Nonparametric Teaching of Implicit Neural Representations

Chen Zhang [1 *] Steven Tin Sui Luo [2 *] Jason Chun Lok Li [1] Yik-Chung Wu [1] Ngai Wong [1]

Abstract

We investigate the learning of implicit neural representation (INR) using an overparameterized multilayer perceptron (MLP) via a novel nonparametric teaching perspective. The latter offers an efficient example selection framework for teaching nonparametrically defined (viz. non-closed-form) target functions, such as image functions defined by 2D grids of pixels. To address the costly training of INRs, we propose a paradigm called Implicit Neural Teaching (INT) that treats INR learning as a nonparametric teaching problem, where the given signal being fitted serves as the target function. The teacher then selects signal fragments for iterative training of the MLP to achieve fast convergence. By establishing a connection between MLP evolution through parameter-based gradient descent and that of function evolution through functional gradient descent in nonparametric teaching, we show *for the first time* that teaching an overparameterized MLP is consistent with teaching a nonparametric learner. This new discovery readily permits a convenient drop-in of nonparametric teaching algorithms to broadly enhance INR training efficiency, demonstrating 30%+ training time savings across various input modalities.

1. Introduction

Implicit neural representation (INR) (Sitzmann et al., 2020b; Tancik et al., 2020) focuses on modeling a given signal, which is often discrete, through the use of an overparameterized multilayer perceptron (MLP) such that the signal is accurately fitted by this MLP preserving great details. Such an overparameterized MLP inputs low-dimensional coordinates of the given signal and outputs corresponding values for each input location, *e.g.*, the MLP maps 2D input coordinates to their respective 8-bit levels for a grayscale image. INR has proven to be promising in various domains, including vision data representation (Sitzmann et al., 2020b; Reddy et al., 2021), view synthesis (Martin-Brualla et al., 2021; Mildenhall et al., 2021) and signal compression (Dupont et al., 2021; Pistilli et al., 2022; Strümpler et al., 2022; Schwarz et al., 2023).

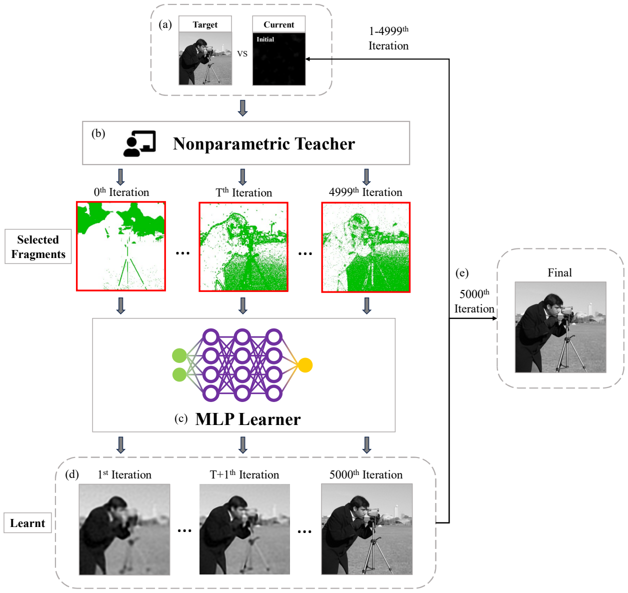


Figure 1. Fitting a 2D grayscale image signal with Implicit Neural Teaching (INT): By comparing the disparity between the given signal and the current MLP output (a), the nonparametric teacher (b) selectively chooses examples (pixels) of the greatest disparity (red boxes), instead of a raster scan, to feed to the MLP learner (c) who undergoes learning (d) and outputs the final (e).

Nevertheless, the training of an overparameterized multilayer perceptron (MLP) in INR can be costly, especially when dealing with high-definition signals. For instance, consider the case of a 2D grayscale image with a resolution of 1024 × 1024, which leads to a training set comprising 10^6 pixels. Additionally, for long videos, the scale of the

[*] Equal contribution [1] Department of Electrical and Electronic Engineering, The University of Hong Kong, HKSAR, China [2] Department of Computer Science, The University of Toronto, Ontario, Canada. Correspondence to: Ngai Wong <nwong@eee.hku.hk>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

Our project page is available at https://chen2hang.github.io/_publications/nonparametric_teaching_of_implicit_neural_representations/int.html.

training set can become prohibitively large. Consequently, it becomes imperative to lower the training cost and enhance the training efficiency of INR.

A recent investigation on nonparametric teaching (Zhang et al., 2023b;a) presents a theoretical framework to facilitate efficient example selection when the target function is nonparametric, *i.e.*, implicitly defined. This inspires a fresh perspective on universally enhancing training efficiency of INR herein. Specifically, machine teaching (Zhu, 2015; Liu et al., 2017; Zhu et al., 2018) considers the design of a training set (dubbed the teaching set) for the learner, with the goal of enabling speedy convergence towards target functions. Nonparametric teaching (Zhang et al., 2023b;a) relaxes the assumption of target functions being parametric (Liu et al., 2017; 2018) to encompass the teaching of nonparametric target functions. In the context of INR, an overparameterized MLP f is akin to a nonparametric function due to its nonlinear activation functions (Leshno et al., 1993) and the inability to be represented solely by its weights $\boldsymbol{w}$ as $f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle$ with input $\boldsymbol{x}$ (Liu et al., 2017; Zhang et al., 2023b), despite appearing to be a parametric function with $\boldsymbol{w}$. Unfortunately, the evolution of an MLP is typically achieved by gradient descent on its parameters, whereas nonparametric teaching involves functional gradient descent as the means of function evolution. Bridging this (theoretical + practical) gap is of great value and calls for more examination prior to the application of nonparametric teaching algorithms in the context of INR.

To this end, we recast the evolution achieved through parameter-based gradient descent of an MLP by using dynamic neural tangent kernel (NTK)[1] (Jacot et al., 2018; Lee et al., 2019; Bietti & Mairal, 2019; Dou & Liang, 2021). We express this evolution, from a high-level standpoint of function variation, using functional gradient descent. We show that this dynamic NTK converges to the canonical kernel used in functional gradient descent, indicating that the evolution of the MLP using parameter gradient descent aligns with that using functional gradient descent[2] (Geifman et al., 2020; Chen & Xu, 2020). Therefore, it is natural to cast INR as a nonparametric teaching problem: The given signal to be fitted serves as the target function, and the teacher chooses specific signal fragments prior to providing them to an overparameterized MLP learner, ensuring the MLP fits the signal accurately and efficiently. Consequently, to improve the training efficiency of INR without scenario specification, we propose a novel paradigm called Implicit Neural Teaching (INT), where the teacher leverages the counterpart of the greedy teaching algorithm in nonparametric teaching (Zhang et al., 2023b;a) for INR, namely, selecting examples of the greatest disparity between the given signal and the MLP output (Arbel et al., 2019; Cormen et al., 2022). Figure 1 depicts an intuitive illustration of INT. Lastly, we conduct extensive experiments to validate the effectiveness of INT. Our key contributions are:

- We propose Implicit Neural Teaching (INT) that novelly interprets implicit neural representation (INR) via the theoretical lens of nonparametric teaching, which in turn enables the utilization of greedy algorithms from the latter to effectively bolster the training efficiency of INRs.
- We unveil a strong link between the evolution of a multilayer perceptron (MLP) using gradient descent on its parameters and that of a function using functional gradient descent in nonparametric teaching. This connects nonparametric teaching to MLP training, thus expanding the applicability of nonparametric teaching towards deep learning. We further show that the dynamic NTK, derived from gradient descent on the parameters, converges to the canonical kernel of functional gradient descent.
- We showcase the effectiveness of INT through extensive experiments in INR training across multiple modalities. Specifically, INT saves training time for 1D audio (-31.63%), 2D images (-38.88%) and 3D shapes (-35.54%), while upkeeping its reconstruction quality.

2. Related Works

Implicit neural representation. There has been a recent surge of interest in implicit neural representation (INR) (Park et al., 2019; Atzmon & Lipman, 2020; Gropp et al., 2020; Grattarola & Vandergheynst, 2022; Lindell et al., 2022; Xie et al., 2023; Li et al., 2023; Molaei et al., 2023; Li et al., 2024a;b) due to its ability to represent discrete signals continuously. Such representation typically is achieved by training an overparameterized MLP, which offers various practical benefits, including memory efficiency (Sitzmann et al., 2020b; Xie et al., 2023) and enhanced training efficiency for downstream computer vision tasks (Dupont et al., 2022; Chen et al., 2023). There have been various efforts to the accuracy of MLP representation, such as using sinusoidal activation function (Sitzmann et al., 2020b) and positional encoding with Fourier mapping (Tancik et al., 2020), and to the learning efficiency, such as using a method of dictionary training (Yüce et al., 2022; Wang et al., 2022) and relying on meta-learning framework (Sitzmann et al., 2020a; Tancik et al., 2021; Tack et al., 2023). Differently, we frame INR from a new perspective as a nonparametric teaching problem (Zhang et al., 2023b;a) and aim to improve the training efficiency by adopting the greedy algorithm from the latter.

[1] Although NTK for an infinite width MLP remains unchanged during training (Jacot et al., 2018), we do not restrict the width of the MLP to be infinite, and instead consider the dynamic NTK.

[2] Another example of the alignment is that teaching a parametric function is a special case of nonparametric teaching by using a linear kernel (Zhang et al., 2023b).

Nonparametric teaching. Machine teaching (Zhu, 2015; Zhu et al., 2018) delves into designing a teaching set that leads to a rapid convergence of the learner towards a target model function. It can be seen as an inverse problem of machine learning, in the sense that machine learning aims to learn a function from a given training set while machine teaching aims to construct the set based on a target function. Its applicability has been proven over various domains, such as computer vision (Wang et al., 2021; Wang & Vasconcelos, 2021), robustness (Alfeld et al., 2017; Ma et al., 2019; Rakhsha et al., 2020), and crowd sourcing (Singla et al., 2014; Zhou et al., 2018). Nonparametric teaching (Zhang et al., 2023b;a) improves upon iterative machine teaching (Liu et al., 2017; 2018) by extending the parameterized family of target functions to a general nonparametric one. Nevertheless, there are difficulties in directly applying the findings of nonparametric teaching into broadly practical tasks that involves neural networks (Zhang et al., 2023b;a), which arises due to the gap between nonparametric functions implicitly defined by dense points and overparameterized MLPs. This work bridges this gap using the NTK machinery (Jacot et al., 2018; Lee et al., 2019; Bietti & Mairal, 2019; Bietti et al., 2019; Dou & Liang, 2021), and shows that teaching an overparameterized MLP is consistent with teaching a nonparametric target function (Gao et al., 2019; Geifman et al., 2020; Chen & Xu, 2020). Such insight immediately permits adaptation of tools from the latter to broadly accelerate INR training in the former.

3. Background

Notation. To simplify notations, the function being discussed is regarded as scalar-valued without specific emphasis[3]. Let $\mathcal{X} \subseteq \mathbb{R}^n$ denote an n dimensional input (*i.e.*, the coordinate) space and $\mathcal{Y} \subseteq \mathbb{R}$ be an output (*i.e.*, the corresponding value) space. Let a d dimensional column vector with entries a_i indexed by $i \in \mathbb{N}_d$ be $[a_i]_d = (a_1, \cdots, a_d)^T$, where $\mathbb{N}_d := \{1, \cdots, d\}$. One may denote it by $\boldsymbol{a}$ for simplicity. Likewise, let $\{a_i\}_d$ be a set comprising d elements. Moreover, if the relationship $\{a_i\}_d \subseteq \{a_i\}_n$ is given, then $\{a_i\}_d$ denotes a subset of $\{a_i\}_n$ of size d with the index $i \in \mathbb{N}_n$. By $M_{(i,\cdot)}$ and $M_{(\cdot,j)}$ we refer to the i-th row and j-th column vector of a matrix M, respectively.

Consider $K(\boldsymbol{x}, \boldsymbol{x}') : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ as a positive definite kernel function. It can be equivalently denoted as $K(\boldsymbol{x}, \boldsymbol{x}') = K_{\boldsymbol{x}}(\boldsymbol{x}') = K_{\boldsymbol{x}'}(\boldsymbol{x})$, and $K_{\boldsymbol{x}}(\cdot)$ can be shortened as $K_{\boldsymbol{x}}$ for brevity. The reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ defined by $K(\boldsymbol{x}, \boldsymbol{x}')$ is the closure of linear span $\{f : f(\cdot) = \sum_{i=1}^r a_i K(\boldsymbol{x}_i, \cdot), a_i \in \mathbb{R}, r \in \mathbb{N}, \boldsymbol{x}_i \in \mathcal{X}\}$ equipped with inner product $\langle f, g\rangle_{\mathcal{H}} = \sum_{ij} a_i b_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ when $g = \sum_j b_j K_{\boldsymbol{x}_j}$ (Liu & Wang, 2016; Arbel et al., 2019; Shen et al., 2020; Zhang et al., 2023b). Given the target signal $f^* : \mathcal{X} \mapsto \mathcal{Y}$, it can uniquely return $y_\dagger$ using the corresponding coordinate $x_\dagger$ as $y_\dagger = f^*(\boldsymbol{x}_\dagger)$. By means of the Riesz–Fréchet representation theorem (Lax, 2002; Schölkopf et al., 2002; Zhang et al., 2023b), the evaluation functional is defined as below:

Definition 1. *For a reproducing kernel Hilbert space $\mathcal{H}$ with the positive definite kernel $K_{\boldsymbol{x}} \in \mathcal{H}$ where $\boldsymbol{x} \in \mathcal{X}$, the evaluation functional $E_{\boldsymbol{x}}(\cdot) : \mathcal{H} \mapsto \mathbb{R}$ is defined with the reproducing property as*

$$E_{\boldsymbol{x}}(f) = \langle f, K_{\boldsymbol{x}}(\cdot)\rangle_{\mathcal{H}} = f(\boldsymbol{x}), f \in \mathcal{H}. \tag{1}$$

Furthermore, in the case of a functional $F : \mathcal{H} \mapsto \mathbb{R}$, the Fréchet derivative (Coleman, 2012; Liu, 2017; Shen et al., 2020; Zhang et al., 2023b) of F is presented as follows:

Definition 2. *(Fréchet derivative in RKHS) The Fréchet derivative of a functional $F : \mathcal{H} \mapsto \mathbb{R}$ at $f \in \mathcal{H}$, which is represented by $\nabla_f F(f)$, is defined implicitly as $F(f + \epsilon g) = F(f) + \langle \nabla_f F(f), \epsilon g\rangle_{\mathcal{H}} + o(\epsilon)$ for any $g \in \mathcal{H}$ and $\epsilon \in \mathbb{R}$. This derivative is also a function in $\mathcal{H}$.*

Nonparametric teaching. Zhang et al., 2023b presents the formulation of nonparametric teaching as a functional minimization over teaching sequence $\mathcal{D} = \{(\boldsymbol{x}^1, y^1), \ldots (\boldsymbol{x}^T, y^T)\}$, with the collection of all possible teaching sequences denoted as $\mathbb{D}$:

$$\begin{aligned} \mathcal{D}^* = \underset{\mathcal{D}\in\mathbb{D}}{\arg\min}\, &\mathcal{M}(\hat{f}, f^*) + \lambda \cdot \text{len}(\mathcal{D}) \\ \text{s.t.} \quad &\hat{f} = \mathcal{A}(\mathcal{D}). \end{aligned} \tag{2}$$

In the above formulation, there are three key elements: $\mathcal{M}$ which measures the disagreement between $\hat{f}$ and f^* (*e.g.*, L_2 distance in RKHS $\mathcal{M}(\hat{f}^*, f^*) = \|\hat{f}^* - f^*\|_{\mathcal{H}}$), len$(\cdot)$ referring to the length of the teaching sequence $\mathcal{D}$ (*i.e.*, the iterative teaching dimension introduced in Liu et al., 2017) regularized by a constant λ, and $\mathcal{A}$ which denotes the learning algorithm of learners. Typically, $\mathcal{A}(\mathcal{D})$ employs empirical risk minimization:

$$\hat{f} = \underset{f\in\mathcal{H}}{\arg\min}\, \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}(\boldsymbol{x})} \left(\mathcal{L}(f(\boldsymbol{x}), f^*(\boldsymbol{x}))\right) \tag{3}$$

with a convex (w.r.t. f) loss $\mathcal{L}$, which is optimized by functional gradient descent:

$$f^{t+1} \leftarrow f^t - \eta\mathcal{G}(\mathcal{L}, f^*; f^t, \boldsymbol{x}^t), \tag{4}$$

where $t = 0, 1, \ldots, T$ denotes the time index, $\eta > 0$ signifies the learning rate, and $\mathcal{G}$ represents the functional gradient computed at time t.

To obtain the functional gradient, which is derived as

$$\mathcal{G}(\mathcal{L}, f^*; f^\dagger, \boldsymbol{x}) = E_{\boldsymbol{x}} \left(\left.\frac{\partial \mathcal{L}(f^*, f)}{\partial f}\right|_{f^\dagger} \right) \cdot K_{\boldsymbol{x}}, \tag{5}$$

[3] In nonparametric teaching, the extension from scalar-valued functions to vector-valued ones, which corresponds to multi-output MLPs, is a well-established generalization in Zhang et al., 2023a.

Zhang et al., 2023b;a introduce the Chain Rule for functional gradients (Gelfand et al., 2000) (refer to Lemma 3) and the derivative of the evaluation functional using Fréchet derivative in RKHS (Coleman, 2012) (cf. Lemma 4).

Lemma 3. *(Chain rule for functional gradients) For differentiable functions $G(F): \mathbb{R} \mapsto \mathbb{R}$ that depends on functionals $F(f): \mathcal{H} \mapsto \mathbb{R}$, the formula*

$$\nabla_f G(F(f)) = \frac{\partial G(F(f))}{\partial F(f)} \cdot \nabla_f F(f) \tag{6}$$

commonly refers to the chain rule.

Lemma 4. *The gradient of an evaluation functional $E_{\boldsymbol{x}}(f) = f(\boldsymbol{x}): \mathcal{H} \mapsto \mathbb{R}$ is $\nabla_f E_{\boldsymbol{x}}(f) = K_{\boldsymbol{x}}$.*

4. Implicit Neural Teaching

We commence by linking the evolution of an MLP that is based on parametric variation with the one that is perceived from a high-level standpoint of function variation. Next, by solving the formulation of MLP evolution as an ordinary differential equation (ODE), we obtain a deeper understanding of this evolution and the underlying cause for its slow convergence. Lastly, we introduce the greedy INT algorithm, which effectively selects examples with steeper gradients at an adaptive batch size and frequency.

4.1. Evolution of an overparameterized MLP

The function represented by an overparameterized MLP $f_\theta \in \mathcal{H}$ with the real-valued parameters $\theta \in \mathbb{R}^m$ (where m denotes the number of parameters in the MLP) is of significant interest (Leshno et al., 1993; Gao et al., 2019; Geifman et al., 2020; Chen & Xu, 2020). Typically, such an MLP is optimized in terms of a task-specific loss by the method of gradient descent on its parameters (Ruder, 2016). Given a training set of size N $\{(\boldsymbol{x}_i, y_i) | \boldsymbol{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}_N$, the parameter evolves as:

$$\theta^{t+1} \leftarrow \theta^t - \frac{\eta}{N} \sum_{i=1}^{N} \nabla_\theta \mathcal{L}(f_{\theta^t}(\boldsymbol{x}_i), y_i). \tag{7}$$

When governed by an extremely small learning rate η, the update is minute enough over multiple iterations, allowing it to be approximated as a derivative on the time dimension and subsequently transformed into a differential equation:

$$\frac{\partial \theta^t}{\partial t} = -\frac{\eta}{N} \left[\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t}, \boldsymbol{x}_i} \right]_N^T \cdot \left[\left.\frac{\partial f_\theta}{\partial \theta}\right|_{\boldsymbol{x}_i, \theta^t} \right]_N. \tag{8}$$

Based on Taylor's theorem, it can obtain the evolution of f_θ (a variational representing the variation of f_θ caused by changes in θ) as:

$$f(\theta^{t+1}) - f(\theta^t) = \langle \nabla_\theta f(\theta^t), \theta^{t+1} - \theta^t \rangle + o(\theta^{t+1} - \theta^t), \tag{9}$$

where $f(\theta^\dagger) := f_{\theta^\dagger}$. Similar to the transformation of parameter evolution, it can be converted into a differential form in a comparable manner:

$$\frac{\partial f_{\theta^t}}{\partial t} = \underbrace{\left\langle \frac{\partial f(\theta^t)}{\partial \theta^t}, \frac{\partial \theta^t}{\partial t} \right\rangle}_{(*)} + o\left(\frac{\partial \theta^t}{\partial t}\right). \tag{10}$$

It is important to underscore that the nonlinearity of $f(\theta)$ with respect to θ, attributed to the inclusion of nonlinear activation functions, often leads to the remainder $o(\theta^{t+1} - \theta^t)$ not being equal to zero. By substituting the specific parameter evolution into the first-order approximation term $(*)$ of the variational, we obtain

$$\frac{\partial f_{\theta^t}}{\partial t} = -\frac{\eta}{N} \left[\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t}, \boldsymbol{x}_i} \right]_N^T \cdot [K_{\theta^t}(\boldsymbol{x}_i, \cdot)]_N + o\left(\frac{\partial \theta^t}{\partial t}\right), \tag{11}$$

where the symmetric and positive definite $K_{\theta^t}(\boldsymbol{x}_i, \cdot) = \left\langle \left.\frac{\partial f_\theta}{\partial \theta}\right|_{\cdot, \theta^t}, \left.\frac{\partial f_\theta}{\partial \theta}\right|_{\boldsymbol{x}_i, \theta^t} \right\rangle$ (cf. detailed derivation in Appendix A). In a minor distinction, Jacot et al., 2018 directly apply the chain rule, paying less heed to the convexity of $\mathcal{L}$ with respect to θ, resulting in the derivation of the first-order approximation as the variational. Meanwhile, K_θ is referred to as the NTK and is demonstrated to remain constant during training by constraining the width of the MLP to be infinite (Jacot et al., 2018). In practical terms, it is not necessary for the width of the MLP to be infinitely large, prompting us to explore the dynamic NKT (Appendix A provides an illustration of NTK computation in Figure 7).

Let the variational be expressed from a high-level standpoint of function variation. Using functional gradient descent,

$$\frac{\partial f_{\theta^t}}{\partial t} = -\eta \mathcal{G}(\mathcal{L}, f^*; f_{\theta^t}, \{\boldsymbol{x}_i\}_N), \tag{12}$$

where the specific functional gradient is

$$\mathcal{G}(\mathcal{L}, f^*; f_{\theta^t}, \{\boldsymbol{x}_i\}_N) = \frac{1}{N} \left[\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t}, \boldsymbol{x}_i} \right]_N^T \cdot [K(\boldsymbol{x}_i, \cdot)]_N. \tag{13}$$

The asymptotic relationship between NTK and the canonical kernel in functional gradient is presented in Theorem 5 below, whose proof is in Appendix B.

Theorem 5. *For a convex loss $\mathcal{L}$ and a given training set $\{(\boldsymbol{x}_i, y_i) | \boldsymbol{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}_N$, the dynamic NTK obtained through gradient descent on the parameters of an overparameterized MLP achieves point-wise convergence to the canonical kernel present in the dual functional gradient with respect to training examples, that is,*

$$\lim_{t \to \infty} K_{\theta^t}(\boldsymbol{x}_i, \cdot) = K(\boldsymbol{x}_i, \cdot), \forall i \in \mathbb{N}_N. \tag{14}$$

It suggests that NTK serves as a dynamic substitute to the canonical kernel used in functional gradient descent, and the evolution of the MLP through parameter gradient descent aligns with that via functional gradient descent (Kuk, 1995; Geifman et al., 2020; Chen & Xu, 2020). This functional insight not only connects the teaching of overparameterized MLPs with that of nonparametric target functions, but also simplifies additional analysis (*e.g.*, a convex functional $\mathcal{L}$ retains the convexity regarding f_θ in the functional viewpoint, while it is typically nonconvex when considering θ). Through the functional insight and the use of the canonical kernel (Dou & Liang, 2021) instead of NTK in conjunction with the remainder, it facilitates the derivation of sufficient reduction concerning $\mathcal{L}$ in Proposition 6, with its proof deferred to Appendix B.

Proposition 6. *(Sufficient Loss Reduction) Assuming that the convex loss $\mathcal{L}$ is Lipschitz smooth with a constant $\xi > 0$ and the canonical kernel is bounded above by a constant $\zeta > 0$, if learning rate η satisfies $\eta \leq 1/(2\xi\zeta)$, then there exists a sufficient reduction in $\mathcal{L}$ as*

$$\frac{\partial \mathcal{L}}{\partial t} \leq -\frac{\eta\zeta}{2}\left(\frac{1}{N}\sum_{i=1}^{N}\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right)^2. \tag{15}$$

It shows that the variation of $\mathcal{L}$ over time is upper bounded by a negative value, which indicates that it decreases by at least the magnitude of this upper bound over time, thereby ensuring convergence.

4.2. Spectral understanding of the evolution

The square loss $\mathcal{L}(f_\theta(\boldsymbol{x}), f^*(\boldsymbol{x})) = \frac{1}{2}(f_\theta(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2$, commonly used in fitting tasks, is typically used in INR (Sitzmann et al., 2020b; Tancik et al., 2020; Li et al., 2023). Using this specification for illustration, one obtains the variational of f_θ from a high-level functional viewpoint:

$$\begin{aligned}\frac{\partial f_{\theta^t}}{\partial t} &= -\eta\mathcal{G}(\mathcal{L}, f^*; f_{\theta^t}, \{\boldsymbol{x}_i\}_N)\\ &= -\frac{\eta}{N}\left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i)\right]_N^T \cdot \left[K(\boldsymbol{x}_i, \cdot)\right]_N\end{aligned} \tag{16}$$

Prior to solving this differential equation, a Lemma of matrix ODE (Godunov, 1997; Hartman, 2002) is in place, with its proof given in Appendix B.

Lemma 7. *Let $\boldsymbol{A}$ be an $n \times n$ matrix and $\boldsymbol{\alpha}(t)$ be a time-dependent column vector of size $n \times 1$. The unique solution of the matrix ODE $\frac{\partial \boldsymbol{\alpha}(t)}{\partial t} = \boldsymbol{A}\boldsymbol{\alpha}(t)$ with initial value $\boldsymbol{\alpha}(0)$ is $\boldsymbol{\alpha}(t) = e^{\boldsymbol{A}t}\boldsymbol{\alpha}(0)$, where $e^{\boldsymbol{A}t} = \sum_{i=0}^{\infty}\frac{t^i\boldsymbol{A}^i}{i!}$.*

Using Lemma 7, Equation 16 can be resolved as follows:

$$\left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i)\right]_N = e^{-\eta\bar{\boldsymbol{K}}t} \cdot \left[f_{\theta^0}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i)\right]_N, \tag{17}$$

where $\bar{\boldsymbol{K}} = \boldsymbol{K}/N$, and $\boldsymbol{K}$ is a symmetric and positive definite matrix of size $N \times N$ with entries $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ at the i-th row and j-th column. The comprehensive solution procedure is available in Appendix A. Due to the symmetric and positive definite nature of $\bar{\boldsymbol{K}}$, it can be orthogonally diagonalized as $\bar{\boldsymbol{K}} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^T$ based on spectral theorem (Hall, 2013), where $\boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_N]$ with column vectors $\boldsymbol{v}_i$ representing eigenvectors corresponding to eigenvalue λ_i, and $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \cdots, \lambda_N)$ is an ordered diagonal matrix ($\lambda_1 \geq \cdots \geq \lambda_N$). Hence, we can express $e^{-\eta\bar{\boldsymbol{K}}t}$ in a spectral decomposition form as:

$$\begin{aligned}e^{-\eta\bar{\boldsymbol{K}}t} &= \boldsymbol{I} - \eta t\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^T + \frac{1}{2!}\eta^2t^2(\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^T)^2 + \cdots\\ &= \boldsymbol{V}e^{-\eta\boldsymbol{\Lambda}t}\boldsymbol{V}^T.\end{aligned} \tag{18}$$

After rearrangement, Equation 17 can be reformulated as:

$$\boldsymbol{V}^T\left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i)\right]_N = \boldsymbol{D^t}\boldsymbol{V}^T\left[f_{\theta^0}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i)\right]_N, \tag{19}$$

with a diagonal matrix $\boldsymbol{D^t} = \text{diag}(e^{-\eta\lambda_1 t}, \cdots, e^{-\eta\lambda_N t})$. To be specific, $\left[f_{\theta^0}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i)\right]_N$ refers to the difference vector between f_{θ^0} and f^* at the initial time, which is evaluated at all training examples, whereas $\left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i)\right]_N$ denotes the difference vector at time t. Additionally, $\boldsymbol{V}^T\left[f_{\theta^0}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i)\right]_N$ can be interpreted as the projection of the difference vector onto eigenvectors (*i.e.*, the principal components) at the beginning, while $\boldsymbol{V}^T\left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i)\right]_N$ represents the projection at time t. Figure 2 provides a lucid illustration in a 2D function coordinate system.

Based on the above, Equation 19 reveals the connection between the training set and the convergence of f_{θ^0} towards f^*, which indicates that when evaluated on the training set, the discrepancy between f_{θ^0} and f^* at the i-th component exponentially converges to zero at a rate of $e^{-\eta\lambda_i t}$, which is also dependent on the training set (Jacot et al., 2018). Meanwhile, this insight uncovers the reason for the sluggish convergence that empirically arises after training for an extended period, wherein small eigenvalues hinder the speed of convergence when continuously training on a static training set. It prompts us to dynamically select examples for fast convergence as described in the next section.

4.3. INT algorithm

Intending to make the gradient steeper, the greedy functional teaching algorithm in nonparametric teaching chooses examples by recklessly maximizing the gradient norm:

$${\{\boldsymbol{x}_i\}_k}^* = \underset{\{\boldsymbol{x}_i\}_k \subseteq \{\boldsymbol{x}_i\}_N}{\arg\max} \left\|\mathcal{G}(\mathcal{L}, f^*; f_\theta, \{\boldsymbol{x}_i\}_k)\right\|_{\mathcal{H}}, \tag{20}$$

where $\mathcal{G}(\mathcal{L}, f^*; f_\theta, \{\boldsymbol{x}_i\}_k) = \frac{1}{k}\left[\left.\frac{\partial \mathcal{L}}{\partial f}\right|_{f_\theta, \boldsymbol{x}_i}\right]_k^T \cdot \left[K(\boldsymbol{x}_i, \cdot)\right]_k$ and $k \leq N$ denotes the size of selected training set. Drawing from the consistency between an MLP and a nonparametric learner, as explored in Section 4.1 (Geifman et al., 2020;

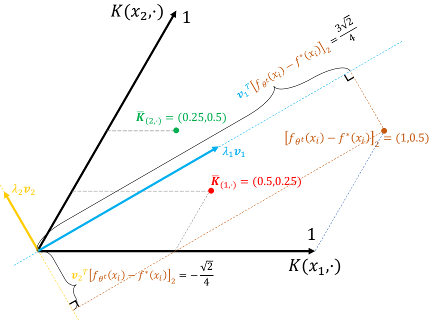

$K(x_2,\cdot)$
1
$v_1^T[f_{\theta^t}(x_i)-f^*(x_i)]_2=\frac{3\sqrt{2}}{4}$
$\bar{K}_{(2,\cdot)}=(0.25,0.5)$
$[f_{\theta^t}(x_i)-f^*(x_i)]_2=(1,0.5)$
$\lambda_1 v_1$
$\lambda_2 v_2$
$\bar{K}_{(1,\cdot)}=(0.5,0.25)$
1
$K(x_1,\cdot)$
$v_2^T[f_{\theta^t}(x_i)-f^*(x_i)]_2=-\frac{\sqrt{2}}{4}$

Figure 2. An illustration of the spectral understanding in a 2D function coordinate system (*i.e.*, RKHS) with the $\{K(\boldsymbol{x}_i,\cdot)\}_2$ basis. The basis can be non-orthogonal if $K(\boldsymbol{x}_i,\boldsymbol{x}_j)\neq 0$ for $i\neq j$. The coordinate of $f_{\theta^t}-f^*$ represents its projection on each axis, which is given by $\langle(f_{\theta^t}-f^*),[K(\boldsymbol{x}_i,\cdot)]_2^T\rangle_{\mathcal{H}}=[f_{\theta^t}(\boldsymbol{x}_i)-f^*(\boldsymbol{x}_i)]_2^T$, and that of $K(\boldsymbol{x}_\dagger,\cdot)$ is $\langle K(\boldsymbol{x}_\dagger,\cdot),[K(\boldsymbol{x}_i,\cdot)]_2^T\rangle_{\mathcal{H}}=[K(\boldsymbol{x}_\dagger,\boldsymbol{x}_i)]_2^T$, which is stored in the †-th row of $\boldsymbol{K}$. Assuming $\bar{\boldsymbol{K}}=\begin{bmatrix}0.5 & 0.25\\ 0.25 & 0.5\end{bmatrix}$, the eigenvalues and the respective eigenvectors can be computed as $\lambda_1=0.75,\lambda_2=0.25$ and $\boldsymbol{v}_1=(\frac{\sqrt{2}}{2},\frac{\sqrt{2}}{2})^T,\boldsymbol{v}_2=(-\frac{\sqrt{2}}{2},\frac{\sqrt{2}}{2})^T$, respectively. Assuming $[f_{\theta^t}(\boldsymbol{x}_i)-f^*(\boldsymbol{x}_i)]_2$ equals $(1,0.5)$, its first and second principal component projections are $\frac{3\sqrt{2}}{4}$ and $-\frac{\sqrt{2}}{4}$, respectively. Moreover, the discrepancy between f_{θ^t} and f^* diminishes at a rate of $e^{-\frac{3\eta t}{4}}$ and $e^{-\frac{\eta t}{4}}$ for the first and second principal components, respectively.

Chen & Xu, 2020), we present the INT algorithm that also aims to increase the steepness of gradients. Differently, INT circumvents the potentially cumbersome computation of $\|K(\boldsymbol{x}_i,\cdot)\|_{\mathcal{H}}$ in $\|\mathcal{G}\|_{\mathcal{H}}$ by utilizing a projection view. To be specific, for $i\in\mathbb{N}_N$, $\frac{\partial\mathcal{L}}{\partial f}|_{f_\theta,\boldsymbol{x}_i}$ can be seen as the component of $\frac{\partial\mathcal{L}}{\partial f}|_{f_\theta}$ projected onto the corresponding element of the basis $\{K(\boldsymbol{x}_i,\cdot)\}_N$. Hence, the gradient represents the total sum of the updates, each weighted by $\frac{\partial\mathcal{L}}{\partial f}|_{f_\theta,\boldsymbol{x}_i}$, throughout $\{K(\boldsymbol{x}_i,\cdot)\}_k$, which is associated with the selected examples (Wright, 2015). Consequently, steepening the gradient simply requires maximizing the coefficient $\frac{\partial\mathcal{L}}{\partial f}|_{f_\theta,\boldsymbol{x}_i}$, bypassing the need to calculate $\|K(\boldsymbol{x}_i,\cdot)\|_{\mathcal{H}}$. This indicates that selecting examples that enlarge $\left|\frac{\partial\mathcal{L}}{\partial f}|_{f_\theta,\boldsymbol{x}}\right|$ or those which correspond to larger components of $\frac{\partial\mathcal{L}}{\partial f}|_{f_\theta}$ can be sufficient to increase the gradient, which means

$$\{\boldsymbol{x}_i\}_k{}^*=\underset{\{\boldsymbol{x}_i\}_k\subseteq\{\boldsymbol{x}_i\}_N}{\arg\max}\left\|\left[\left.\frac{\partial\mathcal{L}}{\partial f}\right|_{f_\theta,\boldsymbol{x}_i}\right]_k\right\|_2. \tag{21}$$

From a functional perspective, when dealing with a convex loss functional $\mathcal{L}$, the norm of the partial derivative of $\mathcal{L}$ with respect to f at f_θ, denoted as $\|\frac{\partial\mathcal{L}}{\partial f}|_{f_\theta}\|_{\mathcal{H}}$, is positively correlated with$\|f_\theta-f^*\|_{\mathcal{H}}$; as f_θ gradually approaches f^*, $\|\frac{\partial\mathcal{L}}{\partial f}|_{f_\theta}\|_{\mathcal{H}}$ decrease (Boyd et al., 2004; Coleman, 2012). This relationship becomes particularly significant when $\mathcal{L}$ is strongly convex with a larger strong convexity constant (Kakade & Tewari, 2008; Arjevani et al., 2016). Based on these findings, the INT algorithm selects examples by

$$\{\boldsymbol{x}_i\}_k{}^*=\underset{\{\boldsymbol{x}_i\}_k\subseteq\{\boldsymbol{x}_i\}_N}{\arg\max}\left\|[f_\theta(\boldsymbol{x}_i)-f^*(\boldsymbol{x}_i)]_k\right\|_2. \tag{22}$$

Pseudo code is in Algorithm 1.

Algorithm 1 Implicit Neural Teaching

```
Input: Target signal $f^*$, initial MLP $f_{\theta^0}$, the size of selected training size $k\leq N$, small constant $\epsilon>0$ and maximal iteration number $T$.
Set $f_{\theta^t}\leftarrow f_{\theta^0}$, $t=0$.
while $t\leq T$ and $\|[f_{\theta^t}(\boldsymbol{x}_i)-f^*(\boldsymbol{x}_i)]_N\|_2\geq\epsilon$ do
    The teacher selects $k$ teaching examples:
    /* Examples corresponding to the $k$ largest $|f_{\theta^t}(\boldsymbol{x}_i)-f^*(\boldsymbol{x}_i)|$.  */
    $\{\boldsymbol{x}_i\}_k{}^*=\arg\max_{\{\boldsymbol{x}_i\}_k\subseteq\{\boldsymbol{x}_i\}_N}\|[f_{\theta^t}(\boldsymbol{x}_i)-f^*(\boldsymbol{x}_i)]_k\|_2$.
    Provide $\{\boldsymbol{x}_i\}_k{}^*$ to the MLP learner.
    The learner updates $f_{\theta^t}$ based on received $\{\boldsymbol{x}_i\}_k{}^*$:
    // Parameter-based gradient descent.
    $\theta^t\leftarrow\theta^t-\frac{\eta}{k}\sum_{\boldsymbol{x}_i\in\{\boldsymbol{x}_i\}_k{}^*}\nabla_\theta\mathcal{L}(f_{\theta^t}(\boldsymbol{x}_i),f^*(\boldsymbol{x}_i))$.
    Set $t\leftarrow t+1$.
end
```

When considering the square loss commonly employed in INR, the aforementioned correlation can be represented as $\|\frac{\partial\mathcal{L}}{\partial f}|_{f_\theta}\|_{\mathcal{H}}\propto\|f_\theta-f^*\|_{\mathcal{H}}$. Besides, it is intriguing that the INT algorithm aligns with the applied variant of the greedy functional teaching algorithm, wherein it is necessary for $\|K(\boldsymbol{x}_i,\cdot)\|_{\mathcal{H}}$ to be uniform or $\|K(\boldsymbol{x}_i,\cdot)\|_{\mathcal{H}}=1$ for all $\boldsymbol{x}_i$ (Zhang et al., 2023b). The convergence analysis of the INT algorithm also aligns with that of the greedy functional teaching algorithm obtained in Zhang et al., 2023b;a.

With the spectral analysis in Section 4.2, a deeper understanding of INT follows. First, we define the entire space as the one spanned by the basis corresponding to the whole training set $\{K(\boldsymbol{x}_i,\cdot)\}_N$. Similarly, $\{K(\boldsymbol{x}_i,\cdot)\}_k\subseteq\{K(\boldsymbol{x}_i,\cdot)\}_N$ spans subspaces associated with the selected examples. The eigenvalue of the transformation from the entire space to the subspace of concern (*i.e.*, spanned by $\{K(\boldsymbol{x}_i,\cdot)\}_k$ associated with selected examples) is one, while it is zero for the subspace without interest (Watanabe & Katagiri, 1995; Burgess & Van Veen, 1996). The spectral understanding indicates that f_{θ^t} approaches f^* swiftly at the early stage within the current subspace, owing to the large eigenvalues (Jacot et al., 2018). Hence, the INT algorithm can be interpreted as dynamically altering the subspace of interest to fully exploit the period when f_{θ^t} approaches f^* rapidly. Meanwhile, by selecting examples based on Equation 22, the subspace of interest is precisely

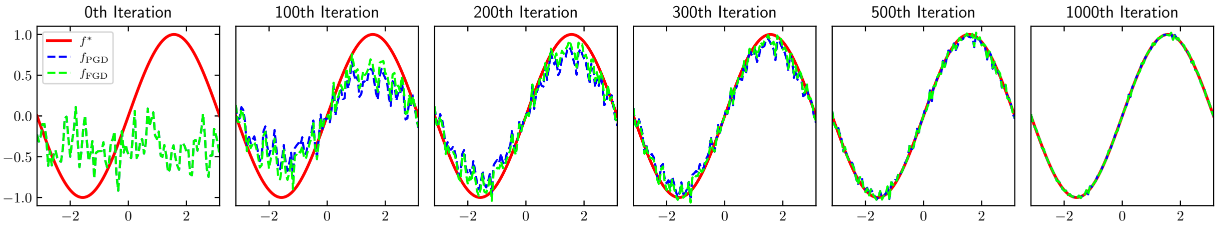


Figure 3. Training dynamics of f using PGD and FGD. Apparently, f_{PGD} closely follows f_{FGD}, empirically showing the evolution consistency between PGD training and FGD training.

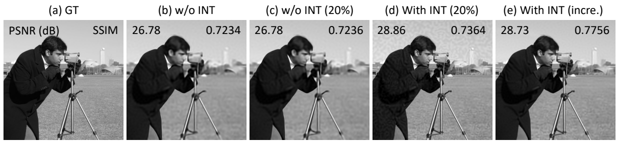


Figure 4. Reconstruction quality of SIREN. (b) trains SIREN without (w/o) INT using all pixels. (c) trains it w/o INT using 20% randomly selected pixels. (d) trains it using INT of 20% selection rate. (e) trains it using progressive INT (*i.e.*, increasing selection rate progressively from 20% to 100%).

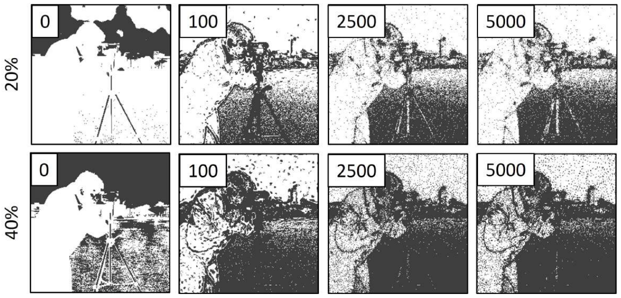


Figure 5. Progression of INT selected pixels (marked as black) at corresponding iterations when training with INT 20% (top) and 40% (bottom).

the one where f_{θ^t} remains significantly distant from f^*. In a nutshell, the INT algorithm, by dynamically altering the subspace of interest, not only maximizes the benefits of the fast convergence stage but also updates f_{θ^t} in the most urgent direction towards f^*, thereby saving computational resources compared to training on the entire dataset.

5. Experiments and Results

We begin by using a synthetic signal to empirically show the evolution consistency between parameter-based gradient descent (PGD) and functional gradient descent (FGD). Next, we assess the behavior of INT on a toy image-fitting instance and explore diverse algorithms with different INT frequencies and ratios. Lastly, we validate the INT efficiency in multiple modalities such as audio (-31.63% training time), images (-38.88%), and 3D shapes (-35.54%), while upkeeping its reconstruction quality. Detailed settings are given in Appendices C.

Synthetic 1D signal. For an intuitive visualization, we utilize a synthetic 1D signal and present the training dynamics of f obtained through both PGD and FGD. Specifically, the signal (*i.e.*, the target function) is $f^*(x) = \sin(x)$ where $x \in \{x_i\}_{100}$ and is uniformly distributed in the range of $[-\pi, \pi]$. The function corresponding to PGD is obtained by inputting $\{x_i\}_{100}$ into the Fourier Feature network (FFN) trained using PGD, while the function corresponding to FGD is represented by dense points of the nonparametric function updated using FGD. As depicted in Figure 3, f^* is well fitted by both PGD and FGD. Moreover, the function obtained through PGD closely mirrors the one obtained through FGD. This observation indicates the consistency in the evolution of the function through both PGD and FGD, suggesting that teaching an overparameterized MLP aligns with teaching a nonparametric target function.

Toy 2D Cameraman fitting. In practice, SIREN (Sitzmann et al., 2020b) is commonly used to encode various modalities of signal such as images. Here, we test the effectiveness of INT in a real-life setting where a SIREN model is used to fit the Cameraman image (Van der Walt et al., 2014). We compare the reconstruction quality of SIREN trained with INT of 20% selection rate (*i.e.*, the size of selected training set is at 20% of the entire set comprised of all pixels) against that trained without INT, (*i.e.*, using all pixels) and that trained with random sampling at the rate of 20% at each iteration. INT training results in a higher PSNR and SSIM but exhibits visible artifacts in the background. As shown in Figure 5 which presents the selected pixels throughout training, we hypothesize that this is due to the over-emphasis of the INT on “boundary” pixels where color changes are usually abrupt and hence loss values are larger, leading to an overfitting on the background pixels. On the contrary, using a higher selection rate permits INT to select more examples on the flat surfaces (background), which serves as a regularizer to alleviate the artifacts. Thus, we train an additional SIREN with a progressively increasing INT selection rate from 20% to nearly 100%, which achieves superior reconstruction quality without the artifacts.

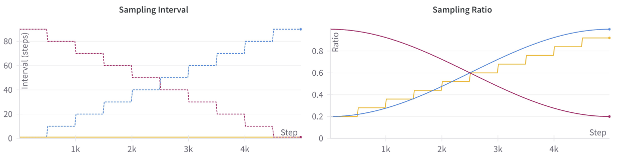


Figure 6. Selecting ratio and interval of various INT algorithms. (Left) Red - decremental; Blue - incremental; Yellow - Dense. (Right) Red - R-Cosine; Blue - Cosine; Yellow - Incremental.

Ratio	Interval	Time (s)	PSNR (dB)↑	SSIM↑
-	-	345.22	35.95±1.89	0.935±0.03
Cosine	Dense	337.00	36.39±2.40	0.941±0.02
Cosine	Incremental	227.84	36.61±2.55	0.942±0.02
R-Cosine	Dense	346.64	35.18±1.44	0.920±0.02
R-Cosine	Decremental	225.30	33.56±2.53	0.894±0.03
Incremental	Dense	468.01	36.84±2.70	**0.946±0.02**
Incremental	**Incremental**	**211.04**	**37.04±2.51**	**0.946±0.02**

Table 1. Performance and training time for different INT strategies on Kodak dataset. The first line ("-" in both Ratio and Interval) corresponds to training without INT.

INT with different frequencies and ratios. While using INT can train an INR with fewer examples without sacrificing reconstruction quality, it should be noted that each selecting process requires inferencing all data through the network to rank the difference between the outputs and f^* from higher to lower, which is rather time-consuming. This counters the effect of reducing training time that could originally be brought by the reduction in training examples. Consequently, we follow the observation that increasing the selection ratio leads to increasing overlaps between each consecutive INT selection, and thus devise several INT algorithms that space out the INT frequency (*i.e.*, selecting frequency) and vary the INT ratio (*i.e.*, sizes of selected training sets) dynamically throughout training. Namely, for selecting ratio, we test *constant* ratio, step-wise *increment* of ratio at fixed intervals, and gradually increasing/decreasing the ratio in a *cosine* annealing manner. On the other hand, for sample interval, we test *densely* sampling per iteration, and step-wise *increment*/*decrement* of sampling intervals between 1 and 100 steps.

Figure 6 visualizes the various algorithms we tested against each other. In particular, as presented in Table 1, our experiment on a subset of 8 representative images from the Kodak dataset (Eastman Kodak Company, 1999) shows that combining an incrementally increasing sampling ratio with an incrementally increasing sampling interval leads to the best performance in terms of both training speed and construction quality. We also want to highlight the severe degradation in reconstruction quality that comes with training an INR via decremental sampling ratio and intervals (comparing rows 4&5 in Table 1). We attribute this to the nature of INRs to progressively learn signals of lower to higher frequencies as shown in (Rahaman et al., 2019) while the decremental strategy goes against it. Specifically, at the beginning of training, the MLP may not be able to learn all the information provided by densely sampled examples. But towards the end of training when the MLP is trying to fit the remaining details of the signal, the decremental INT algorithm provides sparser and sparser samples that do not get updated frequently. This serves as a counter-example that explains the effectiveness of utilizing incremental INT for training general INRs, as we shall see in the following section.

INT	Modality	Time (s)	PSNR(dB) / IoU(%) ↑
✗	Audio	23.05	48.38±3.50
	Image	345.22	36.09±2.51
	Megapixel	16.78K	31.82
	3D Shape	144.58	97.07±0.84
✓	Audio	15.76 (-31.63%)	48.15±3.39
	Image	211.04 (-38.88%)	36.97±3.59
	Megapixel	11.87K (-29.26%)	33.01
	3D Shape	93.19 (-35.54%)	96.68±0.83

Table 2. Signal fitting results for different data modalities. The encoding time is measured excluding data I/O latency.

INT on multiple real-world modalities. To demonstrate the practicality of INT in real-world applications, we conduct experiments on signal fitting tasks across datasets of various modalities, including 1D audio (Librispeech (Panayotov et al., 2015)), 2D images (Kodak (Eastman Kodak Company, 1999)), megapixel images (Pluto (NASA, 2018)), and 3D shapes (Stanford 3D Scanning Repository (Stanford Computer Graphics Laboratory, 2007)). We selected the optimal strategy from Table 1 (*i.e.*, step-wise increments of both sampling ratio and intervals) as the default INT setting and evaluated it against the baseline without INT. The implementation details of the experiment for each modality can be found in Appendix C. As shown in Table 2, it is evident that INT can effectively speed up encoding for all modalities, ranging from 1.41× to 1.64×, with minimal degradation in performance ($<$ 1dB PSNR or $<$ 1% IoU). In the case of 2D images, the PSNR with INT even improves from 36.09dB to 36.97dB with near 40% decrease in training time. We also highlight the results for fitting 3D shapes and megapixel Pluto image (8192×8192), which instead requires mini-batch INT (Zhang et al., 2023a) due to hardware constraints. That is, for each iteration of optimization, we randomly sample a subset of points from the training set and run the INT algorithm to train our model. We make sure that all pixels in the image are sampled for each epoch. This serves as an analogous training procedure to combining stochastic gradient descent with INT and presents the robustness of our INT algorithms in improving training efficiencies.

6. Concluding Remarks and Future Work

This paper has proposed Implicit Neural Teaching (INT), a novel paradigm that enhances the learning efficiency of implicit neural representation (INR) through nonparametric machine teaching. Using an overparameterized multilayer perceptron (MLP) to fit a given signal, INT reduces the wall-clock time for learning INR by over 30% as demonstrated by extensive experiments. Moreover, INT establishes a theoretically rich connection between the evolution of an MLP using parameter-based gradient descent and that of a function using functional gradient descent in nonparametric teaching. This bridge between nonparametric teaching and MLP training readily expands the applicability of nonparametric teaching in the realm of deep learning.

Moving forward, it could be more intriguing to explore other practical utilities related to INT towards data efficiency (Henaff, 2020; Touvron et al., 2021; Arandjelović & Zisserman, 2021; Müller et al., 2022). This will involve developing a deeper theoretical understanding of INT, with the neural tangent kernel playing a crucial role. Additionally, exploring more efficient example selection algorithms tailored to specific tasks, such as fine-tuning and prompt training in large language models, holds promise for future advancements.

Broader Impact

Implicit neural representation (INR) has emerged as a promising paradigm in vision data representation, view synthesis and signal compression, domains with significant societal impacts, for its ability of representing discrete signals continuously. This work focuses on enhancing the training efficiency of INR via a novel nonparametric teaching perspective, which can bring positive impacts to INR-related fields and society.

Meanwhile, this work connects nonparametric teaching to MLP training, which expands the applicability of nonparametric teaching towards deep learning. Thus, it also makes positive contributions to the community of machine teaching.

Lastly, we are confident that the proposed framework, Implicit Neural Teaching (INT), is highly relevant for enhancing data efficiency and has broader applicability to machine learning tasks, especially in scenarios where the target is known and “overfitting” is desired, as exhibited in INRs and nonparametric teaching.

Acknowledgements

We thank all anonymous reviewers for their constructive feedback to improve our paper. This work was supported by the Theme-based Research Scheme (TRS) project T45-701/22-R, and in part by ACCESS – AI Chip Center for Emerging Smart Systems, sponsored by InnoHK funding, Hong Kong SAR.

References

Alfeld, S., Zhu, X., and Barford, P. Explicit defense actions against test-set attacks. In *AAAI*, 2017.

Arandjelović, R. and Zisserman, A. Nerf in detail: Learning to sample for view synthesis. *arXiv preprint arXiv:2106.05264*, 2021.

Arbel, M., Korba, A., Salim, A., and Gretton, A. Maximum mean discrepancy gradient flow. In *NeurIPS*, 2019.

Arjevani, Y., Shalev-Shwartz, S., and Shamir, O. On lower and upper bounds in smooth and strongly convex optimization. *The Journal of Machine Learning Research*, 17 (1):4303–4353, 2016.

Atzmon, M. and Lipman, Y. Sal: Sign agnostic learning of shapes from raw data. In *CVPR*, 2020.

Bietti, A. and Mairal, J. On the inductive bias of neural tangent kernels. In *NeurIPS*, 2019.

Bietti, A., Mialon, G., Chen, D., and Mairal, J. A kernel perspective for regularizing deep neural networks. In *ICML*, 2019.

Boyd, S., Boyd, S. P., and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Burgess, K. A. and Van Veen, B. D. Subspace-based adaptive generalized likelihood ratio detection. *IEEE Transactions on Signal Processing*, 44(4):912–927, 1996.

Chen, H., Yang, H., Fitzmeyer, S., and Hao, C. Rapid-inr: Storage efficient cpu-free dnn training using implicit neural representation. In *ICCAD*, 2023.

Chen, L. and Xu, S. Deep neural tangent kernel and laplace kernel have the same rkhs. In *ICLR*, 2020.

Coleman, R. *Calculus on normed vector spaces*. Springer Science & Business Media, 2012.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. *Introduction to algorithms*. MIT press, 2022.

Dou, X. and Liang, T. Training neural networks as learning data-adaptive kernels: Provable representation and approximation benefits. *Journal of the American Statistical Association*, 116(535):1507–1520, 2021.

Dupont, E., Goliński, A., Alizadeh, M., Teh, Y. W., and Doucet, A. Coin: Compression with implicit neural representations. In *ICLR Neural Compression Workshop*, 2021.

Dupont, E., Kim, H., Eslami, S., Rezende, D., and Rosenbaum, D. From data to functa: Your data point is a function and you can treat it like one. In *ICML*, 2022.

Eastman Kodak Company. Kodak lossless true color image suite. http://r0k.us/graphics/kodak/, 1999. [Accessed 14-08-2023].

Gao, R., Cai, T., Li, H., Hsieh, C.-J., Wang, L., and Lee, J. D. Convergence of adversarial training in overparametrized neural networks. In *NeurIPS*, 2019.

Geifman, A., Yadav, A., Kasten, Y., Galun, M., Jacobs, D., and Ronen, B. On the similarity between the laplace and neural tangent kernels. In *NeurIPS*, 2020.

Gelfand, I. M., Silverman, R. A., et al. *Calculus of variations*. Courier Corporation, 2000.

Godunov, S. K. *Ordinary differential equations with constant coefficient*, volume 169. American Mathematical Soc., 1997.

Grattarola, D. and Vandergheynst, P. Generalised implicit neural representations. In *NeurIPS*, 2022.

Graves, A., Bellemare, M. G., Menick, J., Munos, R., and Kavukcuoglu, K. Automated curriculum learning for neural networks. In *ICML*, 2017.

Gropp, A., Yariv, L., Haim, N., Atzmon, M., and Lipman, Y. Implicit geometric regularization for learning shapes. In *ICML*, 2020.

Hall, B. C. *Quantum theory for mathematicians*. Springer, 2013.

Hartman, P. *Ordinary differential equations*. SIAM, 2002.

Henaff, O. Data-efficient image recognition with contrastive predictive coding. In *ICML*, 2020.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *NeurIPS*, 2018.

Kakade, S. M. and Tewari, A. On the generalization ability of online strongly convex programming algorithms. In *NeurIPS*, 2008.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Kuk, A. Y. Asymptotically unbiased estimation in generalized linear models with random effects. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 57(2):395–407, 1995.

Lax, P. D. *Functional analysis*, volume 55. John Wiley & Sons, 2002.

Lee, J., Xiao, L., Schoenholz, S., Bahri, Y., Novak, R., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. In *NeurIPS*, 2019.

Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867, 1993.

Li, J. C. L., Liu, C., Huang, B., and Wong, N. Learning spatially collaged fourier bases for implicit neural representation. In *AAAI*, 2024a.

Li, J. C. L., Luo, S. T. S., Xu, L., and Wong, N. Asmr: Activation-sharing multi-resolution coordinate networks for efficient inference. In *ICLR*, 2024b.

Li, Z., Wang, H., and Meng, D. Regularize implicit neural representation by itself. In *CVPR*, 2023.

Lindell, D. B., Van Veen, D., Park, J. J., and Wetzstein, G. Bacon: Band-limited coordinate networks for multiscale scene representation. In *CVPR*, 2022.

Liu, Q. Stein variational gradient descent as gradient flow. In *NeurIPS*, 2017.

Liu, Q. and Wang, D. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *NeurIPS*, 2016.

Liu, W., Dai, B., Humayun, A., Tay, C., Yu, C., Smith, L. B., Rehg, J. M., and Song, L. Iterative machine teaching. In *ICML*, 2017.

Liu, W., Dai, B., Li, X., Liu, Z., Rehg, J., and Song, L. Towards black-box iterative machine teaching. In *ICML*, 2018.

Loshchilov, I. and Hutter, F. Online batch selection for faster training of neural networks. In *ICLR Workshop*, 2015.

Ma, Y., Zhang, X., Sun, W., and Zhu, J. Policy poisoning in batch reinforcement learning and control. In *NeurIPS*, 2019.

Martin-Brualla, R., Radwan, N., Sajjadi, M. S., Barron, J. T., Dosovitskiy, A., and Duckworth, D. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *CVPR*, 2021.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2021.

Mindermann, S., Brauner, J. M., Razzak, M. T., Sharma, M., Kirsch, A., Xu, W., Höltgen, B., Gomez, A. N., Morisot, A., Farquhar, S., et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *ICML*, 2022.

Molaei, A., Aminimehr, A., Tavakoli, A., Kazerouni, A., Azad, B., Azad, R., and Merhof, D. Implicit neural representation in medical imaging: A comparative survey. In *ICCV*, 2023.

Müller, T., Evans, A., Schied, C., and Keller, A. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4): 1–15, 2022.

NASA. True colors of pluto. https://solarsystem.nasa.gov/resources/933/true-colors-of-pluto/?category=planets/dwarf-planets_pluto, 2018.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. Librispeech: an asr corpus based on public domain audio books. In *ICASSP*, 2015.

Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019.

Pistilli, F., Valsesia, D., Fracastoro, G., and Magli, E. Signal compression via neural implicit representations. In *ICASSP*, 2022.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. On the spectral bias of neural networks. In *ICML*, 2019.

Rakhsha, A., Radanovic, G., Devidze, R., Zhu, X., and Singla, A. Policy teaching via environment poisoning: Training-time adversarial attacks against reinforcement learning. In *ICML*, 2020.

Reddy, P., Zhang, Z., Wang, Z., Fisher, M., Jin, H., and Mitra, N. A multi-implicit neural representation for fonts. In *NeurIPS*, 2021.

Ruder, S. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

Schölkopf, B., Smola, A. J., Bach, F., et al. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2002.

Schwarz, J. R., Tack, J., Teh, Y. W., Lee, J., and Shin, J. Modality-agnostic variational compression of implicit neural representations. In *ICML*, 2023.

Shen, Z., Wang, Z., Ribeiro, A., and Hassani, H. Sinkhorn barycenter via functional gradient descent. In *NeurIPS*, 2020.

Singla, A., Bogunovic, I., Bartók, G., Karbasi, A., and Krause, A. Near-optimally teaching the crowd to classify. In *ICML*, 2014.

Sitzmann, V., Chan, E., Tucker, R., Snavely, N., and Wetzstein, G. Metasdf: Meta-learning signed distance functions. In *NeurIPS*, 2020a.

Sitzmann, V., Martel, J., Bergman, A., Lindell, D., and Wetzstein, G. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020b.

Stanford Computer Graphics Laboratory. The stanford 3d scanning repository. https://graphics.stanford.edu/data/3Dscanrep/, 2007.

Strümpler, Y., Postels, J., Yang, R., Gool, L. V., and Tombari, F. Implicit neural representations for image compression. In *ECCV*, 2022.

Tack, J., Kim, S., Yu, S., Lee, J., Shin, J., and Schwarz, J. R. Learning large-scale neural fields via context pruned meta-learning. In *NeurIPS*, 2023.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. In *NeurIPS*, 2020.

Tancik, M., Mildenhall, B., Wang, T., Schmidt, D., Srinivasan, P. P., Barron, J. T., and Ng, R. Learned initializations for optimizing coordinate-based neural representations. In *CVPR*, 2021.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

Van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., and Yu, T. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.

Wang, P. and Vasconcelos, N. A machine teaching framework for scalable recognition. In *ICCV*, 2021.

Wang, P., Nagrecha, K., and Vasconcelos, N. Gradient-based algorithms for machine teaching. In *CVPR*, 2021.

Wang, P., Fan, Z., Chen, T., and Wang, Z. Neural implicit dictionary learning via mixture-of-expert training. In *ICML*, 2022.

Watanabe, H. and Katagiri, S. Discriminative subspace method for minimum error pattern recognition. In *IEEE Workshop on Neural Networks for Signal Processing*, 1995.

Wright, S. J. Coordinate descent algorithms. *Mathematical programming*, 151(1):3–34, 2015.

Xie, S., Zhu, H., Liu, Z., Zhang, Q., Zhou, Y., Cao, X., and Ma, Z. Diner: Disorder-invariant implicit neural representation. In *CVPR*, 2023.

Yüce, G., Ortiz-Jiménez, G., Besbinar, B., and Frossard, P. A structured dictionary perspective on implicit neural representations. In *CVPR*, 2022.

Zhang, C., Cao, X., Liu, W., Tsang, I., and Kwok, J. Nonparametric teaching for multiple learners. In *NeurIPS*, 2023a.

Zhang, C., Cao, X., Liu, W., Tsang, I., and Kwok, J. Nonparametric iterative machine teaching. In *ICML*, 2023b.

Zhou, Y., Nelakurthi, A. R., and He, J. Unlearn what you have learned: Adaptive crowd teaching with exponentially decayed memory learners. In *SIGKDD*, 2018.

Zhu, X. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *AAAI*, 2015.

Zhu, X., Singla, A., Zilles, S., and Rafferty, A. N. An overview of machine teaching. *arXiv preprint arXiv:1801.05927*, 2018.

Appendix

A. Additional Discussions

Neural Tangent Kernel (NTK) By substituting the parameter evolution

$$\frac{\partial \theta^t}{\partial t} = -\frac{\eta}{N}\left[\frac{\partial \mathcal{L}}{\partial f_\theta}\bigg|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \cdot \left[\frac{\partial f_\theta}{\partial \theta}\bigg|_{\boldsymbol{x}_i,\theta^t}\right]_N \tag{23}$$

into the first-order approximation term $(*)$ of Equation 10, it obtains

$$\begin{aligned}(*) &= \left\langle \frac{\partial f_\theta}{\partial \theta}\bigg|_{\cdot,\theta^t}, -\frac{\eta}{N}\left[\frac{\partial \mathcal{L}}{\partial f_\theta}\bigg|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \cdot \left[\frac{\partial f_\theta}{\partial \theta}\bigg|_{\boldsymbol{x}_i,\theta^t}\right]_N \right\rangle \\ &= -\frac{\eta}{N}\left[\frac{\partial \mathcal{L}}{\partial f_\theta}\bigg|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \cdot \left\langle \frac{\partial f_\theta}{\partial \theta}\bigg|_{\cdot,\theta^t}, \left[\frac{\partial f_\theta}{\partial \theta}\bigg|_{\boldsymbol{x}_i,\theta^t}\right]_N \right\rangle \\ &= -\frac{\eta}{N}\left[\frac{\partial \mathcal{L}}{\partial f_\theta}\bigg|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \cdot \left[\left\langle \frac{\partial f_\theta}{\partial \theta}\bigg|_{\cdot,\theta^t}, \frac{\partial f_\theta}{\partial \theta}\bigg|_{\boldsymbol{x}_i,\theta^t} \right\rangle\right]_N \\ &= -\frac{\eta}{N}\left[\frac{\partial \mathcal{L}}{\partial f_\theta}\bigg|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \cdot \left[K_{\theta^t}(\boldsymbol{x}_i,\cdot)\right]_N, \end{aligned} \tag{24}$$

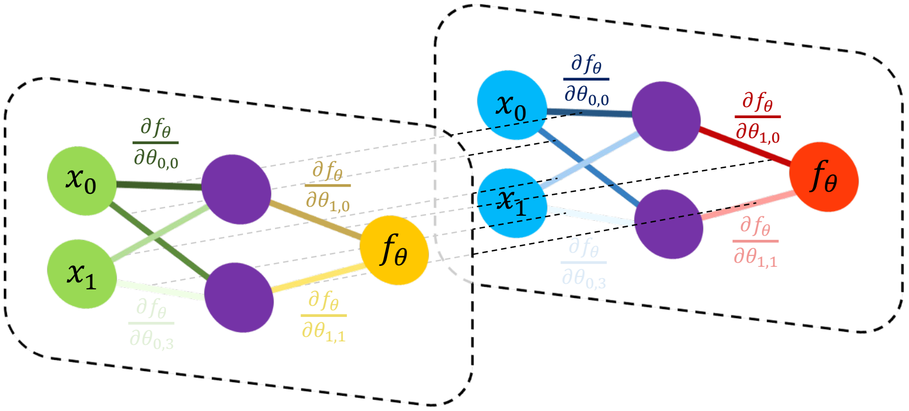
which derives Equation 11 as

$$\frac{\partial f_{\theta^t}}{\partial t} = -\frac{\eta}{N}\left[\frac{\partial \mathcal{L}}{\partial f_\theta}\bigg|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \cdot \left[K_{\theta^t}(\boldsymbol{x}_i,\cdot)\right]_N + o\left(\frac{\partial \theta^t}{\partial t}\right), \tag{25}$$

and K_{θ^t} is referred to as neural tangent kernel (NTK) (Jacot et al., 2018). Figure 7 provides a visual representation that explains the calculation process of NTK in a clear and understandable way. Informally speaking, studying how a model behaves by focusing on the model itself rather than its parameters typically entails the use of kernel functions.

It can be observed that the quantity $\frac{\partial f_\theta}{\partial \theta}\big|_{\cdot,\theta^t}$, present in $K_{\theta^t}(\boldsymbol{x}_i,\cdot) = \left\langle \frac{\partial f_\theta}{\partial \theta}\big|_{\cdot,\theta^t}, \frac{\partial f_\theta}{\partial \theta}\big|_{\boldsymbol{x}_i,\theta^t}\right\rangle$, represents the partial derivative of the MLP with respect to its parameters, determined by both the structure and specific θ^t, but independent of the input. On the other hand, $\frac{\partial f_\theta}{\partial \theta}|_{\boldsymbol{x}_i,\theta^t}$originates from the parameter evolution, which relies not only on the MLP structure and specific θ^t, but also on the input example. Assuming the input of $\frac{\partial f_\theta}{\partial \theta}|_{\boldsymbol{x}_i,\theta^t}$ is not known, the NTK becomes $K_{\theta^t}(\cdot,\cdot)$. On the other hand, if we specify $\boldsymbol{x}_j$ as the input for $\frac{\partial f_\theta}{\partial \theta}|_{\cdot,\theta^t}$, NTK becomes a scalar as $K_{\theta^t}(\boldsymbol{x}_i,\boldsymbol{x}_j) = \langle \frac{\partial f_\theta}{\partial \theta}|_{\boldsymbol{x}_j,\theta^t}, \frac{\partial f_\theta}{\partial \theta}|_{\boldsymbol{x}_i,\theta^t}\rangle$. This indicates that the NTK is a bivariate function represented by $\mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$, and this form aligns with the kernel used in functional gradient descent. By feeding the input example $\boldsymbol{x}_i$, one coordinate of K_{θ^t} is fixed, causing the MLP to update along $K_{\theta^t}(\boldsymbol{x}_i,\cdot)$ based on the magnitude of $\frac{\partial f_\theta}{\partial \theta}|_{\boldsymbol{x}_i,\theta^t}$, which is consistent with the underlying mechanism of functional gradient descent. In a nutshell, NTK and the canonical kernel not only maintain consistency in their mathematical representation, but also exhibit alignment in how they influence the evolution of the corresponding MLP. Additionally, Theorem 5 demonstrates the asymptotic relationship between the NTK and the canonical kernel used in functional gradient descent.

Jacot et al., 2018 introduce kernel gradient descent, which can be considered as an extension of parameter-based gradient descent. Although kernel gradient descent appears to bear resemblance to functional gradient descent (Zhang et al., 2023b;a), they fundamentally differ in terms of specific details. In kernel gradient descent, the kernel gradient is derived by incorporating a kernel weighting (Jacot et al., 2018), where the NTK serves as the weight to modify the conventional gradient of a real-valued loss $\mathcal{L}(f(\boldsymbol{x}), y)$ with respect to $f(\boldsymbol{x})$, which is limited to the training set, thus allowing the weighted gradient (kernel gradient) to be extrapolated to values beyond the training set. Differently, functional gradient descent takes a higher-level perspective on the evolution of the MLP in function space (Zhang et al., 2023b;a). Specifically, $f(\boldsymbol{x}) = E_{\boldsymbol{x}}(f)$ represents the result of evaluating the function f at the example $\boldsymbol{x}$, which is defined as the inner product in RKHS between the function f and $K(\boldsymbol{x},\cdot)$ (the corresponding kernel with one argument $\boldsymbol{x}$) based on the reproducing property. By applying the functional chain rule and Fréchet derivative, the functional gradient is derived accordingly.


x_0
x_1
$\frac{\partial f_\theta}{\partial \theta_{0,0}}$
$\frac{\partial f_\theta}{\partial \theta_{0,3}}$
$\frac{\partial f_\theta}{\partial \theta_{1,0}}$
$\frac{\partial f_\theta}{\partial \theta_{1,1}}$
f_θ
x_0
x_1
$\frac{\partial f_\theta}{\partial \theta_{0,0}}$
$\frac{\partial f_\theta}{\partial \theta_{0,3}}$
$\frac{\partial f_\theta}{\partial \theta_{1,0}}$
$\frac{\partial f_\theta}{\partial \theta_{1,1}}$
f_θ

$$\text{NTK} = [\sum_{l=0}^{L}\sum_{p=0}^{P^l}\frac{\partial f_\theta(x)}{\partial \theta_{l,p}}\frac{\partial f_\theta(x)}{\partial \theta_{l,p}}]_{1\times1} = [\frac{\partial f_\theta(x)}{\partial \theta_{0,0}}\frac{\partial f_\theta(x)}{\partial \theta_{0,0}} + \cdots + \frac{\partial f_\theta(x)}{\partial \theta_{0,3}}\frac{\partial f_\theta(x)}{\partial \theta_{0,3}} + \frac{\partial f_\theta(x)}{\partial \theta_{1,0}}\frac{\partial f_\theta(x)}{\partial \theta_{1,0}} + \frac{\partial f_\theta(x)}{\partial \theta_{1,1}}\frac{\partial f_\theta(x)}{\partial \theta_{1,1}}]$$

No. of layers: L No. of weights (edges) per layer: P^l

Figure 7. Graphical illustration of NTK computation.

Due to the discrete nature of computer operations, functional gradient descent relies on dense pairwise points $\{(\boldsymbol{x}_i, K(\boldsymbol{x}_\dagger, \boldsymbol{x}_i))\}_n$ for representing the kernel $K(\boldsymbol{x}_\dagger, \cdot)$, and in order to express f, it is necessary to store all $K_{\boldsymbol{x}_i}$s as dense points, resulting in significant storage requirements. This issue mirrors the challenge encountered when storing discrete signals, and the solution lies in INR, employing overparameterized MLPs to continuously represent functions, eliminating the need for storing dense points by utilizing a relatively small-sized parameter storage. Besides, in terms of evolution, functional gradient descent requires updating all dense points to derive f^t based on the functional gradient that also relies on $K(\boldsymbol{x}_i, \cdot)$, whereas training an MLP only necessitates updating the parameter θ, providing practical convenience compared to the theoretical analysis facilitated by functional gradient descent. This work establishes a correlation between nonparametric teaching and MLP training, which involves training an MLP to represent general functions, thereby increasing the theoretical framework's potential scope for implementation in deep learning.

The Relationship between Nonparametric Teaching, Implicit Neural Teaching, and Parametric Teaching In simpler terms, nonparametric teaching (Zhang et al., 2023b; Ma et al., 2019) offers a comprehensive framework that encompasses other paradigms, where these paradigms can be viewed as special cases with specific kernels. For instance, this paper focuses on implicit neural teaching, which corresponds to a distinct paradigm by specifying the neural tangent kernel, while parametric teaching (Liu et al., 2017; 2018) considers a particular paradigm utilizing a linear kernel. Furthermore, when the MLP is reduced to a single-layer architecture without nonlinear activation functions, it becomes the linear case examined in parametric teaching (Liu et al., 2017; 2018), resulting in a zero remainder in Equation 10. Figure 8 provides a visualization of these relationships.

Solution of ODE for training with a fixed single input If we allow the MLP to evolve based on a single example $\boldsymbol{x}$, we have

$$\frac{\partial f_{\theta^t}}{\partial t} = -\eta(f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x})) \cdot K(\boldsymbol{x}, \cdot). \tag{26}$$

Since $\frac{\partial f^*}{\partial t} = 0$, we can rewrite the above differential equation as:

$$\frac{\partial f_{\theta^t} - f^*}{\partial t} = -\eta(f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x})) \cdot K(\boldsymbol{x}, \cdot). \tag{27}$$

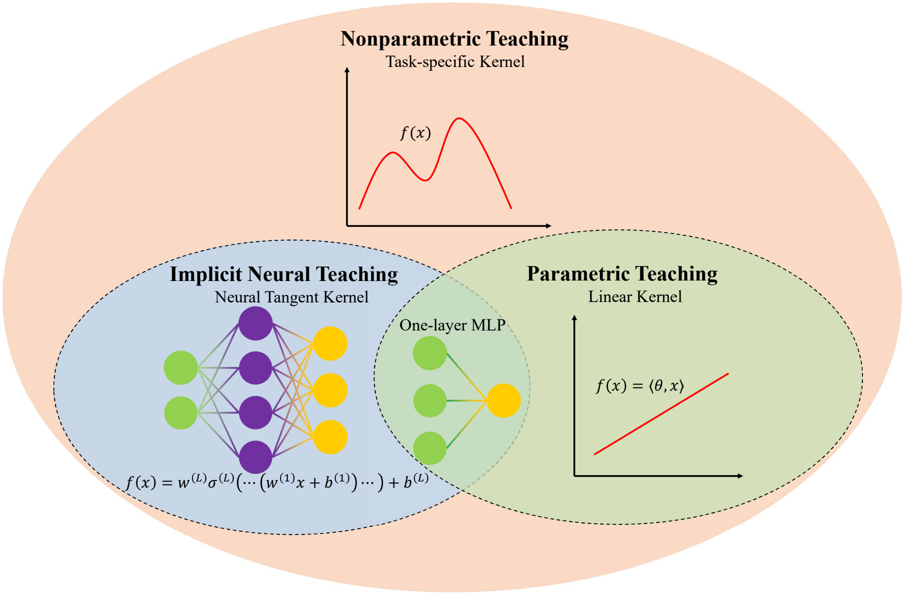


Figure 8. Illustration of the relationship between nonparametric teaching, implicit neural teaching and parametric teaching. Nonparametric teaching deals with general functions corresponding to task-specific kernels. As an instance, implicit neural teaching focuses on neural tangent kernels (Jacot et al., 2018) and is concerned with the functions expressed by an overparameterized MLP. On the other hand, parametric teaching concentrates on parameterized functions of the form $f(x) = \langle\theta, \boldsymbol{x}\rangle$, which is a specific case of nonparametric teaching that uses a linear kernel as the task-specific kernel. Additionally, teaching a one-layer MLP without nonlinear activation functions is essentially equivalent to parametric teaching.

By manipulating both sides of the equation using $\langle K(\boldsymbol{x},\cdot),\cdot\rangle_{\mathcal{H}}$ $(K(\boldsymbol{x},\boldsymbol{x}) \neq 0)$ and rearranging, we can obtain

$$\begin{aligned}
&\mathrm{d}\left(f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x})\right) = -\eta(f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x})) \cdot K(\boldsymbol{x},\boldsymbol{x})\mathrm{d}t \\
&\therefore \frac{\mathrm{d}\left(f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x})\right)}{f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x})} = -\eta K(\boldsymbol{x},\boldsymbol{x})\mathrm{d}t \\
&\therefore \int \frac{\mathrm{d}\left(f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x})\right)}{f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x})} = -\eta K(\boldsymbol{x},\boldsymbol{x}) \int \mathrm{d}t \\
&\therefore \ln|f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x})| = -\eta K(\boldsymbol{x},\boldsymbol{x})t + C.
\end{aligned} \tag{28}$$

When $f_{\theta^t}(\boldsymbol{x})$ approaches $f^*(\boldsymbol{x})$ from below, that is, $f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x}) < 0$, we have

$$\ln\left(f^*(\boldsymbol{x}) - f_{\theta^t}(\boldsymbol{x})\right) = -\eta K(\boldsymbol{x},\boldsymbol{x})t + C. \tag{29}$$

Let t=0, we attain

$$C = \ln\left(f^*(\boldsymbol{x}) - f_{\theta^0}(\boldsymbol{x})\right). \tag{30}$$

Therefore, we have

$$f_{\theta^t}(\boldsymbol{x}) = f^*(\boldsymbol{x}) - e^{-\eta K(\boldsymbol{x},\boldsymbol{x})t}\left(f^*(\boldsymbol{x}) - f_{\theta^0}(\boldsymbol{x})\right). \tag{31}$$

If $f_{\theta^t}(\boldsymbol{x})$ approaches $f^*(\boldsymbol{x})$ from above, which indicates $f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x}) > 0$, we have

$$f_{\theta^t}(\boldsymbol{x}) = f^*(\boldsymbol{x}) + e^{-\eta K(\boldsymbol{x},\boldsymbol{x})t}\left(f_{\theta^0}(\boldsymbol{x}) - f^*(\boldsymbol{x})\right), \tag{32}$$

which is equivalent to the case of $f_{\theta^t}(\boldsymbol{x}) - f^*(\boldsymbol{x}) < 0$ because

$$-e^{-\eta K(\boldsymbol{x},\boldsymbol{x})t}\left(f^*(\boldsymbol{x}) - f_{\theta^0}(\boldsymbol{x})\right) = e^{-\eta K(\boldsymbol{x},\boldsymbol{x})t}\left(f_{\theta^0}(\boldsymbol{x}) - f^*(\boldsymbol{x})\right). \tag{33}$$

Detailed solution procedure of matrix ODE corresponding to Equation 16 The case of $f^*(\boldsymbol{x}) - f_{\theta^t}(\boldsymbol{x}) > 0$. Since $\frac{\partial f^*}{\partial t} = 0$, we can rewrite Equation 16

$$\frac{\partial f_{\theta^t}}{\partial t} = -\frac{\eta}{N} \left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N^T \cdot \left[K(\boldsymbol{x}_i, \cdot) \right]_N \tag{34}$$

as

$$\frac{\partial f_{\theta^t} - f^*}{\partial t} = -\frac{\eta}{N} \left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N^T \cdot \left[K(\boldsymbol{x}_i, \cdot) \right]_N . \tag{35}$$

By applying the inner product $\left\langle \cdot, \left[K(\boldsymbol{x}_j, \cdot) \right]_N^T \right\rangle_{\mathcal{H}}, j \in \mathbb{N}_N$ to both sides of the equation and rearranging, we can derive

$$\begin{aligned} \mathrm{d}\left(\left[f_{\theta^t}(\boldsymbol{x}_j) - f^*(\boldsymbol{x}_j) \right]_N^T \right) &= -\frac{\eta}{N} \left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N^T \cdot \left\langle \left[K(\boldsymbol{x}_i, \cdot) \right]_N, \left[K(\boldsymbol{x}_j, \cdot) \right]_N^T \right\rangle_{\mathcal{H}} \\ \therefore \mathrm{d}\left(\left[f_{\theta^t}(\boldsymbol{x}_j) - f^*(\boldsymbol{x}_j) \right]_N^T \right) &= -\frac{\eta}{N} \left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N^T \cdot \boldsymbol{K} \mathrm{d}t, \end{aligned} \tag{36}$$

where $\boldsymbol{K}$ is a symmetric and positive definite matrix of size $N \times N$ with entries $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ at the i-th row and j-th column. By substituting the index j with i, we can equivalently derive

$$\mathrm{d}\left(\left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N^T \right) = -\frac{\eta}{N} \left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N^T \cdot \boldsymbol{K} \mathrm{d}t, \tag{37}$$

which can be expanded version as

$$\begin{aligned} & \mathrm{d}\left[f_{\theta^t}(\boldsymbol{x}_1) - f^*(\boldsymbol{x}_1), \cdots, f_{\theta^t}(\boldsymbol{x}_N) - f^*(\boldsymbol{x}_N) \right] \\ = \; & -\frac{\eta}{N} \left[f_{\theta^t}(\boldsymbol{x}_1) - f^*(\boldsymbol{x}_1), \cdots, f_{\theta^t}(\boldsymbol{x}_N) - f^*(\boldsymbol{x}_N) \right] \\ & \cdot \begin{bmatrix} K(\boldsymbol{x}_1, \boldsymbol{x}_1) & K(\boldsymbol{x}_1, \boldsymbol{x}_2) & \cdots & K(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ K(\boldsymbol{x}_2, \boldsymbol{x}_1) & K(\boldsymbol{x}_2, \boldsymbol{x}_2) & \cdots & K(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ K(\boldsymbol{x}_N, \boldsymbol{x}_1) & K(\boldsymbol{x}_N, \boldsymbol{x}_2) & \cdots & K(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \mathrm{d}t. \end{aligned} \tag{38}$$

Lemma 7 provides the solution for this first-order matrix ordinary differential equation, where $\boldsymbol{\alpha}(t) = \left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N$, $\boldsymbol{\alpha}(0) = \left[f_{\theta^0}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N$ and $\boldsymbol{A} = \bar{\boldsymbol{K}} = \frac{\boldsymbol{K}}{N}$, as

$$\left[f^*(\boldsymbol{x}_i) - f_{\theta^t}(\boldsymbol{x}_i) \right]_N^T = \left[f^*(\boldsymbol{x}_i) - f_{\theta^0}(\boldsymbol{x}_i) \right]_N^T \cdot e^{-\eta \bar{\boldsymbol{K}} t}. \tag{39}$$

We can obtain an equivalent result by transposing it as

$$\left[f^*(\boldsymbol{x}_i) - f_{\theta^t}(\boldsymbol{x}_i) \right]_N = e^{-\eta \bar{\boldsymbol{K}} t} \cdot \left[f^*(\boldsymbol{x}_i) - f_{\theta^0}(\boldsymbol{x}_i) \right]_N . \tag{40}$$

After rearrangement, it is

$$\left[f_{\theta^t}(\boldsymbol{x}_i) \right]_N = \left[f^*(\boldsymbol{x}_i) \right]_N - e^{-\eta \bar{\boldsymbol{K}} t} \cdot \left[f^*(\boldsymbol{x}_i) - f_{\theta^0}(\boldsymbol{x}_i) \right]_N \tag{41}$$

For the case of $f^*(\boldsymbol{x}) - f_{\theta^t}(\boldsymbol{x}) < 0$, similarly, we have

$$\left[f_{\theta^t}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N = e^{-\eta \bar{\boldsymbol{K}} t} \cdot \left[f_{\theta^0}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N . \tag{42}$$

After rearrangement, we have

$$\left[f_{\theta^t}(\boldsymbol{x}_i) \right]_N = \left[f^*(\boldsymbol{x}_i) \right]_N + e^{-\eta \bar{\boldsymbol{K}} t} \cdot \left[f_{\theta^0}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N , \tag{43}$$

which is equivalent to the case of $f^*(\boldsymbol{x}) - f_{\theta^t}(\boldsymbol{x}) > 0$ since

$$e^{-\eta \bar{\boldsymbol{K}} t} \cdot \left[f_{\theta^0}(\boldsymbol{x}_i) - f^*(\boldsymbol{x}_i) \right]_N = -e^{-\eta \bar{\boldsymbol{K}} t} \cdot \left[f^*(\boldsymbol{x}_i) - f_{\theta^0}(\boldsymbol{x}_i) \right]_N . \tag{44}$$

This concludes the solution.

In the sense that a function can be seen as an infinite-dimensional generalization of a Euclidean vector, Equation 17 can be generalized as:

$$f_{\theta^t}(\cdot) = f^*(\cdot) + \sum_{i=1}^{\infty} e^{-\eta \lambda_i t} \nu_i(\cdot) \cdot \underbrace{\langle \nu_i, (f_{\theta^0} - f^*) \rangle_{\mathcal{H}}}_{\text{It is a constant}},$$

where ν_i denotes the corresponding eigenfunction based on spectral decomposition, *i.e.*, infinite eigenvectors.

B. Detailed Proofs

Proof of Theorem 5 By representing the evolution of an MLP through the variation of parameters and through a high-level standpoint of function variation, we have

$$-\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T\cdot\left[K(\boldsymbol{x}_i,\cdot)\right]_N=-\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T\cdot\left[\left\langle\left.\frac{\partial f_\theta}{\partial\theta}\right|_{\cdot,\theta^t},\left.\frac{\partial f_\theta}{\partial\theta}\right|_{\boldsymbol{x}_i,\theta^t}\right\rangle\right]_N+o\left(\frac{\partial\theta^t}{\partial t}\right). \tag{45}$$

Following the reorganization, we obtain

$$-\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T\cdot\left[K(\boldsymbol{x}_i,\cdot)-K_{\theta^t}(\boldsymbol{x}_i,\cdot)\right]_N=o\left(\frac{\partial\theta^t}{\partial t}\right). \tag{46}$$

By substituting the evolution of the parameters

$$\frac{\partial\theta^t}{\partial t}=-\eta\frac{\partial\mathcal{L}}{\partial\theta^t}=-\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T\cdot\left[\left.\frac{\partial f_\theta}{\partial\theta}\right|_{\boldsymbol{x}_i,\theta^t}\right]_N \tag{47}$$

into the remainder, we obtain

$$-\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T\cdot\left[K(\boldsymbol{x}_i,\cdot)-K_{\theta^t}(\boldsymbol{x}_i,\cdot)\right]_N=o\left(-\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T\cdot\left[\left.\frac{\partial f_\theta}{\partial\theta}\right|_{\boldsymbol{x}_i,\theta^t}\right]_N\right). \tag{48}$$

During the training of an MLP with a convex loss $\mathcal{L}$ (which is convex with respect to f_θ but usually nonconvex with respect to θ), we have the limit of the vector $\lim_{t\to\infty}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N=\boldsymbol{0}$. Since the right-hand side of the equation is of a higher order infinitesimal compared to the left-hand side, to maintain this equality, we can conclude that

$$\lim_{t\to\infty}\left[K(\boldsymbol{x}_i,\cdot)-K_{\theta^t}(\boldsymbol{x}_i,\cdot)\right]_N=\boldsymbol{0}. \tag{49}$$

This implies that for each $\boldsymbol{x}\in\{\boldsymbol{x}_i\}_N$, NTK converges point-wise to the canonical kernel.

■

Proof of Proposition 6 By recollecting the definition of Fréchet derivative in Definition 2, the convexity of $\mathcal{L}$ implies that

$$\frac{\partial\mathcal{L}}{\partial t}\leq\underbrace{\left\langle\frac{\partial\mathcal{L}}{\partial f_{\theta^{t+1}}},\frac{f_{\theta^t}}{\partial t}\right\rangle_{\mathcal{H}}}_{\Xi}. \tag{50}$$

By specifying the Fréchet derivative of $\frac{\partial\mathcal{L}}{\partial f_{\theta^{t+1}}}$ and the evolution of f_{θ^t}, the r.h.s. term Ξ can be expressed as

$$\begin{aligned}\Xi&=\left\langle\mathcal{G}^{t+1},-\eta\mathcal{G}^t\right\rangle_{\mathcal{H}}\\&=\frac{\eta}{N^2}\left\langle\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N^T\cdot\left[K_{\boldsymbol{x}_i}\right]_N,\left[K_{\boldsymbol{x}_i}\right]_N^T\cdot\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right\rangle_{\mathcal{H}}\\&=-\frac{\eta}{N^2}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N^T\cdot\left\langle\left[K_{\boldsymbol{x}_i}\right]_N,\left[K_{\boldsymbol{x}_i}\right]_N^T\right\rangle_{\mathcal{H}}\cdot\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\\&=-\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T\bar{\boldsymbol{K}}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N,\end{aligned} \tag{51}$$

where $\bar{\boldsymbol{K}} = \boldsymbol{K}/N$, and $\boldsymbol{K}$ is a symmetric and positive definite matrix of size $N \times N$ with elements $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ located at the i-th row and j-th column. Furthermore, the last term in Equation 51 can be rewritten as

$$\begin{aligned}
& -\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N \\
= & -\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N + \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right) \\
= & -\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N - \frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right) \\
= & -\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N \\
& +\frac{\eta}{N}\left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N^T - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N^T\right) \bar{\boldsymbol{K}} \left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right)
\end{aligned} \tag{52}$$

The last term in Equation 52 above can be elaborated as

$$\begin{aligned}
& \frac{\eta}{N}\left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N^T - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N^T\right) \bar{\boldsymbol{K}} \left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right) \\
= & \frac{\eta}{N}\left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right)^T \bar{\boldsymbol{K}} \left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right) \\
& -\frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N - \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right) \\
= & \frac{\eta}{N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i} - \left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i} - \left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N \\
& -\frac{\eta}{N}\left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N - \frac{1}{2}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right)^T \bar{\boldsymbol{K}} \left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N - \frac{1}{2}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right) \\
& +\frac{\eta}{4N}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N .
\end{aligned} \tag{53}$$

Since $\bar{\boldsymbol{K}}$ is positive definite, it is clear that $\frac{\eta}{N}\left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N - \frac{1}{2}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right)^T \bar{\boldsymbol{K}} \left(\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i}\right]_N - \frac{1}{2}\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N\right)$ is a non-negative term, and therefore by combining Equation 51, 52, and 53, we have

$$\begin{aligned}
\Xi \leq & -\frac{3\eta}{4N}\underbrace{\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N}_{\textcircled{1}} \\
& +\frac{\eta}{N}\underbrace{\left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i} - \left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left[\left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i} - \left.\frac{\partial\mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N}_{\textcircled{2}} .
\end{aligned} \tag{54}$$

Given the evaluation functional definition and the assumption that $\mathcal{L}$ is Lipschitz smooth with a constant $\xi > 0$, the term $\textcircled{2}$

in the last term of Equation 54 is upper bounded as

$$
\begin{aligned}
\textcircled{2} &= \left[\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i} - \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \bar{\boldsymbol{K}} \left[\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}},\boldsymbol{x}_i} - \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N \\
&= \left[E_{\boldsymbol{x}_i}\left(\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}}} - \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t}}\right)\right]_N^T \bar{\boldsymbol{K}} \left[E_{\boldsymbol{x}_i}\left(\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^{t+1}}} - \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t}}\right)\right]_N \\
&\leq \xi^2 \left[E_{\boldsymbol{x}_i}\left(f_{\theta^{t+1}} - f_{\theta^t}\right)\right]_N^T \bar{\boldsymbol{K}} \left[E_{\boldsymbol{x}_i}\left(f_{\theta^{t+1}} - f_{\theta^t}\right)\right]_N \\
&= \xi^2 \left\langle \left(f_{\theta^{t+1}} - f_{\theta^t}\right), [K_{\boldsymbol{x}_i}]_N^T \right\rangle_{\mathcal{H}} \cdot \bar{\boldsymbol{K}} \cdot \left\langle [K_{\boldsymbol{x}_i}]_N, \left(f_{\theta^{t+1}} - f_{\theta^t}\right)\right\rangle_{\mathcal{H}} \\
&= \eta^2\xi^2 \cdot \left[\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \frac{\left\langle [K_{\boldsymbol{x}_i}]_N, [K_{\boldsymbol{x}_i}]_N^T\right\rangle_{\mathcal{H}}}{N} \cdot \bar{\boldsymbol{K}} \cdot \frac{\left\langle [K_{\boldsymbol{x}_i}]_N, [K_{\boldsymbol{x}_i}]_N^T\right\rangle_{\mathcal{H}}}{N} \cdot \left[\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N .
\end{aligned}
\tag{55}
$$

Based on the assumption that the canonical kernel is bounded above by a constant $\zeta > 0$, we have

$$
\left\langle [K_{\boldsymbol{x}_i}]_N, [K_{\boldsymbol{x}_i}]_N^T\right\rangle_{\mathcal{H}} \leq \zeta \left\langle [1]_N, [1]_N^T \right\rangle ,
$$

and

$$
\bar{\boldsymbol{K}} \leq \frac{\zeta}{N} \left\langle [1]_N, [1]_N^T \right\rangle .
$$

Therefore, $\textcircled{1}$ is bounded above as

$$
\begin{aligned}
\textcircled{1} &\leq \frac{\zeta}{N} \left\langle \left[\sum_{i=1}^N \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T, [1]_N \right\rangle \left\langle [1]_N^T, \left[\sum_{i=1}^N \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N \right\rangle \\
&= \frac{\zeta}{N} \left(\sum_{i=1}^N \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right)^2 .
\end{aligned}
\tag{56}
$$

Simultaneously, the last term in Equation 55 is also bounded from above:

$$
\begin{aligned}
&\eta^2\xi^2 \cdot \left[\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T \frac{\left\langle [K_{\boldsymbol{x}_i}]_N, [K_{\boldsymbol{x}_i}]_N^T\right\rangle_{\mathcal{H}}}{N} \cdot \bar{\boldsymbol{K}} \cdot \frac{\left\langle [K_{\boldsymbol{x}_i}]_N, [K_{\boldsymbol{x}_i}]_N^T\right\rangle_{\mathcal{H}}}{N} \cdot \left[\left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N \\
\leq\ & \eta^2\xi^2 \left[\frac{\zeta}{N}\sum_{i=1}^N \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]^T \cdot \bar{\boldsymbol{K}} \cdot \left[\frac{\zeta}{N}\sum_{i=1}^N \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N \\
\leq\ & \frac{\eta^2\xi^2\zeta^3}{N} \left\langle \left[\frac{1}{N}\sum_{i=1}^N \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N^T, [1]_N \right\rangle \left\langle [1]_N^T, \left[\frac{1}{N}\sum_{i=1}^N \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right]_N \right\rangle \\
=\ & \frac{\eta^2\xi^2\zeta^3}{N} \left(\sum_{i=1}^N \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right)^2 .
\end{aligned}
\tag{57}
$$

Therefore, by combining Equations 54, 55, 56, and 57, we obtain

$$
\Xi \leq -\eta\zeta \left(\frac{3}{4} - \eta^2\xi^2\zeta^2\right) \left(\frac{1}{N}\sum_{i=1}^N \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right)^2 ,
\tag{58}
$$

which indicates

$$
\frac{\partial \mathcal{L}}{\partial t} \leq \Xi \leq -\eta\zeta \left(\frac{3}{4} - \eta^2\xi^2\zeta^2\right) \left(\frac{1}{N}\sum_{i=1}^N \left.\frac{\partial \mathcal{L}}{\partial f_\theta}\right|_{f_{\theta^t},\boldsymbol{x}_i}\right)^2 .
\tag{59}
$$

Hence, if $\eta \leq \frac{1}{2\xi\zeta}$, we have

$$\frac{\partial \mathcal{L}}{\partial t} \leq -\frac{\eta\zeta}{2}\left(\frac{1}{N}\sum_{i=1}^{N}\frac{\partial \mathcal{L}}{\partial f_\theta}\bigg|_{f_{\theta^t},\boldsymbol{x}_i}\right)^2. \tag{60}$$

■

Proof of Lemma 7 For $\boldsymbol{\alpha}(t) = e^{\boldsymbol{A}t}\boldsymbol{c}$, where $e^{\boldsymbol{A}t} = \sum_{i=0}^{\infty}\frac{t^i\boldsymbol{A}^i}{i!}$ and $\boldsymbol{c}$ is a time-independent column vector of size $n \times 1$, we have

$$\begin{aligned}\frac{\partial \boldsymbol{\alpha}(t)}{\partial t} &= \frac{\partial e^{\boldsymbol{A}t}\boldsymbol{c}}{\partial t} = \frac{\partial \sum_{i=0}^{\infty}\frac{t^i\boldsymbol{A}^i}{i!}\boldsymbol{c}}{\partial t} = \sum_{i=1}^{\infty}\frac{\partial t^i}{\partial t}\frac{\boldsymbol{A}^i\boldsymbol{c}}{i!}\\ &= \boldsymbol{A}\sum_{i=1}^{\infty}\frac{\boldsymbol{A}^{i-1}t^{i-1}\boldsymbol{c}}{(i-1)!} = \boldsymbol{A}\sum_{i=0}^{\infty}\frac{\boldsymbol{A}^i t^i\boldsymbol{c}}{i!} = \boldsymbol{A}e^{\boldsymbol{A}t}\boldsymbol{c} = \boldsymbol{A}\boldsymbol{\alpha}(t).\end{aligned} \tag{61}$$

Meanwhile, by setting $t = 0$, we have

$$\boldsymbol{\alpha}(0) = e^0\boldsymbol{c}, \tag{62}$$

which means $\boldsymbol{c} = \boldsymbol{\alpha}(0)$. Therefore, $\boldsymbol{\alpha}(t) = e^{\boldsymbol{A}t}\boldsymbol{\alpha}(0)$ is the unique solution of the matrix ODE $\frac{\partial\boldsymbol{\alpha}(t)}{\partial t} = \boldsymbol{A}\boldsymbol{\alpha}(t)$ with initial value $\boldsymbol{\alpha}(0)$.

■

C. Experiment Details

C.1. Synthetic 1D signal

The FFN consists of 4 layers, each with 256 hidden units, and the value of σ is set to 2 for the random Fourier features used in the FFN. Based on Theorem 5, the canonical kernel used in FGD is approximated by adopting the empirical NTK of the INR obtained through PGD after 5000 iterations.

C.2. Toy 2D Cameraman Fitting

We train SIREN models with 6 layers, each with 256 hidden units, with default settings as mentioned in Sitzmann et al., 2020b to fit the 512×512 Cameraman grayscale image from `scikit-image` (Van der Walt et al., 2014). To have a close resemblance with the theoretical analysis of INT, we train the models with vanilla gradient descent without momentum for 5000 iterations. All models are trained using a cosine annealing scheduler with a starting learning rate of 1e-4 and a minimum learning rate of 1e-6. The specific INT sampling strategies of the 4 different SIREN models presented in Figure 4 are as follows:

- w/o INT - At each optimization step, the entire image is used.
- w/o INT (20%) - At each optimization step, a random 20% of pixels are used.
- With INT (20%) - At each optimization step, pixels with the top 20% error rates from the previous training iteration are used to train the current iteration.
- With INT (incre.) - Similar scheme as “with INT (20%)”, except that we increase the sampling rate by 8% for every 500 iterations from 20% to 92%.

C.3. INT Strategy Experiment

We train identical SIREN models as mentioned in the previous section on 8/24 images from the Kodak dataset (Eastman Kodak Company, 1999). As numerous strategies were tested and we hoped to utilize a wide variety of images to find a robust strategy that works not only across different image datasets but also other modalities, we chose only a representative subset of the Kodak dataset for experimental efficiency. As shown in Figure 9, this subset of images is chosen to include both simple images (e.g. single object), complex images (e.g. multiple objects or high-frequency signals such as grass), and images with humans. To better simulate real-world scenarios of utilizing INRs, We test our strategies with the Adam (Kingma & Ba, 2015) optimizer with a learning rate of 1e-3 and an identical cosine annealing scheduler for the learning rate as in the previous section. All models are trained for 5000 iterations.

We highlight that logging PSNR/SSIM values and saving visualization results during training takes up significant time. Thus, to record the most realistic training time, we retrain all the models with the same seed and configurations but without any logs except for the loss value on a single image. As all images have the same dimensions, this is sufficient to represent the general trend of training times across the strategies.

The specific INT strategies presented in Figure 6 are as follows:

- Ratio
  - Cosine - Increasing sampling ratio from 20% to 100% in a cosine annealing manner.
  - R-Cosine - Decreasing sampling ratio from 100% to 20% in a cosine annealing manner.
  - Step - Incrementing sampling ratio from 20% to 92% in 10 equal intervals, which is 500 iterations in this case where we train for a total of 5000 iterations.
- Interval
  - Dense - Sample points with top <ratio>% error rates for *every training iteration*. Note that the error rates are obtained from the previous iteration.
  - Decremental - Sampling interval decreases from every 90 iterations to 1 iteration incrementally in 10 intervals, which is 500 iterations in this case where we train for a total of 5000 iterations. That is, at every 500 iterations, we decrease the interval by 10, except for the last 500 iterations where we decrease by 9 from 10 to 1.

- Incremental - Sampling interval increases from every 1 iteration to 90 iterations incrementally in 10 intervals, which is 500 iterations in this case where we train for a total of 5000 iterations. That is, at every 500 iterations, we increase the interval by 10, except for the first 500 iterations where we increase by 9 from 1 to 10.

Figure 9. The selected 8/24 images from the Kodak dataset.

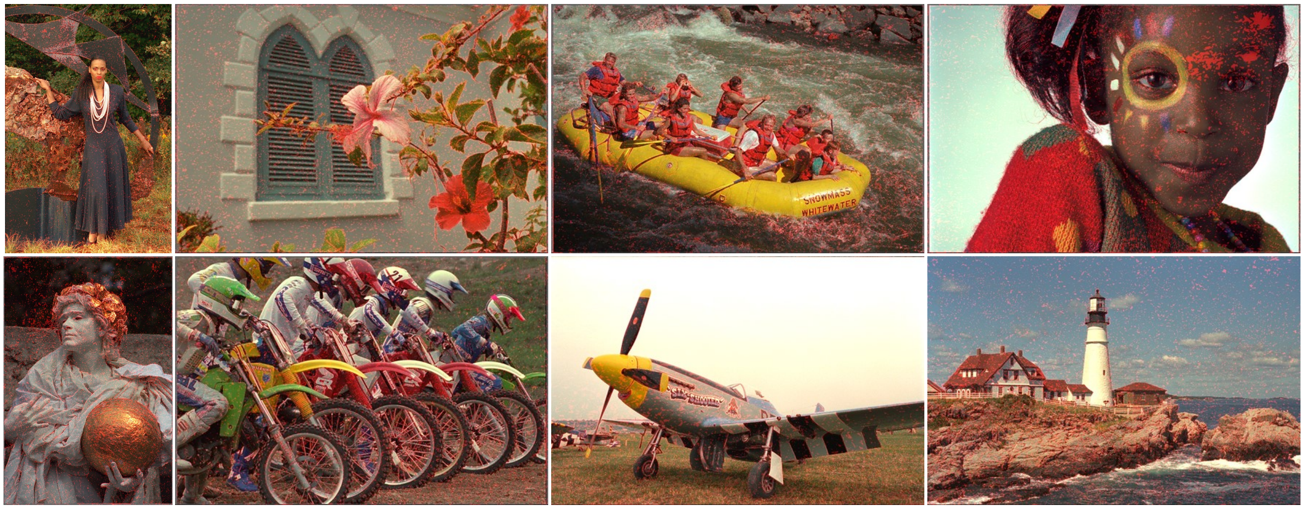


Figure 10. The selected 8/24 images from the Kodak dataset and its selected training points at a particular instance.

Figure 11 presents the sampling progression of SIREN trained with SGD and Adam on the kodak05 image. Besides, examining the applicability of active data selection methods (Loshchilov & Hutter, 2015; Graves et al., 2017; Mindermann et al., 2022) for INRs learning efficiency could be interesting.

C.4. Multi-modality Signal Fitting

For all modalities, we train a SIREN model with Adam optimizer and cosine annealing learning rate scheduler. We set $\omega_0 = 30$ for the SIREN model. All modalities except 2D Kodak images start with a learning rate of 1e-4, while 2D Kodak images start with 1e-3. We select the best INT strategy found in the previous section to train for all modalities: "step-incremental". Note that we always partition the training into 10 same-sized intervals where each interval has its respective INT sampling ratio and sampling interval. For instance, if we train audio samples for 10K iterations, then we start with 20% sampling ratio and a sampling ratio of 1 and progressively add 8% to the sampling ratio and 10 to the sampling interval for every 1K iterations.

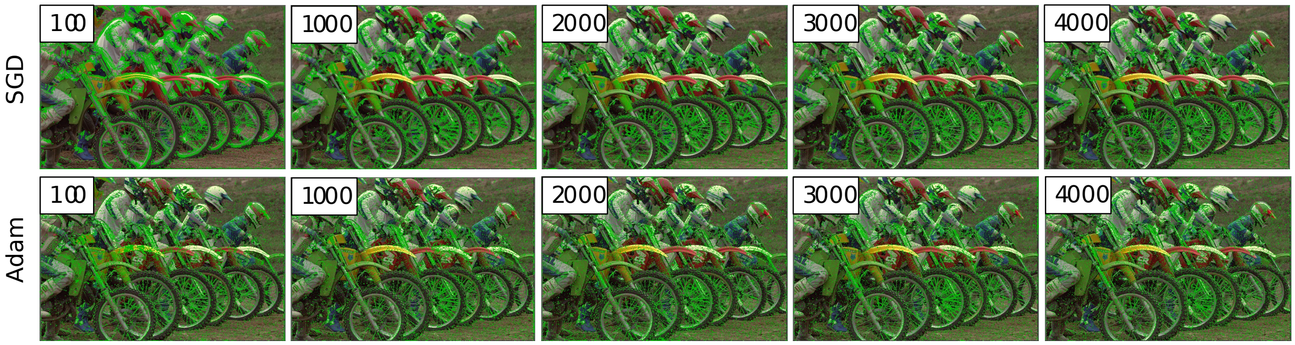


Figure 11. Visualizing the progression of sampled points when trained with SGD vs Adam on kodak05 image.

1D Audio. The Librispeech dataset (Panayotov et al., 2015) is chosen for the audio-fitting task. We select the first 100 samples from the test-clean split that have a duration greater than 2 seconds. For our evaluation benchmark, we clip the first 2 seconds of each sample. We train a SIREN with 5 layers, each having 128 hidden units, resulting in a total of approximately 50K parameters. Each sample is trained for 10K iterations.

2D image. The entire Kodak dataset (Eastman Kodak Company, 1999) is used in this case. Model configuration and training parameters are identical to the previous section, except that we select the "Step-Incremental" strategy for the INT training. The resulting SIREN model has approximately 265K parameters.

Megapixel Image We fit the 8192×8192 Pluto image (NASA, 2018). We use a SIREN model with 6 layers, each having 512 hidden units, resulting in a total of approximately 1M parameters. This model size is necessary to fit the image with 30+ PSNR. Model configurations are identical to that of 2D image fitting. We also train with Adam optimizer and cosine annealing learning rate scheduler, but instead, start with a learning rate of 1e-4. During training, we break the image into mini-batches of 524,288 pixels and have the INT algorithm sample training pixels for each optimization step. This is necessary due to VRAM constraints of consumer-grade GPUs such as NVIDIA RTX3090 (24GB). We train for a total of 500 epochs, where each epoch consists of 128 training steps that progressively sample the entire image. Thus, we also tune the incremental INT sampling interval to increase from 1 to 10 instead of from 1 to 100.

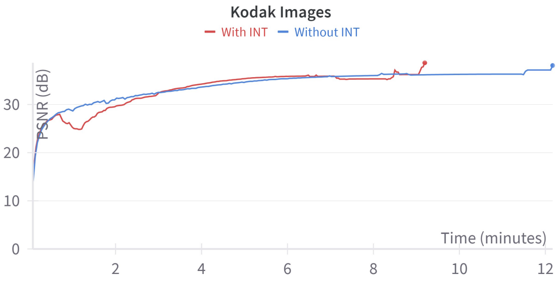


Figure 12. PSNR-Training time curve of Kodak images training with and without INT.

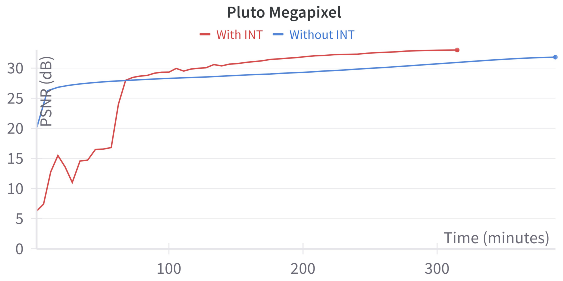


Figure 13. PSNR-Training time curve of megapixel training with and without INT.

The "non-smoothness" of training curve in Figure 12 and 13 is due to the increase in sampling intervals. In particular, the drop in reconstruction quality occurs when changing from densely selecting optimal training points at each iteration to sampling once per several iterations (as a measure of saving training time without sacrificing much "final" reconstruction quality). One can think of sampling at sparser intervals as analogous to training on dynamic minibatches of the data. Hence, at early stages of training when the model has not properly learnt the underlying signal yet, these minibatch training steps may lead to temporary overfitting and more "jumpy" training curves. However, our results show that this does not affect the "final" reconstruction quality. In fact, accompanying increasing INT ratio with sampling intervals is the optimal method of balancing lesser training samplers (faster training time) and retaining training quality.

3D Shape. We conduct 3D shape experiments using the Stanford 3D Scanning Repository dataset (Stanford Computer Graphics Laboratory, 2007). We choose 4 scenes: Asian Dragon, Thai Statue, Lucy, and Armadillo. For our experiments, we utilize an 8-layer SIREN with 256 hidden units, resulting in approximately 400K parameters. Each scene is trained for 10K iterations. Following the approach of Bacon (Lindell et al., 2022) and Scone (Li et al., 2024a), we sample points from the surface using a coarse and fine sampling procedure. We add two levels of Laplacian noise with variances of 1e-1 and 1e-3 for the coarse and fine samples, respectively. During each iteration, we randomly select a batch of 50K points. If INT is utilized, it is applied within each batch. IoU is computed by first transforming the learned signed distance function (SDF) to an occupancy grid of shape $512 \times 512 \times 512$ bounded by $[-0.5, 0.5]^3$. Below, we present the complete results for each scene:

Scene	INT	IoU(%)
Asian Dragon	✗	96.46
	✓	96.05
Armadillo	✗	98.48
	✓	98.25
Thai Statue	✗	96.43
	✓	96.22
Lucy	✗	96.91
	✓	96.19

Table 3. 3D shape representation results for all scenes.