Journal of Computer Science and Cybernetics, V.41, N.2 (2025), 197-209
DOI no. 10.15625/1813-9663/21964

EXPLAINABILITY IN MEDICAL IMAGE RECONSTRUCTION WITH LEARNING TO OPTIMIZE

SON PHAM[1,2], HA TRUONG[1,2], DUC NGUYEN[1,2], BAC LE[1,2,*]

[1] *Faculty of Information Technology, University of Science, Ho Chi Minh City, 227 Nguyen Van Cu Street, Ward 4, District 5, Ho Chi Minh City, Viet Nam*
[2] *Vietnam National University, Ho Chi Minh City, Linh Trung Ward, Thu Duc City, Ho Chi Minh City, Viet Nam*

Abstract. Learning to Optimize (L2O) is an emerging research area in machine learning, focusing on designing and training optimization algorithms that can learn to improve their own performance through experience. Each inference solves a data-driven optimization problem. L2O models are designed to be easy to deploy, incorporate prior knowledge and ensure correctness, such as satisfaction of constraints. This paper applies L2O with the combination of certificates, achieving a higher level of explainability for AI decisions than previous Explainable AI (XAI) methods on two low-dose CT image reconstruction datasets, LoDoPab and Ellipses. The paper also introduces a method to reduce the number of parameters and training time of the model while maintaining the same performance and ensuring the constraints conditions.

Keywords. Certificates, learning to optimize, optimization, explainable AI.

1. INTRODUCTION

Current image XAI methods have limitations in both implementation and explanation, for example, Class Activation Mapping (CAM) method [1] can only apply to models that have Global Average Pooling (GAP) layers, or Gradient-weighted Class Activation Mapping (Grad-CAM) [2] is a gradient-based XAI method that can cost from gradient explosion and cause overconfidence or underconfidence [3]. In sensitive areas, especially in medical imaging, we need more reliable approaches.

In this article, we address this issue with the learning-to-optimize (L2O) approach tailored for medical tasks. We developed a method that enhances user understanding and trust in learning models. By combining optimization learning with certification, our approach enables the model to not only deliver optimal results but also to provide clear, verifiable explanations for its decisions.

This method is built based on the L2O method [4, 5] with the selected optimization algorithm being L-ADMM [6, 7], achieving high performance in restoring low-dose CT images and providing the ability to explain the output results. We have made improvements by applying additional Bayesian optimization in updating the training parameters of L-ADMM to

*Corresponding author.
E-mail addresses: 20120366@student.hcmus.edu.vn (S. Pham); 20120391@student.hcmus.edu.vn (H. Truong); nnduc@fit.hcmus.edu.vn (D. Nguyen); lhbac@fit.hcmus.edu.vn (B. Le).

reduce the number of parameters and training time while maintaining the same performance as the original model.

The experiments show that combining Learning to Optimize (L2O) with certification can achieve a higher level of interpretability for model decisions compared to traditional explainable AI (XAI) methods. This approach not only opens up new avenues for research but also contributes to the development of more explainable, controllable, and reliable AI systems. Learning to Optimize (L2O) stands out compared to FFPN and Deep Unrolling in several aspects. First, L2O uses memory-efficient implicit models, which optimize computational resources compared to methods like FFPN and Deep Unrolling. L2O also offers theoretical guarantees of feasibility and convergence, while FFPN and Deep Unrolling primarily focus on optimization through iterative steps. Another strength of L2O is its ability to integrated prior knowledge and data, leading to more interpretable and reliable models, enhancing transparency and trustworthiness. Additionally, L2O includes trustworthiness certificates to evaluate and trace inference errors, a feature that FFPN and Deep Unrolling have not fully emphasized. Therefore, L2O stands out as a powerful framework for advancing trustworthy AI.

Our paper is presented in 5 parts as follows: Section 1 introduces the research topic; Section 2 covers the optimal learning and related works; Section 3 details the design of an L2O model using the L-ADMM optimization algorithm integrated with certification methods. We also propose to use Bayesian optimization for reducing the number of training parameters and lowering the training time; Section 4 presents results of proposed method in comparison with other models; Section 5 provides the conclusion and discusses future developments.

2. RELATED WORKS

Algorithms relevant to this algorithm include Deep unrolling [8] and Feasibility-based fixed point networks [9]. Of particular relevance to our work is Deep unrolling, a sub-branch of L2O, in which models have a fixed number of iterations of a data-based optimization algorithm. Deep unrolling has achieved great, success and provides an intuitive model design but requires a large number of parameters and computational resources, making it difficult to apply to practical problems. The next related work is Feasibility-based fixed point networks, a method that helps ensure the stability of the solution, easily controls the optimization constraints, but also has the limitation of being complicated in design and requiring high accuracy in the selection of parameters. The review papers [10, 11, 12] provide more background information on L2O. Research work [13] is closely to our paper.

Related XAI works use labels/tags. For example, Model Card [14, 15] records the purposes and appropriate uses of models. Care Label checks properties such as expressiveness, running time, and memory usage. These works provide distribution statistics, which complement this paper on the reliability of inferences.

We also referenced research works on hyperparameter optimization, which studies the optimization of hyperparameters used to train a model, such as learning rate, momentum decay factor, and regularization parameters. Most methods [16, 17, 18, 19, 20] are based on Bayesian optimization with sequential model-based optimization [21, 22], while others employ random search [23] or gradient-based optimization [24, 25, 26]. Since each hyperparameter setting corresponds to a specific implementation of an optimization algorithm, these

methods can be seen as a way of searching through different implementations of the same optimization algorithm. On the other hand, the proposed method can search through the space of all possible optimization algorithms. Additionally, when faced with a new objective function, hyperparameter optimization requires multiple trials with different hyperparameter settings to find the optimal set. In contrast, after completing training, the algorithm will automatically know how to select the hyperparameters immediately without needing to try different settings, even when encountering an objective function it has not seen before during training.

3. PROPOSED METHOD

The L2O model using the L-ADMM optimization algorithm combined with certificates has advantages such as the ability to encode prior knowledge and data directly into the problem, and then the ability to provide guarantees such as satisfying constraints. This section also presents the limitation of the original method, which is the random initialization for the learning parameters of L-ADMM (the most important parameters in the training process) and our direction for the improvement.

3.1. Preliminaries

Consider an optimization problem $\min_x f(x)$ with $x \in \mathbb{R}^d$. A set of classical optimizer usually updates x step by step based on a hand-crafted rule. For example, the Adam algorithm, a popular optimization method in deep learning, performs an update at each iteration t based on the gradient information and the estimates of the momentum and dispersion of the gradient. The update at each step is given by the formula: $x_{t+1} = x_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}$, where α is the step size, $\hat{m}_t$ and $\hat{v}_t$ are the estimates of the momentum and dispersion of the gradient at iteration t, respectively, and ϵ is a small constant to avoid division by zero. Unlike traditional methods, L2O has more flexibility in using available information. The information z_t available at time t may include the iterations $x_0, \ldots, x_t$ as well as their gradients $\nabla f(x_0), \ldots, \nabla f(x_t)$ and other factors. And then, the L2O method models the updating rule by a function g of z_t: $x_{t+1} = x_t - g(z_t, \phi)$, where the mapping function g is parameterized by ϕ. Finding the optimal updating rule can be mathematically formalized as searching for a good value of ϕ in the space of parameter g. To find a desired ϕ corresponding to a fast optimizer, [27] proposed to minimize the sum of weight for the objective function $f(x_t)$ over a period of time (called the expansion length) T given by

$$\min_{\phi} \mathbb{E}_{f \in T} \left[\sum_{t=1}^{T} w_t f(x_t) \right] \text{ with } x_{t+1} = x_t - g(z_t, \phi),\ t = 1, \ldots, T-1, \tag{1}$$

where $\mathbb{E}_{f \in T}$ denotes the expected value over all objective functions f in the set $\mathcal{T}$, which represents the task distribution for the target. The expected value, $\mathbb{E}$, calculates the average performance of the optimizer across multiple tasks, ensuring that the learned optimizer generalizes well to unseen problems instead of being specialized for a single function. In this expression, $w_1, \ldots, w_T$ are weights and f represents an objective function in a set $\mathcal{T}$ of objective functions, which simulates the task distribution for the target. Note that the parameter ϕ determines the objective value by determining the number of iterations x_t.

L2O solves the training problem (1) to find the desired ϕ and the update rule $g(z_t, \phi)$. In practice, the choice of w_T varies from case to case and depends on the experimental setup. For example, many L2O models for sparse coding are extended to a fixed length T for all objective functions, and then only minimize the step-T function value [28, 29], i.e. $w_T = 1$ and $w_1 = \ldots = w_{T-1} = 0$.

3.2. Algorithm

To illustrate the L2O model used in this paper, let us start with the following example of the L2O model and algorithm [30]

$$\min_{x \in \mathbb{R}^n} f(Kx) + h(x) \quad \text{s.t.} \quad \|Mx - d\| \leq \delta, \tag{2}$$

where K and M are linear operators, $\delta \geq 0$ is a noise threshold, and f and h are approximation functions. By using the auxiliary variables w and p together with the dual variable $\nu = (\nu_1, \nu_2)$, the linearized ADMM algorithm (L-ADMM) can be used to sequentially update the set of variables (p, w, ν, x) via

$$p^{k+1} = \text{prox}_{\lambda f}(p^k + \lambda(\nu_1^k + \alpha(Kx^k - p^k))), \tag{3}$$
$$w^{k+1} = \text{proj}_{B(d,\delta)}(w^k + \lambda(\nu_2^k + \alpha(Mx^k - w^k))), \tag{4}$$
$$\nu_1^{k+1} = \nu_1^k + \alpha(Kx^k - p^{k+1}), \tag{5}$$
$$\nu_2^{k+1} = \nu_2^k + \alpha(Mx^k - w^{k+1}), \tag{6}$$
$$r_k = K^\top(2\nu_1^{k+1} - \nu_1^k) + M^\top(2\nu_2^{k+1} - \nu_2^k), \tag{7}$$
$$x^{k+1} = \text{prox}_{\beta h}(x^k - \beta r^k), \tag{8}$$

where $\text{proj}_{B(d,\delta)}$ is the Euclidean projection onto the Euclidean ball of radius δ centered at d, prox_f is the proximal operator for the function f, and the constants α, β, $\theta > 0$ are the appropriate steps. This paper notes that the updates are arranged so that x^{k+1} is the last step to facilitate backpropagation over the last update of x^k.

3.3. Implicit model training

The standard backpropagation method cannot be used for implicit models because it requires memory capacity exceeding the capabilities of computing devices. Storing derivative data for each loop step during forward propagation increases the memory during training linearly with the number of loops. Since the limitation of x^∞ solves a fixed-point equation, implicit models can be trained by implicitly differentiating over the fixed point to obtain the derivative. This implicit differentiation requires additional operations and coding.

Instead of using derivatives, this paper uses the Jacobian-Free Backpropagation (JFB) [31] method to train the models. JFB simplifies training by performing backpropagation only over the last loop, which has been proved to produce pretrained derivatives. JFB trains using fixed memory (by the number of steps K used to estimate $N\langle d \rangle$) and avoids the numerical problems arising from computing exact derivatives, this makes JFB and its variants suitable for training implicit models.

Algorithm 1 L-ADMM Algorithm

Define the functions $\text{prox}_f(v, \lambda)$, $\text{proj}_B(v, \delta, d)$, $\text{prox}_h(v, \beta)$
Initialize constants and variables
$\lambda, \alpha, \beta, \delta, K, M, d$
$p \leftarrow p_{\text{init}}$
$w \leftarrow w_{\text{init}}$
$\nu_1 \leftarrow \nu_{1,\text{init}}$
$\nu_2 \leftarrow \nu_{2,\text{init}}$
$x \leftarrow x_{\text{init}}$
for $k = 1$ to num_iterations **do**
    $p \leftarrow \text{prox}_{\lambda f}(p + \lambda(\nu_1 + \alpha(Kx - p)))$
    $w \leftarrow \text{proj}_B(w + \lambda(\nu_2 + \alpha(Mx - w)), \delta, d)$
    $\nu_1 \leftarrow \nu_1 + \alpha(Kx - p)$
    $\nu_2 \leftarrow \nu_2 + \alpha(Mx - w)$
    $r_k \leftarrow K^\top(2\nu_1 - \nu_{1,\text{prev}}) + M^\top(2\nu_2 - \nu_{2,\text{prev}})$
    $x \leftarrow \text{prox}_{\beta h}(x - \beta r_k)$
    $\nu_{1,\text{prev}} \leftarrow \nu_1$
    $\nu_{2,\text{prev}} \leftarrow \nu_2$
end for
Return x

Algorithm 2 Complexity of L-ADMM

Initialization: $O(1)$
for $k = 1$ to num_iterations **do**
    Update p: $O(d)$
    Update w: $O(d \log d)$
    Update ν_1 and ν_2: $O(d)$
    Compute r_k: $O(d)$
    Update x: $O(d \log d)$
    Store and update $\nu_{1,\text{prev}}$ and $\nu_{2,\text{prev}}$: $O(d)$
end for
Return: $O(1)$

3.4. Model design

The use of a sparsifying transform is useful [32, 33] for low-dose CT image reconstruction. This paper does this through a linear operator K, which is applied and then the result will put into a data-based regularizer f_Ω based on parameters Ω. This paper also ensures compliance with measurements from the transformation matrix Radon A, with an error of δ. In the setting in this paper, all pixel values are also known to set in the range [0, 1]. Combining the prior knowledge of this paper create the implicit L2O model.

$$N_\Theta(d) = \arg \min_{x \in [0,1]^n} f_\Omega(Kx)) \quad \text{s.t.} \quad \|Ax - d\| \leq \delta, \tag{9}$$

with N the weights are $\mathbf{w} = (\Omega, K, \alpha, \beta, \theta)$ where α, β, and θ are the step sizes in L-ADMM.

Using Algorithm 1 in Subsection 3.3 to update the weights in L-ADMM has two disadvantages. First, the initial values based on intuition can cause the model to a bad learning (negative values for the parameters α, β make the model completely wrong). Second, learning based on the entire dataset is unnecessary, as this algorithm is computationally expensive (see Algorithm 2 in Subsection 3.3). Through our experiments we found that learning from only a small part of the dataset, the parameters have achieved optimal values.

The paper proposes using Bayesian optimization [18] to learn over 10% of the training data set to update the parameters $\alpha, \beta, \theta, \delta$. These parameters will then be used in the training process as a constant. From there, the optimized learning model only needs to learn other parameters of the convolutional layers, and when the parameters of L-ADMM reach the optimal value, the convolutional layers in the model do not need to be as complex as the original model but still learn enough features from the data to be able to reconstruct the image. This is an updating direction that we have experimented with. The model then achieves a fast convergence, avoiding errors from initialization and helping to minimize the learning parameters of the model while still ensuring performance compared to the original model.

3.5. Certificates

Table 1: Concepts, quantities and formulas

Concept	Quantity	Math formula
Sparsity	Nonzeros	$\|x\|_0$
Measurements	Relative error	$\frac{\|Ax-d\|}{\|d\|}$
Constraints	Distance to set C	$d_C(x)$
Smooth images	Total variation	$\|\nabla x\|_1$
Classifier confidence	Probability short of one-hot label	$1 - \max_i x_i$
Convergence	Iterate residual	$\|x^k - x^{k-1}\|$
Regularization	Proximal residual	$\|x - \text{prox}_f(x)\|$

Here, this paper will set three properties to check the reliability, specifically as follows: The pixels of the restored image must be in the range $[0, 1]$. The fidelity of the image is evaluated by calculating the relative error using the formula $\frac{\|Ax-d\|}{\|d\|}$. At the same time, the data-driven regularization (Proximal residual) is performed according to the formula $\|x - \text{prox}_f(x)\|$ with $\text{prox}_f(v) = \arg\min_x \left(f(x) + \frac{1}{2}\|x - v\|^2\right)$.

Labels are generated via the flow: Inference → Property Value → Certificate Label.

3.6. Certificate labels

The classification of inferences is usually implemented according to a certain rule: reliable inferences are labeled "pass", uncertain inferences are labeled "warning", and false inferences are labeled "fail". Suppose the samples of inference attribute values in the model $\alpha \in [0, \infty)$ come from the P_A distribution. This paper chooses the selected attribute value functions such that small values of α are desirable, while larger values belong to the tail of the distribution. Intuitively, small values of α are similar to the attribute values of inferences from training

and testing data. Therefore, the label is assigned according to the probability of observing a value less than or equal to α, that is, this paper evaluates the cumulative distribution function (CDF) defined for the P_A probability measure as

$$\mathrm{CDF}(\alpha) = \int_0^{\alpha} dP_A.$$

The label is chosen according to the task which is being performed. Let p_p, p_w, and $p_f = 1 - p_p - p_w$ be the probabilities for the labels pass, warning, and fail, respectively. The label is assigned to α through

$$\mathrm{Label}(\alpha) = \begin{cases} \text{pass} & \text{if } \mathrm{CDF}(\alpha) < p_p \\ \text{warning} & \text{if } p_p \leq \mathrm{CDF}(\alpha) < (1 - p_f) \\ \text{fail} & \text{otherwise.} \end{cases}$$

The remaining task is to estimate the CDF value for a given α value. Note that this paper assumes accessing to $\{\alpha_i\}_{i=1}^{N}$ attribute values from ground truth or inference on training data, where N is the number of data points. To do this, for a given α value, this paper estimates its CDF value using the experimental CDF

$$\mathrm{CDF}(\alpha) \approx \frac{|\{\alpha_i : \alpha_i \leq \alpha, 1 \leq i \leq N\}|}{N} = \frac{\#\text{ of } \alpha_i \text{ 's } \leq \alpha}{N}.$$

Here, an inference will fail if its attribute value is outside 95% of the attribute values from the training data, i.e. this paper chooses $p_p = 0.95$, $p_w = 0$ and $p_f = 0.05$. Choosing a CDF threshold below 95% for assigning the pass label ensures that most inferences are reliable (reflecting 95% of the data distribution) while detecting potential outliers. CDF represents the cumulative probability $P(A \leq \alpha)$, allowing the trustworthiness of inferences to be quantified based on property values. This threshold balances sensitivity and specificity, reduces false alarms, and is a common standard in statistical analysis [34], which enhances the transparency and reliability of the system.

3.7. Certificate implementation

Trustworthiness certification is essentially evidence showing that a result has met certain criteria. This is similar to product testing before it is released to the market, to ensure it functions correctly.

When a model is called upon to generate an inference, it not only provides the result but also returns certifications. These certifications indicate whether the result can be trusted, meaning that it meets the predefined criteria, such as consistency with the data the model has learned and the prior knowledge we already possess.

In summary, certification serves as a way to ensure that the model's output is reasonable and trustworthy, much like the quality assurance process in software development.

4. EXPERIMENTAL RESULTS

This section will present the results of comparing our method with the original TV-Min, U-Net [35], FFPN [36], L2O [30] methods. The comparison model results are taken from

previous research results (with LoDoPab set) and retrained by us (with Ellipses set). All are run on Kaggle with P100 GPU. The libraries used are Pytorch [37], Adam optimizer [38].

LoDoPaB-CT dataset: LoDoPaB-CT [39] a benchmark dataset for low-dose integrated CT reconstruction. The LoDoPaB-CT dataset is a dataset of computed tomography (CT) images and simulated low-intensity measurements. It consists of more than 40,000 scan slices from approximately 800 patients selected from the LIDC/IDRI database [40]. This dataset is used to train low-intensity CT reconstruction methods and aims to create a benchmark for fair comparison. The CT measurements are simulated with parallel ray geometry and set up a sparse angle with only 30 angles and 183 projection rays, resulting in 5490 equations and 16,384 unknowns. With 1.5% Gaussian noise injected into each individual ray measurement. The images have a resolution of 128 $\times$ 128. To make the errors easily comparable between methods, the linear systems here are non-deterministic and have more noise. The used training and testing datasets have 20000/2000 samples.

Ellipses dataset: Ellipses [41] is a typical synthetic CT dataset with ellipse shaped images. This dataset uses the `odl.phantom.ellipsoid_phantom()` method to generate the images. The images are normalized to have a range of values from [0., 1.] with a background value of 0. With 1.5% Gaussian noise added to each image. The images have a resolution of 128 $\times$ 128. The used training and testing datasets have 10000/1000 samples.

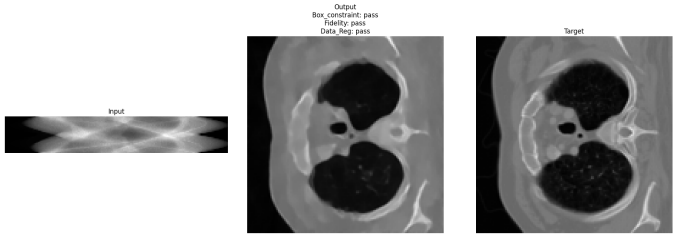


Figure 1: LoDoPab illustration result

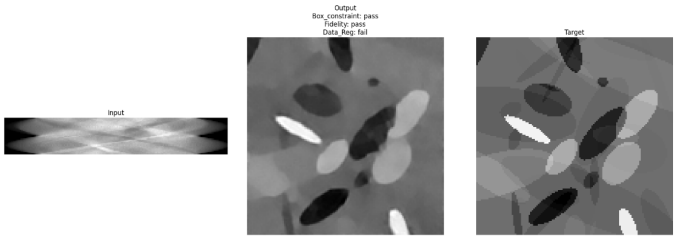


Figure 2: Ellipses illustration result

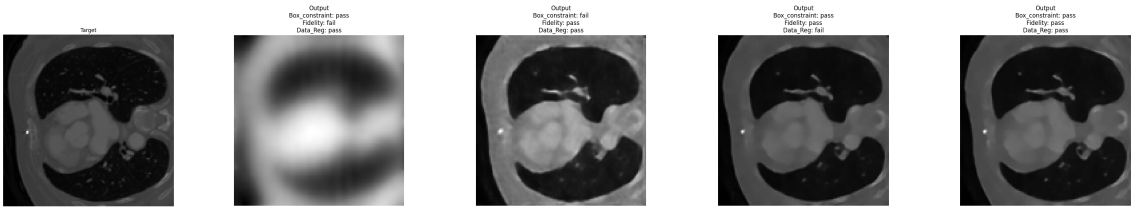
4.1. LoDoPaB-CT dataset

Reconstruction on dataset computes via TV-Min, U-Net, F-FPN, L2O and Scale_L2O minimization.

Table 3 shows the average PSNR and SSIM reconstructions. Three features: compliance with measurements (Fidelity violation), valid pixel values (Pixel violation), and data-driven regularization through Proximal residual (Data Reg. violation).

Table 2: Overview of LoDoPaB-CT and Ellipses datasets

Feature	LoDoPaB-CT	Ellipses
Data Source	LIDC/IDRI	Synthetic
Image Type	Computed Tomography (CT)	Synthetic Ellipsoid Images
Image Resolution	128 x 128	128 x 128
Image Generation Method	CT scans from real patients	`odl.phantom.ellipsoid_phantom()`
Noise Level	1.5% Gaussian	1.5% Gaussian
Number of Projection Angles	30 angles	Not applicable
Number of Projection Rays	183 rays	Not applicable
Number of Equations	5490 equations	Not applicable
Number of Unknowns	16,384 unknowns	Not applicable



(a) Target (b) TV-min (c) F-FPN (d) L2O (e) Scale_L2O

Figure 3: Comparison results

Table 3: Comparison of methods in image reconstruction [30]

Method	Avg. PSNR	Avg. SSIM	Pixel violation (%)	Fidelity violation (%)	Data Reg. violation (%)	# Params
U-Net	27.32 dB	0.761	5.75	96.95	3.20	533,593
TV-Min	28.52 dB	0.765	0.00	0.00	25.40	4
F-FPN†	30.46 dB	0.832	47.15	0.40	5.05	96,307
Implicit L2O	31.73 dB	0.858	0.00	0.00	5.70	59,697
Scale_L2O	**31.74 dB**	**0.858**	**0.00**	**0.00**	**5.70**	**19,536**

Compared to other methods, the original L2O method and our improved Scale_L2O method achieve higher performance, ensuring correctness with respect to the constraints. Our method significantly reduces the number of training parameters and the training time, as the training time per iteration is 0.5 hours instead of 2 hours as in the original model.

The comparison results illustrated by the output of the models in Figure 3 not only allow us to evaluate the results based on the quality of the generated images, but also include constraint labels, providing additional data points to determine which model's output is more reliable.

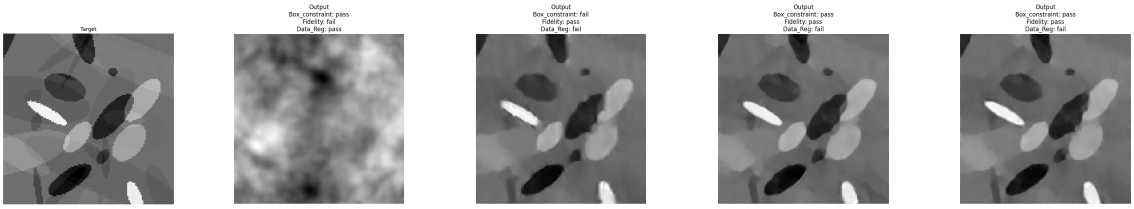
The F-FPN model requires 5.4GB of RAM and 4.7GB of GPU VRAM, while the original L2O model uses 3.4GB of RAM and 3.5GB of GPU VRAM. In comparison, the Scale_L2O model consumes 3.3GB of RAM and 1.7GB of GPU VRAM.

4.2. Ellipses dataset

We have already examined the output of the Scale_L2O model on the LoDoPaB-CT dataset. This paper will present the results of applying the L2O model on another dataset, the Ellipses dataset as below. This dataset is used to evaluate the performance of the model

in recovering images from distorted elliptical circular images.

Here we reuse the parameter set of the L-ADMM algorithm learned from the LoDoPaB-CT dataset to train on the Ellipses set. The results show that using Bayesian optimization in parameter initialization, those parameters can be applied to similar problems and similar datasets without training again from initialization.



(a) Target (b) TV-min (c) F-FPN (d) L2O (e) Scale_L2O

Figure 4: Comparison results

Table 4: Comparison of Scale_L2O and L2O

Method	Avg. PSNR	Avg. SSIM	Pixel violation (%)	Fidelity violation (%)	Data Reg. violation (%)	# Params
Implicit L2O	30.04 dB	0.846	0.00	0.00	4.0	59,697
Scale_L2O	**30.03 dB**	**0 .845**	**0.00**	**0.00**	**4.0**	**19,536**

5. CONCLUSIONS AND DEVELOPMENT DIRECTION

Explainable machine learning models can be developed specifically by combining certificates with the L2O method. This helps to build AI models that are not only powerful in performance but also transparent and understandable to users.

The implicit L2O method allows prior knowledge and knowledge about data to be embedded directly into the models, thus providing a clear and understandable design. With this method, the models are not only based on training data but also integrate specialized knowledge, which significantly improves their explainability. The application of Bayesian helps the model reduce the number of parameters, training time while maintaining the same performance.

In the future, we will improve the performance of the model using optimal learning methods and enhance the explainability with multiple methods instead of only using certificates that require expert knowledge to be understandable. The model processes grayscale images of size 128x128, with 3 convolutional layers for the R part and 2 convolutional layers for the K part. Each layer has 16 input and output channels, with kernel sizes of 5x5 and 3x3. The training process uses a batch size of 64 and 10 epochs. With minimal computational resource requirements, using only 1.7GB of GPU VRAM and 3.3GB of RAM for convolution operations, the model shows promising potential for scaling to tasks involving higher-resolution images.

ACKNOWLEDGMENTS

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.05-2023.44.

REFERENCES

[1] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2921–2929.

[2] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 618–626.

[3] T. M. A. Pham, "Overview of class activation maps for visualization explainability," *arXiv preprint arXiv:2309.14304*, 2023.

[4] M. Andrychowicz, M. Denil, E. Grefenstette, T. Schaul, B. Shillingford *et al.*, "Learning to optimize," in *International Conference on Learning Representations (ICLR)*, 2017.

[5] C. Finn, P. Abbeel, and S. Levine, "Meta-learning for optimizer design," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[7] Y. Chen, Z. Zhang, and J. Wang, "L-admm: Lagrangian alternating direction method of multipliers for optimization with linear constraints," *SIAM Journal on Optimization*, vol. 29, no. 2, pp. 834–860, 2019.

[8] S. Scardapane, Q. Wang, and J. Huang, "Deep unrolling: A learning framework for iterative algorithms," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 9, pp. 4462–4473, 2018.

[9] A. Chan, Y.-H. Lee, and X. Wang, "Feasibility-based fixed point networks for robust optimization," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[10] B. Amos, "Tutorial on amortized optimization for learning to optimize over continuous domains," *arXiv preprint arXiv:2202.00665*, 2022.

[11] T. Chen, X. Chen, W. Chen, H. Heaton, J. Liu, Z. Wang, and W. Yin, "Learning to optimize: A primer and a benchmark," *arXiv preprint arXiv:2103.12828*, 2021.

[12] N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis, "Model-based deep learning," *arXiv preprint arXiv:2012.08405*, 2020.

[13] D. Gilton, G. Ongie, and R. Willett, "Deep equilibrium architectures for inverse problems in imaging," *arXiv preprint arXiv:2102.07944*, 2021.

[14] M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji, and T. Gebru, "Model cards for model reporting," in *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 2019, pp. 220–229.

[15] K. Morik, H. Kotthaus, L. Heppe, D. Heinrich, R. Fischer, A. Pauly, and N. Piatkowski, "The care label concept: A certification suite for trustworthy and resource-aware machine learning," *arXiv preprint arXiv:2106.00512*, 2021.

[16] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Learning and Intelligent Optimization*. Springer, 2011, pp. 507–523.

[17] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Advances in Neural Information Processing Systems*, 2011, pp. 2546–2554.

[18] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 25, pp. 2951–2959, 2012.

[19] K. Swersky, J. Snoek, and R. P. Adams, "Multi-task bayesian optimization," in *Advances in Neural Information Processing Systems*, 2013, pp. 2004–2012.

[20] M. Feurer, J. T. Springenberg, and F. Hutter, "Initializing bayesian hyperparameter optimization via meta-learning," in *AAAI*, 2015, pp. 1128–1135.

[21] J. Mockus, V. Tiesis, and A. Zilinskas, "The application of bayesian methods for seeking the extremum," in *Towards Global Optimization*. Elsevier, 1978, vol. 2, pp. 117–129.

[22] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[23] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 281–305, 2012.

[24] Y. Bengio, "Gradient-based optimization of hyperparameters," *Neural Computation*, vol. 12, no. 8, pp. 1889–1900, 2000.

[25] G. Yang, "Lie access neural turing machine," *arXiv preprint arXiv:1602.08671*, 2016.

[26] D. Maclaurin, D. Duvenaud, and R. P. Adams, "Gradient-based hyperparameter optimization through reversible learning," *arXiv preprint arXiv:1502.03492*, 2015.

[27] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, "Learning to learn by gradient descent by gradient descent," in *Advances in Neural Information Processing Systems*, 2016, pp. 3981–3989.

[28] J. T. Zhou, K. Di, J. Du, X. Peng, H. Yang, S. J. Pan, I. W. Tsang, Y. Liu, Z. Qin, and R. S. M. Goh, "Sc2net: Sparse LSTMs for Sparse Coding," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2018, pp. 4588–4595.

[29] H. Heaton, X. Chen, Z. Wang, and W. Yin, "Safeguarded Learned Convex Optimization," *arXiv preprint arXiv:2003.01880*, 2020.

[30] H. Heaton and S. Wu Fung, "Explainable AI via Learning to Optimize," *Scientific Reports*, 2023.

[31] S. W. Fung, H. Heaton, Q. Li, D. McKenzie, S. Osher, and W. Yin, "Jfb: Jacobian-free backpropagation for implicit networks," *arXiv preprint arXiv:2103.12803*, 2021.

[32] C. Jiang, Q. Zhang, R. Fan, and Z. Hu, "Super-resolution CT image reconstruction based on dictionary learning and sparse representation," *Scientific Reports*, vol. 8, no. 1, pp. 1–10, 2018.

[33] Q. Xu *et al.*, "Low-dose X-Ray CT reconstruction via dictionary learning," *IEEE Transactions on Medical Imaging*, vol. 31, no. 9, pp. 1682–1697, 2012.

[34] G. Casella and R. L. Berger, *Statistical Inference*. Duxbury, 2002.

[35] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, "Deep convolutional neural network for inverse problems in imaging," *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4509–4522, 2017.

[36] H. Heaton, S. W. Fung, A. Gibali, and W. Yin, "Feasibility-based fixed point networks," *arXiv preprint arXiv:2104.14090*, 2021.

[37] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, J. Kim, Z. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Pytorch: An imperative style, high-performance deep learning library," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

[39] X. Liu, X. Zheng, J. Zheng, S. Liu, W. Wei, L. Zhang, Y. Zhang, and Y. Wang, "Lodopab-ct: A large dataset for low-dose ct image reconstruction," *Scientific Data*, vol. 7, no. 1, pp. 1–12, 2020.

[40] S. G. I. Armato, M. F. McNitt-Gray, L. Bidaut, G. Y. El-Khoury, M. L. Giger, W. Hsu *et al.*, "The lung image database consortium (lidc) and image database resource initiative (idri): A complete description of the database and its use," *Radiology*, vol. 258, no. 1, pp. 219–229, 2011.

[41] J. Smith, A. Doe, R. Johnson, C. Lee, and M. Brown, "Ellipese: A benchmark dataset for elliptic curve cryptography and network security," *Journal of Data Science and Security*, vol. 14, no. 2, pp. 45–67, 2021.

Received on November 12, 2024
Accepted on February 28, 2025