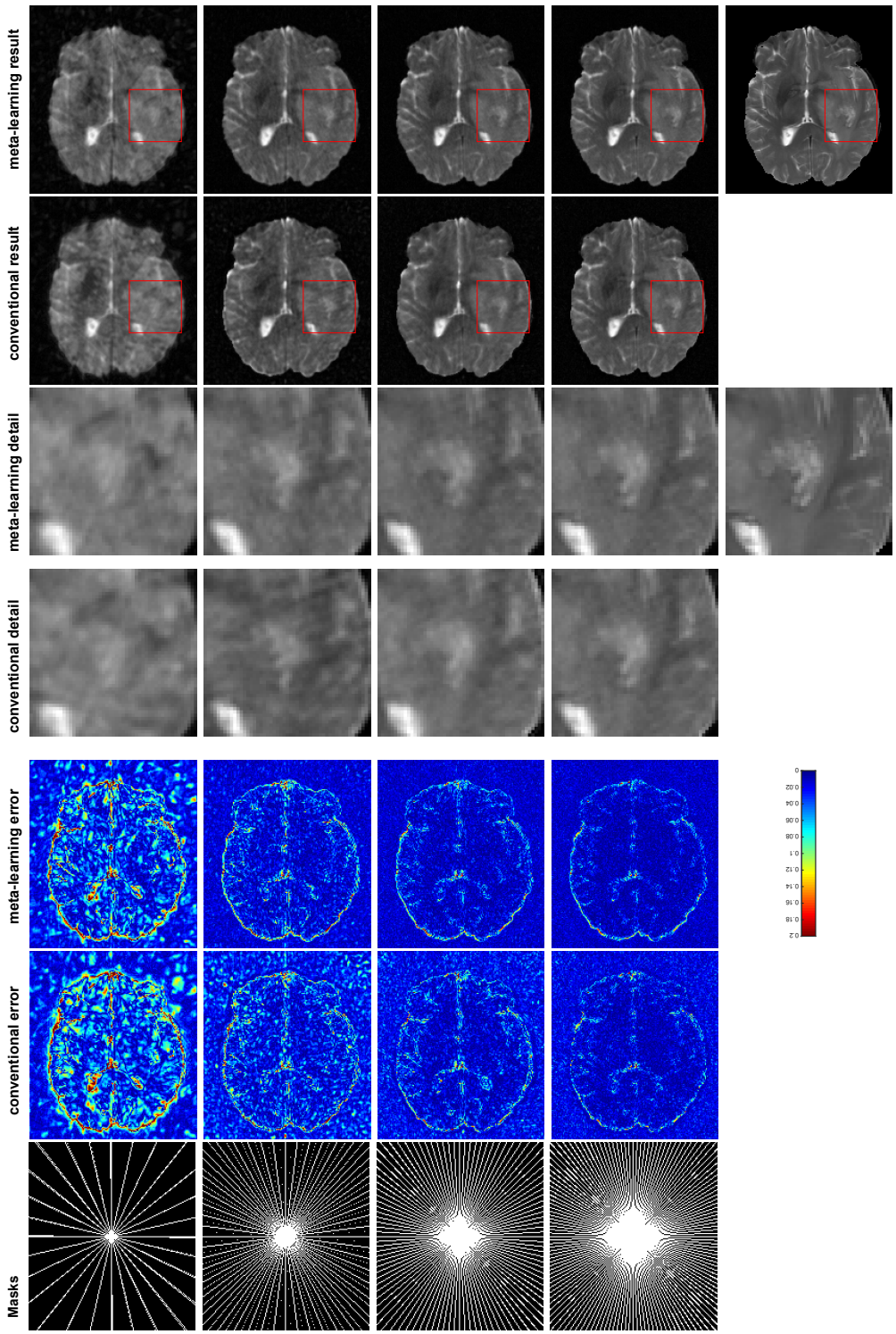

Figure 2. The pictures (from top to bottom) display the T2 brain image reconstruction results, zoomed-in details, point-wise errors with a color bar, and associated **radial** masks for meta-learning and conventional learningwith four different CS ratios of 10%, 20%, 30%, 40% (from left to right). The top-right iage is the ground truth fully-sampled image.

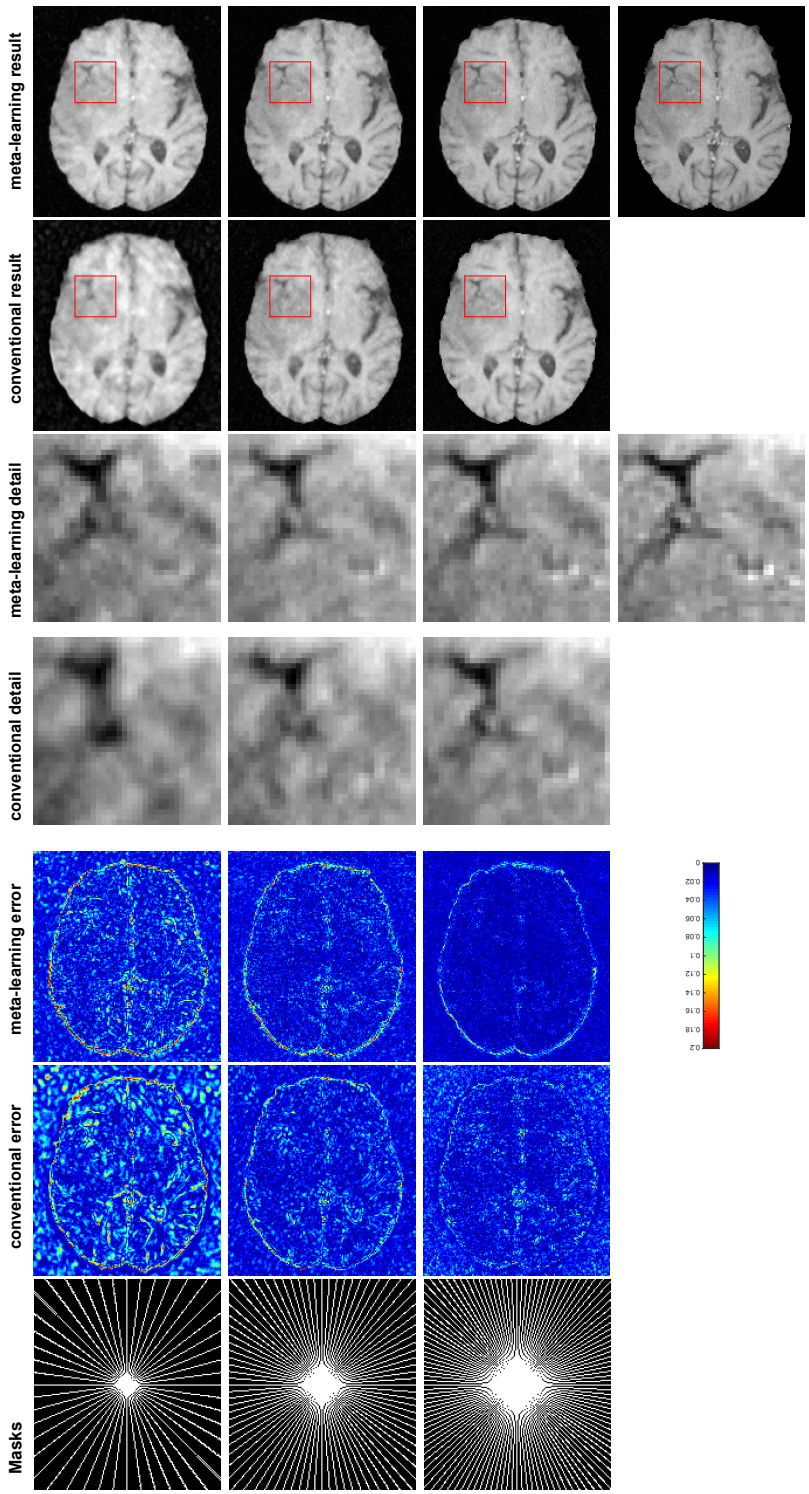

Figure 3. The pictures (from top to bottom) display the T1 brain image reconstruction results, zoomed-in details, point-wise errors with a color bar, and associated **radial** masks for meta-learning and conventional learning. Meta-learning was trained with CS ratios of 10%, 20%, 30%, and 40% and tested with three different unseen CS ratios of 15%, 25%, and 35% (from left to right). Conventional learning was trained and tested with the same CS ratios of 15%, 25%, and 35%. The top-right image is the ground truth fully-sampled image.

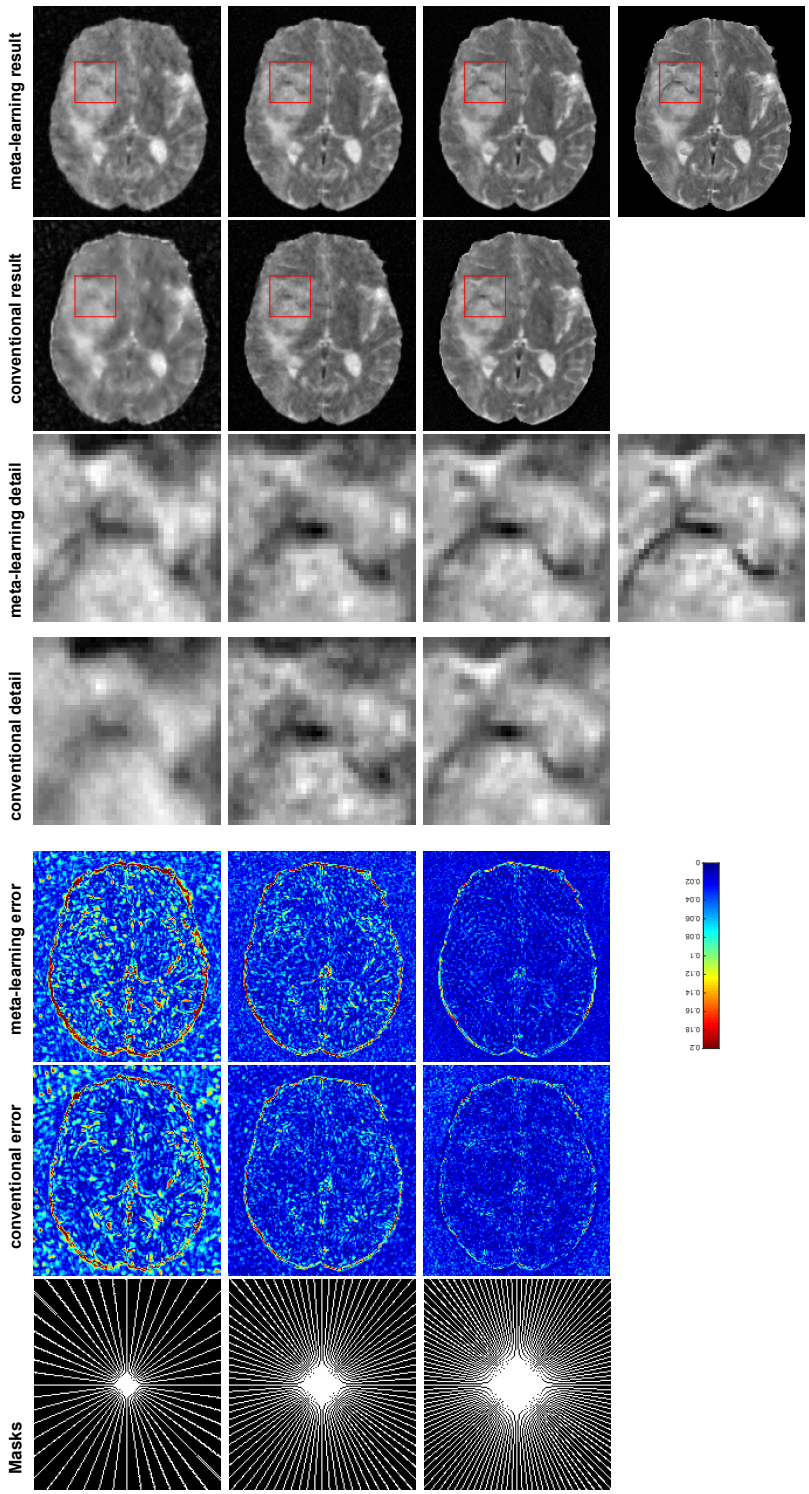

Figure 4. The pictures (from top to bottom) display the T2 brain image reconstruction results, zoomed-in details, point-wise errors with a color bar, and associated **radial** masks for meta-learning and conventional learning. Meta-learning was trained with CS ratios of 10%, 20%, 30%, and 40% and test edwith three different unseen CS ratios of 15%, 25%, and 35% (from left to right). Conventional learning was trained and tested with the same CS ratios of 15%, 25%, and 35%. The top-right image is the ground truth fully-sampled image.

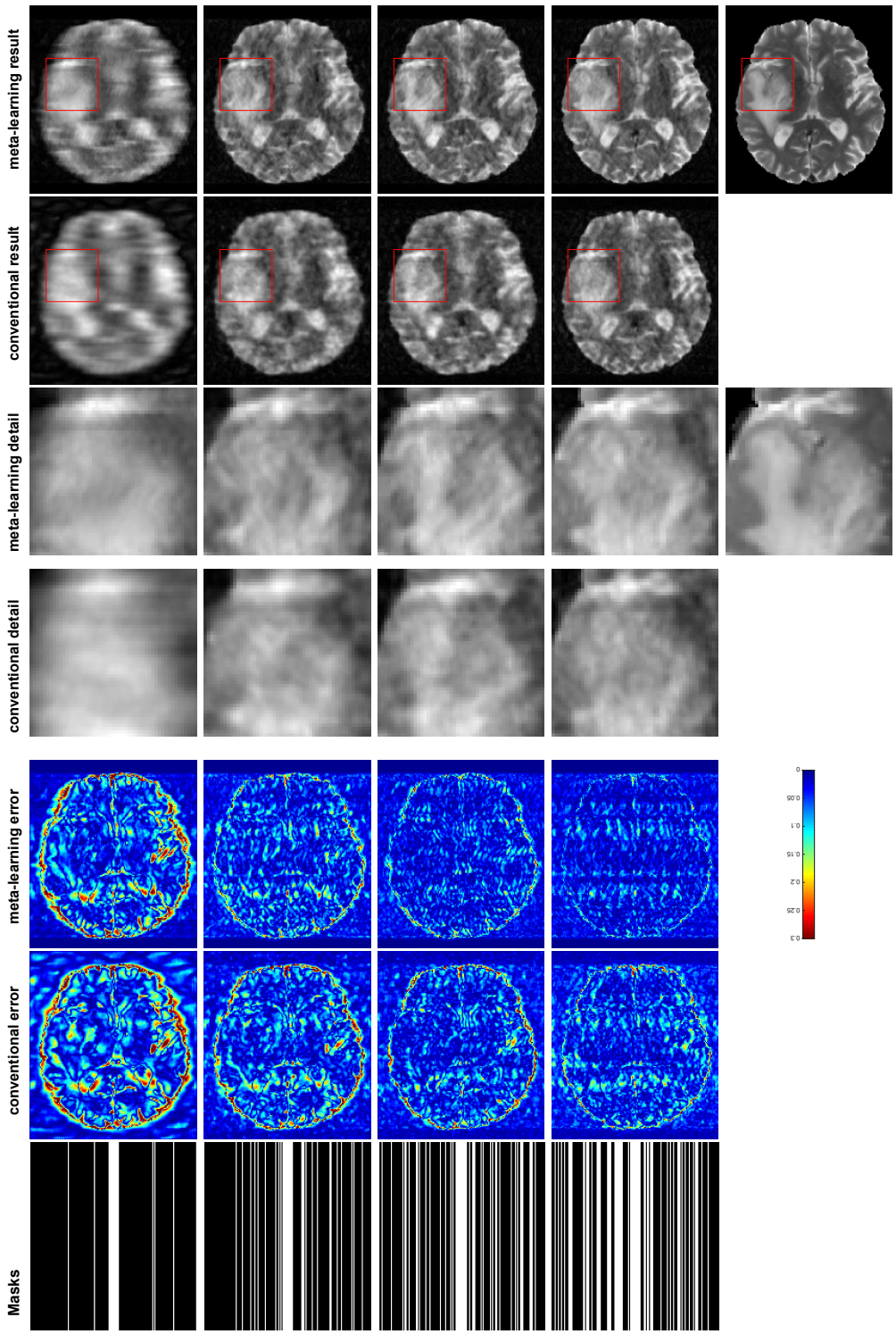

Figure 5. The pictures (from top to bottom) display the T2 brain image reconstruction results, zoomed-in details, point-wise errors with a color bar, and associated **Cartesian** masks for meta-learning and conventional learningwith four different CS ratios of 10%, 20%, 30%, and 40% (from left to right). The top-right image is the ground truth fully-sampled image.

Next, we empirically demonstrate the convergence of Algorithm 1 in Figure 6. This shows that the objective function value ϕ decreases and the PSNR value for testing data increases steadily as the number of phases increases, which indicates that the learned algorithm is indeed minimizing the learned function as we desired.

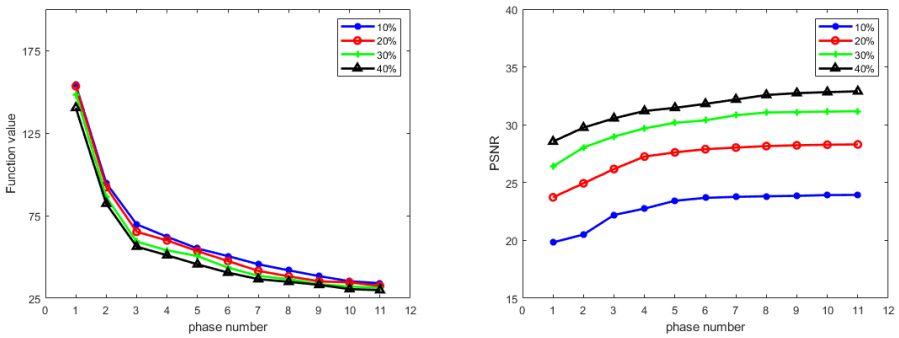

Figure 6. Convergence behavior of Algorithm 1 on T1 weighted MRI image reconstruction with four different CS ratios using radial mask. **Left**: Objective function value ϕ versus phase number. **Right**: PSNR value versus phase number.

5.5. Future Work and Open Challenges

Deep optimization-based meta-learning techniques have shown great generalizability, but there are several open challenges that can be discussed and can potentially be addressed in future work. A major issue is the memorization problem, since the base learner needs to be optimized for a large number of phases and the training algorithm contains multiple gradient steps; furthermore, the computation is very expensive in terms of time and memory costs. In addition to reconstructing MRI through different trajectories, another potential application for medical imaging could be multi-modality reconstruction and synthesis. Capturing images of anatomy with multi-modality acquisitions enhances the diagnostic information and could be cast as a multi-task problem that could benefit from meta-learning.

6. Conclusions

In this paper, we put forward a novel deep model for MRI reconstructions via meta-learning. The proposed method has the ability to solve multi-tasks synergistically, and the well-trained model could generalize well to new tasks. Our baseline network is constructed by unfolding an LOA, which inherits the convergence property, improves the interpretability, and promotes the parameter efficiency of the designed network structure. The designated adaptive regularizer consists of a task-invariant learner and task-specific meta-knowledge. Network training follows a bilevel optimization algorithm that minimizes task-specific parameter ω in the upper level for the validation data and minimizes task-invariant parameters θ in the lower level for the training data with fixed ω. The proposed approach is the first model designed to solve the inverse problem by applying meta-training on the adaptive regularization in the variational model. We consider the recovery of undersampled raw data across different sampling trajectories with various sampling patterns as different tasks. Extensive numerical experiments on various MRI datasets demonstrate that the proposed method generalizes well at various sampling trajectories and is capable of fast adaption to unseen trajectories and sampling patterns. The reconstructed images achieve higher quality compared to conventional supervised learning for both seen and unseen k-space trajectory cases.

Author Contributions: All authors are contributed equally. Conceptualization, W.B., Y.C., X.Y. and Q.Z.; methodology, W.B., Y.C., X.Y. and Q.Z.; software, W.B. and Q.Z.; validation, Y.C. and X.Y.; formal analysis, W.B., Y.C., X.Y. and Q.Z.; investigation, W.B., Y.C., X.Y. and Q.Z.; resources, Y.C.; data curation, W.B. and Q.Z.; writing—original draft preparation, W.B. and Q.Z.; writing—review and editing, X.Y. and Y.C.; visualization, X.Y. and Y.C.; supervision, Y.C.; project administration, Y.C.; funding acquisition, X.Y. and Y.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by NSF grants DMS-1319050, DMS-1719932, DMS-1818886, DMS-1925263 and University of Florida AI Catalyst Grants.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: The data presented in this study are available on Multimodal Brain Tumor Segmentation Challenge (BraTs) 2018 [63] at https://www.med.upenn.edu/sbia/brats2018/data.html (accessed on 29 October 2021).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Convergence Analysis

We make the following assumptions regarding f and $\mathbf{g}$ throughout this work:

- (a1): f is differentiable and (possibly) nonconvex, and ∇f is L_f-Lipschitz continuous.
- (a2): Every component of $\mathbf{g}$ is differentiable and (possibly) nonconvex, and $\nabla \mathbf{g}$ is L_g-Lipschitz continuous.
- (a3): $\sup_{\mathbf{x}\in\mathcal{X}} \|\nabla \mathbf{g}(\mathbf{x})\| \leq M$ for some constant $M > 0$.
- (a4): ϕ is coercive, and $\phi^* = \min_{\mathbf{x}\in\mathcal{X}} \phi(\mathbf{x}) > -\infty$.

First, we state the Clark subdifferential [13] of $r(\mathbf{x})$ in Lemma A1, and we show that the gradient of r_ε is Lipschitz continuous in Lemma A2.

Lemma A1. *Let $r(\mathbf{x})$ be defined in (4); then, the Clarke subdifferential of r at $\mathbf{x}$ is*

$$\partial r(\mathbf{x}) = \{ \sum_{j\in I_0} \nabla \mathbf{g}_i(\mathbf{x})^\top \mathbf{w}_j + \sum_{j\in I_1} \nabla \mathbf{g}_j(\mathbf{x})^\top \frac{\mathbf{g}_j(\mathbf{x})}{\|\mathbf{g}_j(\mathbf{x})\|} \;\Big|\; \mathbf{w}_j \in \mathbb{R}^d,\ \|\Pi(\mathbf{w}_j; \mathcal{C}(\nabla \mathbf{g}_i(\mathbf{x})))\| \leq 1,\ \forall\, j \in I_0 \}, \tag{A1}$$

where $I_0 = \{j \in [m] \mid \|\mathbf{g}_j(\mathbf{x})\| = 0\}$, $I_1 = [m] \setminus I_0$, and $\Pi(\mathbf{w}; \mathcal{C}(\mathbf{A}))$ is the projection of $\mathbf{w}$ onto $\mathcal{C}(\mathbf{A})$ which stands for the column space of $\mathbf{A}$.

Lemma A2. *The gradient of r_ε is Lipschitz continuous with constant $m(L_g + \frac{2M^2}{\varepsilon})$.*

Proof. From $r_\varepsilon(\mathbf{x}) = \sum_{j=1}^m (\|\mathbf{g}_j(\mathbf{x})\|^2 + \varepsilon^2)^{\frac{1}{2}} - \varepsilon$, it follows that

$$\nabla r_\varepsilon(\mathbf{x}) = \sum_{j=1}^m \nabla \mathbf{g}_j(\mathbf{x})^\top \mathbf{g}_j(\mathbf{x}) (\|\mathbf{g}_j(\mathbf{x})\|^2 + \varepsilon^2)^{-\frac{1}{2}}. \tag{A2}$$

For any $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}$, we first define $h(\mathbf{x}) = \mathbf{g}_j(\mathbf{x})(\|\mathbf{g}_j(\mathbf{x})\|^2 + \varepsilon^2)^{-\frac{1}{2}}$, so $\|h(\mathbf{x})\| < 1$.

$$\|h(\mathbf{x}_1) - h(\mathbf{x}_2)\| \tag{A3a}$$
$$= \left\| \frac{\mathbf{g}_j(\mathbf{x}_1)}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2 + \varepsilon^2}} - \frac{\mathbf{g}_j(\mathbf{x}_2)}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2}} \right\| \tag{A3b}$$
$$= \left\| \frac{\mathbf{g}_j(\mathbf{x}_1)}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2 + \varepsilon^2}} - \frac{\mathbf{g}_j(\mathbf{x}_1)}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2}} + \frac{\mathbf{g}_j(\mathbf{x}_1)}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2}} - \frac{\mathbf{g}_j(\mathbf{x}_2)}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2}} \right\| \tag{A3c}$$
$$\leq \left\| \mathbf{g}_j(\mathbf{x}_1) \left(\frac{\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2} - \sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2 + \varepsilon^2}}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2 + \varepsilon^2}\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2}} \right) \right\| + \left\| \frac{\mathbf{g}_j(\mathbf{x}_1) - \mathbf{g}_j(\mathbf{x}_2)}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2}} \right\| \tag{A3d}$$
$$\leq \left\| \frac{\mathbf{g}_j(\mathbf{x}_1)}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2 + \varepsilon^2}} \right\| \left\| \frac{\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2} - \sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2 + \varepsilon^2}}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2}} \right\| + \frac{1}{\varepsilon}\|\mathbf{g}_j(\mathbf{x}_1) - \mathbf{g}_j(\mathbf{x}_2)\| \tag{A3e}$$
$$\leq \frac{1}{\varepsilon}\left\| \sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2} - \sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2 + \varepsilon^2} \right\| + \frac{1}{\varepsilon}\|\mathbf{g}_j(\mathbf{x}_1) - \mathbf{g}_j(\mathbf{x}_2)\| \tag{A3f}$$
$$\leq \frac{1}{\varepsilon} \frac{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 - \|\mathbf{g}_j(\mathbf{x}_1)\|^2}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2} + \sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2 + \varepsilon^2}} + \frac{1}{\varepsilon}\|\mathbf{g}_j(\mathbf{x}_1) - \mathbf{g}_j(\mathbf{x}_2)\| \tag{A3g}$$
$$\leq \frac{1}{\varepsilon} \underbrace{\frac{\|\mathbf{g}_j(\mathbf{x}_2)\| + \|\mathbf{g}_j(\mathbf{x}_1)\|}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_2)\|^2 + \varepsilon^2} + \sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2 + \varepsilon^2}}}_{<1} (\|\mathbf{g}_j(\mathbf{x}_2)\| - \|\mathbf{g}_j(\mathbf{x}_1)\|) + \frac{1}{\varepsilon}\|\mathbf{g}_j(\mathbf{x}_1) - \mathbf{g}_j(\mathbf{x}_2)\| \tag{A3h}$$
$$\leq \frac{1}{\varepsilon}\|\mathbf{g}_j(\mathbf{x}_2) - \mathbf{g}_j(\mathbf{x}_1)\| + \frac{1}{\varepsilon}\|\mathbf{g}_j(\mathbf{x}_1) - \mathbf{g}_j(\mathbf{x}_2)\| \tag{A3i}$$
$$= \frac{2}{\varepsilon}\|\mathbf{g}_j(\mathbf{x}_1) - \mathbf{g}_j(\mathbf{x}_2)\|. \tag{A3j}$$

where to obtain (A3f) we used $\left\| \frac{\mathbf{g}_j(\mathbf{x}_1)}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2+\varepsilon^2}} \right\| < 1$ and $\frac{1}{\sqrt{\|\mathbf{g}_j(\mathbf{x}_1)\|^2+\varepsilon^2}} < \frac{1}{\varepsilon}$.

Therefore, we have

$$\|\nabla r_\varepsilon(\mathbf{x}_1) - \nabla r_\varepsilon(\mathbf{x}_2)\| \tag{A4a}$$
$$= \sum_{j=1}^{m} \left\| \nabla \mathbf{g}_j(\mathbf{x}_1)^\top h(\mathbf{x}_1) - \nabla \mathbf{g}_j(\mathbf{x}_2)^\top h(\mathbf{x}_2) \right\| \tag{A4b}$$
$$= \sum_{j=1}^{m} \left\| \nabla \mathbf{g}_j(\mathbf{x}_1)^\top h(\mathbf{x}_1) - \nabla \mathbf{g}_j(\mathbf{x}_2)^\top h(\mathbf{x}_1) + \nabla \mathbf{g}_j(\mathbf{x}_2)^\top h(\mathbf{x}_1) - \nabla \mathbf{g}_j(\mathbf{x}_2)^\top h(\mathbf{x}_2) \right\| \tag{A4c}$$
$$\leq \sum_{j=1}^{m} \left\| \left(\nabla \mathbf{g}_j(\mathbf{x}_1) - \nabla \mathbf{g}_j(\mathbf{x}_2)\right)^\top h(\mathbf{x}_1) \right\| + \left\| \nabla \mathbf{g}_j(\mathbf{x}_2)(h(\mathbf{x}_1) - h(\mathbf{x}_2)) \right\| \tag{A4d}$$
$$\leq \sum_{j=1}^{m} \left\| \nabla \mathbf{g}_j(\mathbf{x}_1) - \nabla \mathbf{g}_j(\mathbf{x}_2) \right\| \|h(\mathbf{x}_1)\| + \|\nabla \mathbf{g}_j(\mathbf{x}_2)\| \|h(\mathbf{x}_1) - h(\mathbf{x}_2)\|$$
$$\leq \sum_{j=1}^{m} \left\| \nabla \mathbf{g}_j(\mathbf{x}_1) - \nabla \mathbf{g}_j(\mathbf{x}_2) \right\| + \|\nabla \mathbf{g}_j(\mathbf{x}_2)\| \frac{2}{\varepsilon} \|\mathbf{g}_j(\mathbf{x}_1) - \mathbf{g}_j(\mathbf{x}_2)\| \text{ by (A3) and} \tag{A4e}$$
$$\leq m(L_g\|\mathbf{x}_1 - \mathbf{x}_2\| + M\frac{2}{\varepsilon} \cdot M\|\mathbf{x}_1 - \mathbf{x}_2\|), \tag{A4f}$$

where the first term of the last inequality is due to the L_g-Lipschitz continuity of $\nabla \mathbf{g}_j$. The second term is because of $\|\mathbf{g}_j(\mathbf{x}_1) - \mathbf{g}_j(\mathbf{x}_2)\| = \|\nabla \mathbf{g}_j(\tilde{\mathbf{x}})(\mathbf{x}_1 - \mathbf{x}_2)\|$ for some $\tilde{\mathbf{x}} \in \mathcal{X}$ due to the mean value theorem and $\|\nabla \mathbf{g}_j(\tilde{\mathbf{x}})\| \leq \sup_{\mathbf{x}\in\mathcal{X}} \|\nabla \mathbf{g}_j(\mathbf{x})\| \leq M$. Therefore, we obtain

$$\|\nabla r_\varepsilon(\mathbf{x}_1) - \nabla r_\varepsilon(\mathbf{x}_2)\| \leq m(L_g + \frac{2M^2}{\varepsilon})\|\mathbf{x}_1 - \mathbf{x}_2\|. \tag{A5}$$

□

Lemma A3. *Let $\varepsilon, \eta, \tau_t, a > 0$, $0 < \rho < 1$, and choose the initial $\mathbf{x}_0 \in \mathcal{X}$. Suppose the sequence $\{\mathbf{x}_t\}$ is generated by executing Lines 3–14 of Algorithm 1 with fixed $\varepsilon_t = \varepsilon$ and that $0 < \delta < \frac{L_\varepsilon}{2/a+L_\varepsilon} < 1$ exists such that $\alpha_t \geq \frac{\delta}{L_\varepsilon} > 0$, where $L_\varepsilon = L_f + L_h + mL_g + \frac{2M^2}{\varepsilon}$ and $\phi^* := \min_{\mathbf{x}\in\mathcal{X}} \phi(\mathbf{x})$. Then, the following statements hold:*

1. $\|\nabla \phi_\varepsilon(\mathbf{x}_t)\| \to 0$ *as* $t \to \infty$.
2. $\max\{t \in \mathbb{N} \mid \|\nabla \phi_\varepsilon(\mathbf{x}_{t+1})\| \geq \eta\} \leq \max\{\frac{aL_\varepsilon^2}{\delta^2}, a^3\}(\phi_\varepsilon(\mathbf{x}_0) - \phi^* + \varepsilon)$.

Proof. 1. In each iteration, we compute $\mathbf{u}_{t+1} = \mathbf{z}_{t+1} - \tau_t \sigma(\omega_i)\nabla r_{\varepsilon_t}(\mathbf{z}_{t+1})$.

1.1. In the case the condition

$$\|\nabla \phi_\varepsilon(\mathbf{x}_t)\| \leq a\|\mathbf{u}_{t+1} - \mathbf{x}_t\| \quad \text{and} \quad \phi_\varepsilon(\mathbf{u}_{t+1}) - \phi_\varepsilon(\mathbf{x}_t) \leq -\frac{1}{a}\|\mathbf{u}_{t+1} - \mathbf{x}_t\|^2 \tag{A6}$$

holds with $a > 0$, we put $\mathbf{x}_{t+1} = \mathbf{u}_{t+1}$, and we have $\phi_\varepsilon(\mathbf{u}_{t+1}) \leq \phi_\varepsilon(\mathbf{x}_t)$.

1.2. Otherwise, we compute $\mathbf{v}_{t+1} = \mathbf{x}_t - \alpha_t \nabla \phi_\varepsilon(\mathbf{x}_t)$, where α_t is found through the line search until the criteria

$$\phi_\varepsilon(\mathbf{v}_{t+1}) - \phi_\varepsilon(\mathbf{x}_t) \leq -\frac{1}{a}\|\mathbf{v}_{t+1} - \mathbf{x}_t\|^2 \tag{A7}$$

holds, and then put $\mathbf{x}_{t+1} = \mathbf{v}_{t+1}$. From Lemma A2, we know that the gradient $\nabla r_\varepsilon(\mathbf{x})$ is Lipschitz continuous with constant $m(L_g + \frac{2M^2}{\varepsilon})$. Furthermore, we assumed in (a1) that ∇f is L_f-Lipschitz continuous. Hence, putting $L_\varepsilon = L_f + m(L_g + \frac{2M^2}{\varepsilon})$, we find that $\nabla \phi_\varepsilon$ is L_ε-Lipschitz continuous, which implies

$$\phi_\varepsilon(\mathbf{v}_{t+1}) \leq \phi_\varepsilon(\mathbf{x}_t) + \langle \nabla \phi_\varepsilon(\mathbf{x}_t), \mathbf{v}_{t+1} - \mathbf{x}_t \rangle + \frac{L_\varepsilon}{2}\|\mathbf{v}_{t+1} - \mathbf{x}_t\|^2. \tag{A8}$$

Furthermore, by the optimality condition of

$$\mathbf{v}_{t+1} = \arg\min_{\mathbf{x}} \langle \nabla f(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle + \sigma(\omega_i)\langle \nabla r_\varepsilon(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle + \frac{1}{2\alpha_t}\|\mathbf{x} - \mathbf{x}_t\|^2,$$

we have

$$\langle \nabla \phi_\varepsilon(\mathbf{x}_t), \mathbf{v}_{t+1} - \mathbf{x}_t \rangle + \frac{1}{2\alpha_t}\|\mathbf{v}_{t+1} - \mathbf{x}_t\|^2 \leq 0. \tag{A9}$$

Combining (A8) and (A9) and $\mathbf{v}_{t+1} = \mathbf{x}_t - \alpha_t \nabla \phi_\varepsilon(\mathbf{x}_t)$ in line 8 of Algorithm 1 yields

$$\phi_\varepsilon(\mathbf{v}_{t+1}) - \phi_\varepsilon(\mathbf{x}_t) \leq -\left(\frac{1}{2\alpha_t} - \frac{L_\varepsilon}{2}\right)\|\mathbf{v}_{t+1} - \mathbf{x}_t\|^2. \tag{A10}$$

Therefore, it is sufficient for $\alpha_t \leq \frac{1}{2/a+L_\varepsilon}$ for the criteria (A7) to be satisfied. This process only take finitely many iterations since we can find a finite t such that $\rho^t \alpha_t \leq \frac{1}{2/a+L_\varepsilon}$, and through the line search, we can obtain $\phi_\varepsilon(\mathbf{v}_{t+1}) \leq \phi_\varepsilon(\mathbf{x}_t)$.

Therefore, in either case of 1.1. or 1.2. where we take $\mathbf{x}_{t+1} = \mathbf{u}_{t+1}$ or $\mathbf{v}_{t+1}$, we can obtain

$$\phi_\varepsilon(\mathbf{x}_{t+1}) \leq \phi_\varepsilon(\mathbf{x}_t), \text{ for all } t \geq 0. \tag{A11}$$

Now, from case 1.1., (A6) gives

$$\|\nabla\phi_\varepsilon(\mathbf{x}_t)\|^2 \le a^2\|\mathbf{u}_{t+1}-\mathbf{x}_t\|^2 \le a^3(\phi_\varepsilon(\mathbf{x}_t)-\phi_\varepsilon(\mathbf{u}_{t+1})), \tag{A12a}$$

$$\text{therefore if } \mathbf{x}_{t+1}=\mathbf{u}_{t+1} \text{ we get } \|\nabla\phi_\varepsilon(\mathbf{x}_t)\|^2 \le a^3(\phi_\varepsilon(\mathbf{x}_t)-\phi_\varepsilon(\mathbf{x}_{t+1})). \tag{A12b}$$

From case 1.2. and $\mathbf{v}_{t+1}=\mathbf{x}_t-\alpha_t\nabla\phi_\varepsilon(\mathbf{x}_t)$, we have

$$\phi_\varepsilon(\mathbf{v}_{t+1})-\phi_\varepsilon(\mathbf{x}_t) \le -\frac{1}{a}\|\mathbf{v}_{t+1}-\mathbf{x}_t\|^2 = -\frac{1}{a}\alpha_t^2\|\nabla\phi_\varepsilon(\mathbf{x}_t)\|^2 \tag{A13a}$$

$$\Longrightarrow \|\nabla\phi_\varepsilon(\mathbf{x}_t)\|^2 \le \frac{a}{\alpha_t^2}\big(\phi_\varepsilon(\mathbf{x}_t)-\phi_\varepsilon(\mathbf{v}_{t+1})\big), \tag{A13b}$$

$$\text{then if } \mathbf{x}_{t+1}=\mathbf{v}_{t+1}, \text{ we have } \|\nabla\phi_\varepsilon(\mathbf{x}_t)\|^2 \le \frac{a}{\alpha_t^2}\big(\phi_\varepsilon(\mathbf{x}_t)-\phi_\varepsilon(\mathbf{x}_{t+1})\big). \tag{A13c}$$

Since $\frac{\delta}{L_\varepsilon} \le \alpha_t \le \frac{1}{2/a+L_\varepsilon}$, we have

$$\|\nabla\phi_\varepsilon(\mathbf{v}_{t+1})\|^2 \le \frac{aL_\varepsilon^2}{\delta^2}\big(\phi_\varepsilon(\mathbf{v}_{t+1})-\phi_\varepsilon(\mathbf{x}_{t+1})\big). \tag{A14}$$

Combining (A12b) and (A14) and selecting $C=\max\{\frac{aL_\varepsilon^2}{\delta^2}, a^3\}$, we obtain

$$\|\nabla\phi_\varepsilon(\mathbf{x}_t)\|^2 \le C\big(\phi_\varepsilon(\mathbf{x}_t)-\phi_\varepsilon(\mathbf{x}_{t+1})\big). \tag{A15}$$

Summing up (A15) for $t=0,\cdots,T$, we have

$$\sum_{t=0}^{T}\|\nabla\phi_\varepsilon(\mathbf{x}_t)\|^2 \le C\big(\phi_\varepsilon(\mathbf{x}_0)-\phi_\varepsilon(\mathbf{x}_{T+1})\big). \tag{A16}$$

Combined with the fact that $\phi_\varepsilon(\mathbf{x}) \ge \phi(\mathbf{x})-\varepsilon \ge \phi^*-\varepsilon$ for every $\mathbf{x}\in\mathcal{X}$, we have

$$\sum_{t=0}^{T}\|\nabla\phi_\varepsilon(\mathbf{x}_t)\|^2 \le C(\phi_\varepsilon(\mathbf{x}_0)-\phi^*+\varepsilon). \tag{A17}$$

The right-hand side is a finite constant, and hence by letting $t\to\infty$, we know that $\|\nabla\phi_\varepsilon(\mathbf{x}_t)\|\to 0$, which proves the first statement.

2. Denote $\kappa := \max\{t\in\mathbb{N} \mid \|\nabla\phi_\varepsilon(\mathbf{x}_{t+1})\| \ge \eta\}$; then, we know that $\|\nabla\phi_\varepsilon(\mathbf{x}_{t+1})\| \ge \eta$ for all $t \le \kappa-1$. Hence, we have

$$\kappa\eta^2 \le \sum_{t=0}^{\kappa-1}\|\nabla\phi_\varepsilon(\mathbf{x}_{t+1})\|^2 = \sum_{t=1}^{\kappa}\|\nabla\phi_\varepsilon(\mathbf{x}_t)\|^2 \le C(\phi_\varepsilon(\mathbf{x}_0)-\phi^*+\varepsilon). \tag{A18}$$

which implies the second statement.

□

Lemma A4. *Suppose that the sequence $\{\mathbf{x}_t\}$ is generated by Algorithm 1 with an initial guess $\mathbf{x}_0$. Then, for any $t\ge 0$, we have $\phi_{\varepsilon_{t+1}}(\mathbf{x}_{t+1})+\varepsilon_{t+1} \le \phi_{\varepsilon_t}(\mathbf{x}_t)+\varepsilon_t$.*

Proof. To prove this statement, we can prove

$$\phi_{\varepsilon_{t+1}}(\mathbf{x}_{t+1})+\varepsilon_{t+1} \le \phi_{\varepsilon_t}(\mathbf{x}_{t+1})+\varepsilon_t \le \phi_{\varepsilon_t}(\mathbf{x}_t)+\varepsilon_t. \tag{A19}$$

The second inequality is immediately obtained from (A11). Now, we prove the first inequality.

For any $\varepsilon>0$, denote

$$r_{\varepsilon,j}(\mathbf{x}) = \sqrt{\|\mathbf{g}_j(\mathbf{x})\|_2^2+\varepsilon^2}-\varepsilon. \tag{A20}$$

Since $\phi_\varepsilon(\mathbf{x}) = f(\mathbf{x}) + \sigma(\omega_i)\sum_{j=1}^m r_{\varepsilon,j}(\mathbf{x})$, it suffices to show that

$$r_{\varepsilon_{t+1},j}(\mathbf{x}_{t+1}) + \varepsilon_{t+1} \le r_{\varepsilon_t,j}(\mathbf{x}_{t+1}) + \varepsilon_t \tag{A21}$$

If $\varepsilon_{t+1} = \varepsilon_t$, then the two quantities above are identical and the first inequality holds. Now, suppose $\varepsilon_{t+1} = \gamma\varepsilon_t \le \varepsilon_t$; then,

$$r_{\varepsilon_{t+1},j}(\mathbf{x}_{t+1}) + \varepsilon_{t+1} = \sqrt{\|\mathbf{g}_j(\mathbf{x})\|_2^2 + \varepsilon_{t+1}^2} \le \sqrt{\|\mathbf{g}_j(\mathbf{x})\|_2^2 + \varepsilon_t^2} = r_{\varepsilon_t,j}(\mathbf{x}_{t+1}) + \varepsilon_t, \tag{A22}$$

which implies the first inequality of (A19). □

Theorem A5. *Suppose that $\{\mathbf{x}_t\}$ is the sequence generated by Algorithm 1 with any initial $\mathbf{x}_0$, $\epsilon_{\text{tol}} = 0$ and $T = \infty$. Let $\{\mathbf{x}_{t_l+1}\}$ be the subsequence that satisfies the reduction criterion in step 15 of Algorithm 1, i.e., $\|\nabla\phi_{\varepsilon_{t_l}}(\mathbf{x}_{t_l+1})\| \le \sigma\varepsilon_{t_l}\gamma$ for $t = t_l$ and $l = 1, 2, \cdots$. Then $\{\mathbf{x}_{t_l+1}\}$ has at least one accumulation point, and every accumulation point of $\{\mathbf{x}_{t_l+1}\}$ is a clarke stationary point of $\min_{\mathbf{x}} \phi(\mathbf{x}) := f(\mathbf{x}) + \sigma(\omega_i)r(\mathbf{x})$.*

Proof. By Lemma A4 and $\phi(\mathbf{x}) \le \phi_\varepsilon(\mathbf{x}) + \varepsilon$ for all $\varepsilon > 0$ and $\mathbf{x} \in \mathcal{X}$, we know that

$$\phi(\mathbf{x}_t) \le \phi_{\varepsilon_t}(\mathbf{x}_t) + \varepsilon_t \le \cdots \le \phi_{\varepsilon_0}(\mathbf{x}_0) + \varepsilon_0 < \infty. \tag{A23}$$

Since ϕ is coercive, we know that $\{\mathbf{x}_t\}$ is bounded, and the selected subsequence $\{\mathbf{x}_{t_l+1}\}$ is also bounded and has at least one accumulation point.

Note that $\|\nabla\phi_{\varepsilon_{t_l}}(\mathbf{x}_{t_l+1})\| \le \sigma\varepsilon_{t_l}\gamma = \sigma\varepsilon_0\gamma^{l+1} \to 0$ as $l \to \infty$. Let $\{\mathbf{x}_{p+1}\}$ be any convergent subsequence of $\{\mathbf{x}_{t_l+1}\}$ and denote ε_p as the corresponding ε_t used in Algorithm 1 that generates $\mathbf{x}_{p+1}$. Then, there exists $\mathbf{x}^* \in \mathcal{X}$ such that $\mathbf{x}_{p+1} \to \mathbf{x}^*$ as $\varepsilon_p \to 0$, and $\nabla\phi_{\varepsilon_p}(\mathbf{x}_{p+1}) \to 0$ as $p \to \infty$.

Note that the Clarke subdifferential of ϕ at $\mathbf{x}^*$ is given by $\partial\phi(\mathbf{x}^*) = \partial f(\mathbf{x}^*) + \sigma(\omega_i)\partial r(\mathbf{x}^*)$:

$$\partial\phi(\hat{\mathbf{x}}) = \{\nabla f(\hat{\mathbf{x}}) + \sigma(\omega_i)\sum_{j\in I_0}\nabla\mathbf{g}_j(\hat{\mathbf{x}})^\top\mathbf{w}_j + \sigma(\omega_i)\sum_{j\in I_1}\nabla\mathbf{g}_j(\hat{\mathbf{x}})^\top\frac{\mathbf{g}_j(\hat{\mathbf{x}})}{\|\mathbf{g}_j(\hat{\mathbf{x}})\|} \Big| \|\Pi(\mathbf{w}_j; \mathcal{C}(\nabla\mathbf{g}_j(\hat{\mathbf{x}})))\| \le 1,\ \forall\, j \in I_0\}, \tag{A24}$$

where $I_0 = \{j \in [m] \mid \|\mathbf{g}_i(\hat{\mathbf{x}})\| = 0\}$ and $I_1 = [m] \setminus I_0$.

If $j \in I_0$, we have $\|\mathbf{g}_j(\mathbf{x})\| = 0 \iff \mathbf{g}_j(\mathbf{x}) = 0$: then,

$$\partial r_\varepsilon(\mathbf{x}) = \sum_{j\in I_0}\nabla\mathbf{g}_j(\mathbf{x})^\top\frac{\mathbf{g}_j(\mathbf{x})}{\left(\|\mathbf{g}_j(\mathbf{x})\|^2 + \varepsilon^2\right)^{\frac{1}{2}}} + \sum_{j\in I_1}\nabla\mathbf{g}_j(\mathbf{x})^\top\frac{\mathbf{g}_j(\mathbf{x})}{\left(\|\mathbf{g}_j(\mathbf{x})\|^2 + \varepsilon^2\right)^{\frac{1}{2}}} \tag{A25a}$$

$$= \mathbf{0} + \sum_{j\in I_1}\nabla\mathbf{g}_j(\mathbf{x})^\top\frac{\mathbf{g}_j(\mathbf{x})}{\left(\|\mathbf{g}_j(\mathbf{x})\|^2 + \varepsilon^2\right)^{\frac{1}{2}}} \tag{A25b}$$

Therefore, we obtain

$$\nabla\phi_{\varepsilon_p}(\mathbf{x}_{p+1}) = \nabla f(\mathbf{x}_{p+1}) + \sigma(\omega_i)\sum_{j\in I_1}\nabla\mathbf{g}_j(\mathbf{x})^\top\frac{\mathbf{g}_j(\mathbf{x})}{\left(\|\mathbf{g}_j(\mathbf{x})\|^2 + \varepsilon_p^2\right)^{\frac{1}{2}}}. \tag{A26}$$

Comparing (A24) and (A26), we can see that the first term on the right-hand side of (A26) converges to that of (A24), due to the fact that $\mathbf{x}_{p+1} \to \hat{\mathbf{x}}$ and the continuity of ∇f. Together with the continuity of $\mathbf{g}_i$ and $\nabla\mathbf{g}_i$, the last term of (A24) converges to the last term of (A26)

as $\varepsilon_p \to 0$ and $\|\mathbf{g}_j(\mathbf{x})\| > 0$. Furthermore, apparently $\mathbf{0}$ is a special case of the second term in (A26). Hence, we know that

$$\operatorname{dist}(\nabla\phi_{\varepsilon_p}(\mathbf{x}_{p+1}), \partial\phi(\hat{\mathbf{x}})) \to 0,$$

as $p \to \infty$. Since $\nabla\phi_{\varepsilon_p}(\mathbf{x}_{p+1}) \to 0$ and $\partial\phi(\hat{\mathbf{x}})$ is closed, we conclude that $0 \in \partial\phi(\hat{\mathbf{x}})$. □

References

1. Li, D.; Yang, Y.; Song, Y.Z.; Hospedales, T.M. Learning to generalize: Meta-learning for domain generalization. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
2. Quiñonero-Candela, J.; Sugiyama, M.; Lawrence, N.D.; Schwaighofer, A. *Dataset Shift in Machine Learning*; Mit Press: Cambridge, MA, USA, 2009.
3. Ben-David, S.; Blitzer, J.; Crammer, K.; Pereira, F. Analysis of representations for domain adaptation. *Adv. Neural Inf. Process. Syst.* **2007**, *19*, 137.
4. Balaji, Y.; Sankaranarayanan, S.; Chellappa, R. Metareg: Towards domain generalization using meta-regularization. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 998–1008.
5. Fan, J.; Han, F.; Liu, H. Challenges of big data analysis. *Natl. Sci. Rev.* **2014**, *1*, 293–314. [CrossRef] [PubMed]
6. Day, O.; Khoshgoftaar, T.M. A survey on heterogeneous transfer learning. *J. Big Data* **2017**, *4*, 1–42. [CrossRef]
7. Munkhdalai, T.; Yu, H. Meta networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 2554–2563.
8. Finn, C.; Abbeel, P.; Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the International Conference on Machine Learning, Sydney, NSW, Australia, 6–11 August 2017; pp. 1126–1135.
9. Rusu, A.A.; Rao, D.; Sygnowski, J.; Vinyals, O.; Pascanu, R.; Osindero, S.; Hadsell, R. Meta-Learning with Latent Embedding Optimization. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
10. Yao, H.; Huang, L.K.; Zhang, L.; Wei, Y.; Tian, L.; Zou, J.; Huang, J. Improving generalization in meta-learning via task augmentation. In Proceedings of the International Conference on Machine Learning, Virtual Event, 18–24 July 2021; pp. 11887–11897.
11. Thrun, S.; Pratt, L. Learning to learn: Introduction and overview. In *Learning to Learn*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 3–17.
12. Hospedales, T.M.; Antoniou, A.; Micaelli, P.; Storkey, A.J. Meta-Learning in Neural Networks: A Survey. *arXiv* **2020**, arXiv:2004.05439.
13. Chen, Y.; Liu, H.; Ye, X.; Zhang, Q. Learnable Descent Algorithm for Nonsmooth Nonconvex Image Reconstruction. *SIAM J. Imaging Sci.* **2021**, *14*, 1532–1564. [CrossRef]
14. Yao, H.; Wu, X.; Tao, Z.; Li, Y.; Ding, B.; Li, R.; Li, Z. Automated relational meta-learning. *arXiv* **2020**, arXiv:2001.00745.
15. Lee, Y.; Choi, S. Gradient-based meta-learning with learned layerwise metric and subspace. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 2927–2936.
16. Huisman, M.; van Rijn, J.N.; Plaat, A. A survey of deep meta-learning. *Artif. Intell. Rev.* **2021**, *54*, 4483–4541. [CrossRef]
17. Koch, G.; Zemel, R.; Salakhutdinov, R. Siamese neural networks for one-shot image recognition. In Proceedings of the ICML Deep Learning Workshop, Lille, France, 6– 11 July 2015; Volume 2.
18. Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D. Matching networks for one shot learning. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 3630–3638.
19. Snell, J.; Swersky, K.; Zemel, R.S. Prototypical networks for few-shot learning. *arXiv* **2017**, arXiv:1703.05175.
20. Mishra, N.; Rohaninejad, M.; Chen, X.; Abbeel, P. A simple neural attentive meta-learner. *arXiv* **2017**, arXiv:1707.03141.
21. Ravi, S.; Larochelle, H. Optimization as a Model for Few-Shot Learning, 2016. Available online: https://openreview.net/forum?id=rJY0-Kcll (accessed on 29 October 2021).
22. Qiao, S.; Liu, C.; Shen, W.; Yuille, A.L. Few-shot image recognition by predicting parameters from activations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7229–7238.
23. Graves, A.; Wayne, G.; Danihelka, I. Neural turing machines. *arXiv* **2014**, arXiv:1410.5401.
24. Rajeswaran, A.; Finn, C.; Kakade, S.; Levine, S. Meta-Learning with Implicit Gradients, 2019. Available online: https://papers.nips.cc/paper/2019/hash/072b030ba126b2f4b2374f342be9ed44-Abstract.html (accessed on 29 October 2021)
25. Li, Z.; Zhou, F.; Chen, F.; Li, H. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv* **2017**, arXiv:1707.09835.
26. Antoniou, A.; Edwards, H.; Storkey, A. How to train your MAML. *arXiv* **2018**, arXiv:1810.09502.
27. Nichol, A.; Achiam, J.; Schulman, J. On first-order meta-learning algorithms. *arXiv* **2018**, arXiv:1803.02999.
28. Finn, C.; Rajeswaran, A.; Kakade, S.; Levine, S. Online meta-learning. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 1920–1930.
29. Grant, E.; Finn, C.; Levine, S.; Darrell, T.; Griffiths, T. Recasting gradient-based meta-learning as hierarchical bayes. *arXiv* **2018**, arXiv:1801.08930.
30. Finn, C.; Xu, K.; Levine, S. Probabilistic model-agnostic meta-learning. *arXiv* **2018**, arXiv:1806.02817.

31. Yoon, J.; Kim, T.; Dia, O.; Kim, S.; Bengio, Y.; Ahn, S. Bayesian model-agnostic meta-learning. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; pp. 7343–7353.
32. Vuorio, R.; Sun, S.H.; Hu, H.; Lim, J.J. Multimodal model-agnostic meta-learning via task-aware modulation. *arXiv* **2019**, arXiv:1910.13616.
33. Yao, H.; Wei, Y.; Huang, J.; Li, Z. Hierarchically structured meta-learning. In Proceedings of the International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 7045–7054.
34. Yin, M.; Tucker, G.; Zhou, M.; Levine, S.; Finn, C. Meta-Learning without Memorization. *arXiv* **2019**, arXiv:1912.03820.
35. Jenni, S.; Favaro, P. Deep bilevel learning. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 618–633.
36. Chen, W.Y.; Liu, Y.C.; Kira, Z.; Wang, Y.C.; Huang, J.B. A Closer Look at Few-shot Classification. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
37. Li, Y.; Yang, Y.; Zhou, W.; Hospedales, T. Feature-critic networks for heterogeneous domain generalization. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 3915–3924.
38. Rebuffi, S.A.; Bilen, H.; Vedaldi, A. Learning multiple visual domains with residual adapters. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
39. Triantafillou, E.; Zhu, T.; Dumoulin, V.; Lamblin, P.; Evci, U.; Xu, K.; Goroshin, R.; Gelada, C.; Swersky, K.J.; Manzagol, P.A.; et al. Meta-Dataset: A Dataset of Datasets for Learning to Learn from Few Examples. In Proceedings of the International Conference on Learning Representations, Addis Ababa, Ethiopia, 26–30 April 2020.
40. Yu, T.; Quillen, D.; He, Z.; Julian, R.; Hausman, K.; Finn, C.; Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In Proceedings of the Conference on Robot Learning, Virtual, 16–18 November 2020; pp. 1094–1100.
41. Haldar, J.P.; Hernando, D.; Liang, Z.P. Compressed-sensing MRI with random encoding. *IEEE Trans. Med Imaging* **2010**, *30*, 893–903. [CrossRef]
42. Blomgren, P.; Chan, T. Color TV: Total variation methods for restoration of vector-valued images. *IEEE Trans. Image Process.* **1998**, *7*, 304–309. [CrossRef]
43. Lundervold, A.S.; Lundervold, A. An overview of deep learning in medical imaging focusing on MRI. *Z. Für Med. Phys.* **2019**, *29*, 102–127. [CrossRef] [PubMed]
44. Liang, D.; Cheng, J.; Ke, Z.; Ying, L. Deep magnetic resonance image reconstruction: Inverse problems meet neural networks. *IEEE Signal Process. Mag.* **2020**, *37*, 141–151. [CrossRef]
45. Sandino, C.M.; Cheng, J.Y.; Chen, F.; Mardani, M.; Pauly, J.M.; Vasanawala, S.S. Compressed sensing: From research to clinical practice with deep neural networks: Shortening scan times for magnetic resonance imaging. *IEEE Signal Process. Mag.* **2020**, *37*, 117–127. [CrossRef]
46. McCann, M.T.; Jin, K.H.; Unser, M. Convolutional neural networks for inverse problems in imaging: A review. *IEEE Signal Process. Mag.* **2017**, *34*, 85–95. [CrossRef]
47. Zhou, S.K.; Greenspan, H.; Davatzikos, C.; Duncan, J.S.; van Ginneken, B.; Madabhushi, A.; Prince, J.L.; Rueckert, D.; Summers, R.M. A Review of Deep Learning in Medical Imaging: Image Traits, Technology Trends, Case Studies with Progress Highlights, and Future Promises. 2020. [CrossRef]
48. Singha, A.; Thakur, R.S.; Patel, T. Deep Learning Applications in Medical Image Analysis. *Biomed. Data Min. Inf. Retr. Methodol. Tech. Appl.* **2021**, *6*, 293–350.
49. Chandra, S.S.; Bran Lorenzana, M.; Liu, X.; Liu, S.; Bollmann, S.; Crozier, S. Deep learning in magnetic resonance image reconstruction. *J. Med Imaging Radiat. Oncol.* **2021**, *5*, 564–577. [CrossRef]
50. Ahishakiye, E.; Van Gijzen, M.B.; Tumwiine, J.; Wario, R.; Obungoloch, J. A survey on deep learning in medical image reconstruction. *Intell. Med.* **2021**. Available online: https://www.sciencedirect.com/science/article/pii/S2667102621000061 (accessed on 29 October 2021).
51. Hammernik, K. Learning a variational network for reconstruction of accelerated MRI data. *Magn. Reson. Med.* **2018**, *79*, 3055–3071. [CrossRef] [PubMed]
52. Aggarwal, H.K.; Mani, M.P.; Jacob, M. MoDL: Model-Based Deep Learning Architecture for Inverse Problems. *IEEE Trans. Med Imaging* **2019**, *38*, 394–405. [CrossRef]
53. Sun, J.; Li, H.; Xu, Z. Deep ADMM-Net for compressive sensing MRI. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 10–18.
54. Zhang, J.; Ghanem, B. ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 23 June 2018; pp. 1828–1837.
55. Lions, P.L.; Mercier, B. Splitting algorithms for the sum of two nonlinear operators. *SIAM J. Numer. Anal.* **1979**, *16*, 964–979. [CrossRef]
56. Beck, A.; Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.* **2009**, *2*, 183–202. [CrossRef]
57. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 30 June 2016; pp. 770–778.
58. Mehra, A.; Hamm, J. Penalty method for inversion-free deep bilevel optimization. *arXiv* **2019**, arXiv:1911.03432.
59. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2015**, arXiv:1412.6980.

60. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the International Conference on Artificial Intelligence and Statistics. Society for Artificial Intelligence and Statistics, Virtual Event, 13–15 April 2021.
61. Bernstein, M.A.; Fain, S.B.; Riederer, S.J. Effect of windowing and zero-filled reconstruction of MRI data on spatial resolution and acquisition strategy. *J. Magn. Reson. Imaging Off. J. Int. Soc. Magn. Reson. Med.* **2001**, *14*, 270–280. [CrossRef]
62. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015. Software. Available online: tensorflow.org (accessed on 29 October 2021).
63. Menze, B.H.; Jakab, A.; Bauer, S.; Kalpathy-Cramer, J.; Farahani, K.; Kirby, J.; Burren, Y.; Porz, N.; Slotboom, J.; Wiest, R.; et al. The multimodal brain tumor image segmentation benchmark (BRATS). *IEEE Trans. Med Imaging* **2014**, *34*, 1993–2024. [CrossRef]
64. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. Tensorflow: A system for large-scale machine learning. In Proceedings of the 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16), Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
65. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Sardinia, Italy, 13–15 May 2010; pp. 249–256.
66. Wang, S.; Cheng, H.; Ying, L.; Xiao, T.; Ke, Z.; Zheng, H.; Liang, D. DeepcomplexMRI: Exploiting deep residual network for fast parallel MR imaging with complex convolution. *Magn. Reson. Imaging* **2020**, *68*, 136–147. [CrossRef]
67. Hore, A.; Ziou, D. Image quality metrics: PSNR vs. SSIM. In Proceedings of the 2010 20th International Conference on Pattern Recognition, Istanbul, Turkey, 23–26 August 2010; pp. 2366–2369.
68. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef]
69. Poli, A.; Cirillo, M. On the use of the normalized mean square error in evaluating dispersion model performance. *Atmos. Environ. Part A Gen. Top.* **1993**, *27*, 2427–2434. [CrossRef]