Understanding Dynamics of Nonlinear Representation Learning and Its Application

Kenji Kawaguchi
kkawaguchi@fas.harvard.edu
Harvard University, Cambridge, MA 02138, U.S.A.

Linjun Zhang
linjun.zhang@rutgers.edu
Rutgers University, New Brunswick, NJ 08901

Zhun Deng
zhundeng@g.harvard.edu
Harvard University Cambridge, MA 02138, U.S.A.

Representations of the world environment play a crucial role in artificial intelligence. It is often inefficient to conduct reasoning and inference directly in the space of raw sensory representations, such as pixel values of images. Representation learning allows us to automatically discover suitable representations from raw sensory data. For example, given raw sensory data, a deep neural network learns nonlinear representations at its hidden layers, which are subsequently used for classification (or regression) at its output layer. This happens implicitly during training through minimizing a supervised or unsupervised loss. In this letter, we study the dynamics of such implicit nonlinear representation learning. We identify a pair of a new assumption and a novel condition, called the *on-model structure assumption* and the *data architecture alignment condition*. Under the on-model structure assumption, the data architecture alignment condition is shown to be sufficient for the global convergence and necessary for global optimality. Moreover, our theory explains how and when increasing network size does and does not improve the training behaviors in the practical regime. Our results provide practical guidance for designing a model structure; for example, the on-model structure assumption can be used as a justification for using a particular model structure instead of others. As an application, we then derive a new training framework, which satisfies the data architecture alignment condition without assuming it by automatically modifying any given training algorithm dependent on data and architecture. Given a standard training algorithm, the framework running its modified version is empirically shown to maintain competitive (practical) test performances while providing global convergence guarantees for deep residual neural

networks with convolutions, skip connections, and batch normalization with standard benchmark data sets, including MNIST, CIFAR-10, CIFAR-100, Semeion, KMNIST, and SVHN.

1 Introduction

LeCun, Bengio, and Hinton (2015) described deep learning as one of hierarchical nonlinear representation learning approaches:

> Deep-learning methods are representation-learning methods with multiple levels of representation, obtained by composing simple but non-linear modules that each transform the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level (p. 436).

In applications such as computer vision and natural language processing, the success of deep learning can be attributed to its ability to learn hierarchical nonlinear representations by automatically changing nonlinear features and kernels during training based on the given data. This is in contrast to classical machine-learning methods where representations or equivalently nonlinear features and kernels are fixed during training.

Deep learning in practical regimes, which has the ability to learn nonlinear representation (Bengio, Courville, & Vincent, 2013), has had a profound impact in many areas, including object recognition in computer vision (Rifai et al., 2011; Hinton, Osindero, & Teh, 2006; Bengio, Lamblin, Popovici, & Larochelle, 2007; Ciregan, Meier, & Schmidhuber, 2012; Krizhevsky, Sutskever, & Hinton, 2012), style transfer (Gatys, Ecker, & Bethge, 2016; Luan, Paris, Shechtman, & Bala, 2017), image super-resolution (Dong, Loy, He, & Tang, 2014), speech recognition (Dahl, Ranzato, Mohamed, & Hinton, 2010; Deng et al., 2010; Seide, Li, & Yu, 2011; Mohamed, Dahl, & Hinton, 2011; Dahl, Yu, Deng, & Acero, 2011; Hinton et al., 2012), machine translation (Schwenk, Rousseau, & Attik, 2012; Le, Oparin, Allauzen, Gauvain, & Yvon, 2012), paraphrase detection (Socher, Huang, Pennington, Ng, & Manning, 2011), word sense disambiguation (Bordes, Glorot, Weston, & Bengio, 2012), and sentiment analysis (Glorot, Bordes, & Bengio, 2011; Socher, Pennington, Huang, Ng, & Manning, 2011). However, we do not yet know the precise condition that makes deep learning tractable in the practical regime of representation learning.

To initiate a study toward such a condition, we consider the following problem setup that covers deep learning in the practical regime and other nonlinear representation learning methods in general. We are given a training data set $((x_i, y_i))_{i=1}^n$ of n samples where $x_i \in \mathcal{X} \subseteq \mathbb{R}^{m_x}$ and $y_i \in \mathcal{Y} \subseteq \mathbb{R}^{m_y}$ are the ith input and the ith target, respectively. We would like to learn a predictor (or hypothesis) from a parametric family $\mathcal{H} = \{f(\cdot, \theta) : \mathbb{R}^{m_x} \to \mathbb{R}^{m_y} \mid \theta \in \mathbb{R}^d\}$ by minimizing the objective $\mathcal{L}$,

$$\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i, \theta), y_i), \tag{1.1}$$

where $\ell : \mathbb{R}^{m_y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ is the loss function that measures the discrepancy between the prediction $f(x_i, \theta)$ and the target y_i for each sample. In this letter, it is allowed to let $y_i = x_i$ (for $i = 1, \ldots, n$) to include the setting of unsupervised learning. The function f is also allowed to represent a wide range of machine learning models. For a (deep) neural network, $f(x_i, \theta)$ represents the preactivation output of the last layer of the network, and the parameter vector $\theta \in \mathbb{R}^d$ contains all the trainable parameters, including weights and bias terms of all layers.

For example, one of the simplest models is the linear model in the form of $f(x, \theta) = \phi(x)^\top \theta$, where $\phi : \mathcal{X} \to \mathbb{R}^d$ is a fixed function and $\phi(x)$ is a nonlinear representation of input data x. This is a classical machine learning model where much of the effort goes into the design of the handcrafted feature map ϕ via feature engineering (Turner, Fuggetta, Lavazza, & Wolf, 1999; Zheng & Casari, 2018). In this linear model, we do not learn the representation $\phi(x)$ because the feature map ϕ is fixed without dependence on the model parameter θ that is optimized with the data set $((x_i, y_i))_{i=1}^n$.

Similar to many definitions in mathematics, where an intuitive notion in a special case is formalized to a definition for a more general case, we now abstract and generalize this intuitive notion of the representation $\phi(x)$ of the linear model to that of all differentiable models as follows:

Definition 1. *Given any $x \in \mathcal{X}$ and differentiable function f, we define $\frac{\partial f(x,\theta)}{\partial \theta}$ to be the gradient representation of the data x under the model f at θ.*

This definition recovers the standard representation $\phi(x)$ in the linear model as $\frac{\partial f(x,\theta)}{\partial \theta} = \frac{\partial \phi(x)^\top \theta}{\partial \theta} = \phi(x)$ and is applicable to all differentiable nonlinear models in representation learning. Moreover, this definition captures the key challenge of understanding the dynamics of nonlinear representation learning well, as illustrated below. Using the notation of $\frac{d\theta^t}{dt} = \Delta^t$, the dynamics of the model $f(x, \theta^t)$ over the time t can be written by

$$\frac{d}{dt} f(x, \theta^t) = \frac{\partial f(x, \theta^t)}{\partial \theta^t} \frac{d\theta^t}{dt} = \frac{\partial f(x, \theta^t)}{\partial \theta^t} \Delta^t. \tag{1.2}$$

Here, we can see that the dynamics are *linear* in Δ^t if there is no gradient representation learning as $\frac{\partial f(x,\theta^t)}{\partial \theta^t} \approx \frac{\partial f(x,\theta^0)}{\partial \theta^0}$. However, with representation learning, the gradient representation $\frac{\partial f(x,\theta^t)}{\partial \theta^t}$ changes depending on t (and Δ^t), resulting in the dynamics that are *nonlinear* in Δ^t. Therefore, the definition of the gradient representation can distinguish fundamentally different dynamics in machine learning.

In this letter, we initiate the study of the dynamics of learning gradient representation that are nonlinear in Δ^t. That is, we focus on the regime where the gradient representation $\frac{\partial f(x,\theta^t)}{\partial \theta^t}$ at the end of training time t differs greatly from the initial representation $\frac{\partial f(x,\theta^0)}{\partial \theta^0}$. This regime was studied in the past for the case where the function $\phi(x) \mapsto f(x, \theta)$ is affine for some fixed feature map ϕ (Saxe, McClelland, & Ganguli, 2014; Kawaguchi, 2016, 2021; Laurent & Brecht, 2018; Bartlett, Helmbold, & Long, 2019; Zou, Long, & Gu, 2020; Xu et al., 2021). Unlike any previous studies, we focus on the problem setting where the function $\phi(x) \mapsto f(x, \theta)$ is nonlinear and nonaffine, with the effect of nonlinear (gradient) representation learning. The results of this letter avoid the curse of dimensionality by studying the global convergence of the gradient-based dynamics instead of the dynamics of global optimization (Kawaguchi et al., 2016) and Bayesian optimization (Kawaguchi, Kaelbling, & Lozano-Pérez, 2015). Importantly, we do not require any wide layer or large input dimension throughout this letter. Our main contributions are summarized as follows:

1. In section 2, we identify a pair of a novel assumption and a new condition, called the *common model structure assumption* and the *data-architecture alignment condition*. Under the common model structure assumption, the data-architecture alignment condition is shown to be a necessary condition for the globally optimal model and a sufficient condition for the global convergence. The condition is dependent on both data and architecture. Moreover, we empirically verify and deepen this new understanding. When we apply representation learning in practice, we often have overwhelming options regarding which model structure to be used. Our results provide a practical guidance for choosing or designing model structure via the common model structure assumption, which is indeed satisfied by many representation learning models used in practice.
2. In section 3, we discard the assumption of the data-architecture alignment condition. Instead, we derive a novel training framework, called the exploration-exploitation wrapper (EE wrapper), which satisfies the data-architecture alignment condition time-independently a priori. The EE wrapper is then proved to have global convergence guarantees under the *safe-exploration condition*. The safe-exploration condition is what allows us to explore various gradient representations safely without getting stuck in the states where we cannot provably satisfy the data-architecture alignment condition. The safe-exploration condition is shown to hold true for ResNet-18 with standard benchmark data sets, including MNIST, CIFAR-10, CIFAR-100, Semeion, KMNIST, and SVHN time-independently.
3. In section 3.4, the EE wrapper is shown to not degrade practical performances of ResNet-18 for the standard data sets, MNIST, CIFAR-10, CIFAR-100, Semeion, KMNIST, and SVHN. To our knowledge, this

letter provides the first practical algorithm with global convergence guarantees without degrading practical performances of ResNet-18 on these standard data sets, using convolutions, skip connections, and batch normalization without any extremely wide layer of the width larger than the number of data points. To the best of our knowledge, we are not aware of any similar algorithms with global convergence guarantees in the regime of learning nonlinear representations without degrading practical performances.

It is empirically known that increasing network size tends to improve training behaviors. Indeed, the size of networks correlates well with the training error in many cases in our experiments (e.g., see Figure 1b). However, the size and the training error do not correlate well in some experiments (e.g., see Figure 1c). Our new theoretical results explain that the training behaviors correlate more directly with the data-architecture alignment condition instead. The seeming correlation with the network size is caused by another correlation between the network size and the data-architecture alignment condition. This is explained in more detail in section 2.3.

2 Understanding Dynamics via Common Model Structure and Data-Architecture Alignment

In this section, we identify the common model structure assumption and study the data-architecture alignment condition for the global convergence in nonlinear representation learning. We begin by presenting an overview of our results in section 2.1, deepen our understandings with experiments in section 2.2, discuss implications of our results in section 2.3, and establish mathematical theories in section 2.4.

2.1 Overview. We introduce the common model structure assumption in section 2.1.1 and define the data-architecture alignment condition in section 2.1.2. Using the assumption and the condition, we present the global convergence result in section 2.1.3.

2.1.1 Common Model Structure Assumption. Through examinations of representation learning models used in applications, we identified and formalized one of their common properties as follows:

Assumption 1 (Common Model Structure Assumption). There exists a subset $S \subseteq \{1, 2, \ldots, d\}$ such that $f(x_i, \theta) = \sum_{k=1}^{d} \mathbb{1}\{k \in S\}\theta_k\left(\frac{\partial f(x_i,\theta)}{\partial \theta_k}\right)$ for any $i \in \{1, \ldots, n\}$ and $\theta \in \mathbb{R}^d$.

Assumption 1 is satisfied by common machine learning models, such as kernel models and multilayer neural networks, with or without convolutions, batch normalization, pooling, and skip connections. For example, consider a multilayer neural network of the form $f(x, \theta) = Wh(x, u) + b$, where $h(x, u)$ is an output of its last hidden layer and the parameter vector θ consists of the parameters (W, b) at the last layer and the parameters u in all other layers as $\theta = \text{vec}([W, b, u])$. Here, for any matrix $M \in \mathbb{R}^{m\times\bar{m}}$, we let $\text{vec}(M) \in \mathbb{R}^{m\bar{m}}$ be the standard vectorization of the matrix M by stacking columns. Then, assumption 1 holds because $f(x, \theta) = \sum_{k=1}^{d} \mathbb{1}\{k \in S\}\theta_k(\frac{\partial f(x_i,\theta)}{\partial \theta_k})$, where S is defined by $\{\theta_k : k \in S\} = \{\text{vec}([W, b])_k : k = 1, 2, \ldots, \xi\}$ with $\text{vec}([W, b]) \in \mathbb{R}^{\xi}$. Since h is arbitrary in this example, the common model structure assumption holds, for example, for any multilayer neural networks with a fully connected last layer. In general, because the nonlinearity at the output layer can be treated as a part of the loss function ℓ while preserving convexity of $q \mapsto \ell(q, y)$ (e.g., cross-entropy loss with softmax), this assumption is satisfied by many machine learning models, including ResNet-18 and all models used in the experiments in this letter (as well as all linear models). Moreover, assumption 1 is automatically satisfied in the next section by using the EE wrapper.

2.1.2 Data-Architecture Alignment Condition. Given a target matrix $Y = (y_1, y_2, \ldots, y_n)^\top \in \mathbb{R}^{n\times m_y}$ and a loss function ℓ, we define the modified target matrix $Y_\ell = (y_1^\ell, y_2^\ell, \ldots, y_n^\ell)^\top \in \mathbb{R}^{n\times m_y}$ by $Y_\ell = Y$ for the squared loss ℓ, and by $(Y_\ell)_{ij} = 2Y_{ij} - 1$ for the (binary and multiclass) cross-entropy losses ℓ with $Y_{ij} \in \{0, 1\}$. Given input matrix $X = (x_1, x_2, \ldots, x_n)^\top \in \mathbb{R}^{n\times m_x}$, the output matrix $f_X(\theta) \in \mathbb{R}^{n\times m_y}$ is defined by $f_X(\theta)_{ij} = f(x_i, \theta)_j \in \mathbb{R}$. For any matrix $M \in \mathbb{R}^{m\times\bar{m}}$, we let $\text{Col}(M) \subseteq \mathbb{R}^m$ be its column space. With these notations, we are now ready to introduce the data-architecture alignment condition:

Definition 2 (Data-Architecture Alignment Condition). *Given any data set (X, Y), differentiable function f, and loss function ℓ, the data-architecture alignment condition is said to be satisfied at θ if* $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta))}{\partial \theta})$.

The data-architecture alignment condition depends on both data (through the target Y and the input X) and architecture (through the model f). It is satisfied only when the data and architecture align well to each other. For example, in the case of linear model $f(x, \theta) = \phi(x)^\top\theta \in \mathbb{R}$, the condition can be written as $\text{vec}(Y_\ell) \in \text{Col}(\Phi(X))$ where $\Phi(X) \in \mathbb{R}^{n\times d}$ and $\Phi(X)_{ij} = \phi(x_i)_j$. In definition 2, $f_X(\theta)$ is a matrix of the preactivation outputs of the last layer. Thus, in the case of classification tasks with a nonlinear activation at the output layer, $f_X(\theta)$ and Y are not in the same space, which is the reason we use Y_ℓ here instead of Y.

Importantly, the data-architecture alignment condition does not make any requirements on the the rank of the Jacobian matrix $\frac{\partial \text{vec}(f_X(\theta))}{\partial \theta} \in \mathbb{R}^{nm_y\times d}$: the rank of $\frac{\partial \text{vec}(f_X(\theta))}{\partial \theta}$ is allowed to be smaller than nm_y and d. Thus, for example, the data-architecture alignment condition can be satisfied depending on the given data and architecture even if the minimum eigenvalue of the matrix $\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta})^\top$ is zero, in both cases of overparameterization (e.g., $d \gg n$) and underparameterization (e.g., $d \ll n$). This is further illustrated in section 2.2 and discussed in section 2.3. We note that we further discard the assumption of the data-architecture alignment condition in section 3 as it is automatically satisfied by using the EE wrapper.

2.1.3 Global Convergence. Under the common model structure assumption, the data-architecture alignment condition is shown to be what lets us avoid the failure of the global convergence and suboptimal local minima. More concretely, we prove a global convergence guarantee under the data-architecture alignment condition as well as the necessity of the condition for the global optimality:

Theorem 1 (Informal Version). *Let assumption 1 hold. Then the following two statements hold for gradient-based dynamics:*

(i) The global optimality gap bound decreases per iteration toward zero at the rate of $O(1/\sqrt{|\mathcal{T}|})$ for any $\mathcal{T}$ such that the data-architecture alignment condition is satisfied at θ^t for $t \in \mathcal{T}$.
(ii) For any $\theta \in \mathbb{R}^d$, the data-architecture alignment condition at θ is necessary to have the globally optimal model $f_X(\theta) = \eta Y_\ell$ at θ for any $\eta \in \mathbb{R}$.

Theorem 1i guarantees the global convergence without the need to satisfy the data-architecture alignment condition at every iteration or at the limit point. Instead, it shows that the bound on the global optimality gap decreases toward zero per iteration whenever the data-architecture alignment condition holds. Theorem 1ii shows that the data-architecture alignment condition is necessary for the global optimality. Intuitively, this is because the expressivity of a model class satisfying the common model structure assumption is restricted such that it is required to align the architecture to the data in order to contain the globally optimal model $f_X(\theta) = \eta Y_\ell$ (for any $\eta \in \mathbb{R}$).

To better understand the statement of theorem 1i, consider a counterexample with a data set consisting of the single point $(x, y) = (1, 0)$, the model $f(x, \theta) = \theta^4 - 10\theta^2 + 6\theta + 100$, and the squared loss $\ell(q, y) = (q - y)^2$. In this example, we have $\mathcal{L}(\theta) = f(x, \theta)^2$, which has multiple suboptimal local minima of different values. Then, via gradient descent, the model converges to the closest local minimum and, in particular, does not necessarily converge to a global minimum. Indeed, this example violates the common model structure assumption (assumption 1) (although it satisfies the data-architecture alignment condition), showing the importance of the common model structure assumption along with the data-architecture alignment. This also illustrates the nontriviality of theorem 1i in that the data-architecture alignment is not sufficient, and we needed to understand what types of model structures are commonly used in practice and formalize the understanding as the common model structure assumption.

To further understand the importance of the common model structure assumption in theorem 1, we now consider the case where we do not require the assumption. Indeed, we can guarantee the global convergence without the common model structure assumption if we ensure that the minimum eigenvalue of the matrix $\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta})^\top$ is nonzero. This can be proved by the following derivation. Let θ be an arbitrary stationary point of $\mathcal{L}$. Then we have $0 = \frac{\partial \mathcal{L}(\theta)}{\partial\theta} = \frac{1}{n}\sum_{i=1}^n (\frac{\partial \ell(q,y_i)}{\partial q}|_{q=f(x_i,\theta)})\frac{\partial f(x_i,\theta)}{\partial\theta}$, which implies that

$$\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta} v = 0, \tag{2.1}$$

where $v = \mathrm{vec}((\frac{\partial \ell(q,u_1)}{\partial q}|_{q=f(x_1,\theta)})^\top, \ldots, (\frac{\partial \ell(q,u_N)}{\partial q}|_{q=f(x_n,\theta)})^\top) \in \mathbb{R}^{nm_y}$. Therefore, if the minimum eigenvalue of the matrix $\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta})^\top$ is nonzero, then we have $v = 0$: $\frac{\partial \ell(q,u_i)}{\partial q}|_{q=f(x_i,\theta)} = 0$ for all $i \in \{1, 2, \ldots, n\}$. Using the convexity of the map $q \mapsto \ell(q, y)$ (which is satisfied by the squared loss and cross-entropy loss), this implies that for any $q_1, q_2, \ldots, q_n \in \mathbb{R}^{m_y}$,

$$\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(f(x_i,\theta), y_i) \tag{2.2}$$

$$\leq \frac{1}{n}\sum_{i=1}^n \left(\ell(q_i, y_i) - \left(\frac{\partial \ell(q,u_i)}{\partial q}\bigg|_{q=f(x_i,\theta)}\right)(q_i - f(x_i,\theta))\right) \tag{2.3}$$

$$\leq \frac{1}{n}\sum_{i=1}^n \ell(q_i, y_i). \tag{2.4}$$

Since $f(x_1), f(x_2), \ldots, f(x_n) \in \mathbb{R}^{m_y}$, this implies that any stationary point θ is a global minimum if the minimum eigenvalue of the matrix $\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta})^\top$ is nonzero, without the common model structure assumption (see assumption 1). Indeed, in the above example with the model $f(x,\theta) = \theta^4 - 10\theta^2 + 6\theta + 100$, the common model structure assumption is violated, but we still have the global convergence if the minimum eigenvalue is nonzero—for example, $f(x,\theta) = y = 0$ at any stationary point θ such that the minimum eigenvalue of the matrix $\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta})^\top$ is nonzero. In contrast, theorem 1 allows the global convergence even when the minimum eigenvalue of the matrix $\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta})^\top$ is zero by utilizing the common model structure assumption.

The formal version of theorem 1 is presented in section 2.4 and is proved in appendix A in the supplementary information that relies on the additional previous works of Clanuwat et al. (2019), Krizhevsky and Hinton (2009), Mityagin (2015), Netzer et al. (2011), Paszke et al. (2019a, 2019b), and Poggio et al. (2017). Before proving the statement, we first examine the meaning and implications of our results through illustrative examples in sections 2.2 and 2.3.

2.2 Illustrative Examples in Experiments. Theorem 1 suggests that data-architecture alignment condition $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$ has the ability to distinguish the success and failure cases, even when the minimum eigenvalue of the matrix $\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})^\top$ is zero for all $t \geq 0$. In this section, we conduct experiments to further verify and deepen this theoretical understanding.

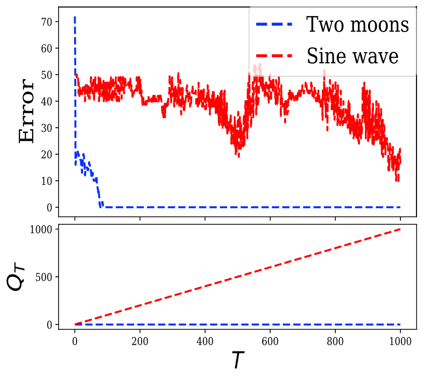
We employ a fully connected network having four layers with 300 neurons per hidden layer, and a convolutional network, LeNet (LeCun et al., 1998), with five layers. For the fully connected network, we use the two-moons data set (Pedregosa et al., 2011) and a sine wave data set. To create the sine wave data set, we randomly generated the input x_i from the uniform distribution on the interval $[-1, 1]$ and set $y_i = \mathbb{1}\{\sin(20x_i) < 0\} \in \mathbb{R}$ for all $i \in [n]$ with $n = 100$. For the convolutional network, we use the Semeion data set (Srl & Brescia, 1994) and a random data set. The random data set was created by randomly generating each pixel of the input image $x_i \in \mathbb{R}^{16\times16\times1}$ from the standard normal distribution and by sampling y_i uniformly from $\{0, 1\}$ for all $i \in [n]$ with $n = 1000$. We set the activation functions of all layers to be softplus $\bar{\sigma}(z) = \ln(1 + \exp(\varsigma z))/\varsigma$ with $\varsigma = 100$, which approximately behaves as the ReLU activation as shown in appendix C in the supplementary information. See appendix B in the supplementary information for more details of the experimental settings.

The results of the experiments are presented in Figure 1. In each panel of the figure, the training errors and the values of Q_T are plotted over time T. Here, Q_T counts the number of Y_ℓ *not* satisfying the condition $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$ during $t \in \{0, 1, \ldots, T\}$ and is defined by

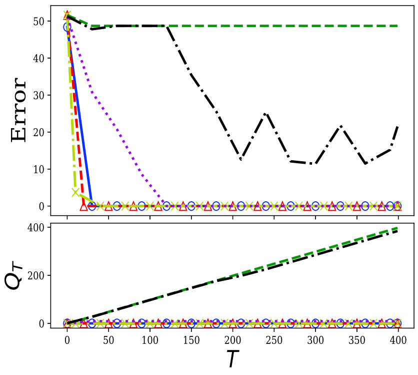
$$Q_T = \sum_{t=0}^{T} \mathbb{1}\left\{\text{vec}(Y_\ell) \notin \text{Col}\left(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t}\right)\right\}. \tag{2.5}$$

Figure 1a shows the results for the fully connected network. For the two-moons data set, the network achieved the zero training error with $Q_T = 0$ for all T (i.e., $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^T))}{\partial \theta^T})$ for all T). For the sine wave data set, it obtained high training errors with $Q_T = T$ for all T (i.e., $\text{vec}(Y_\ell) \notin \text{Col}(\frac{\partial \text{vec}(f_X(\theta^T))}{\partial \theta^T})$ for all T). This is consistent with our theory. Our theory explains that what makes a data set easy to be fitted or not is whether the condition $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$ is satisfied or not.

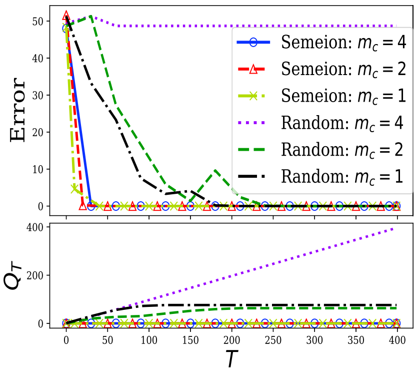
Figures 1b and 1c show the results for the convolutional networks with two random initial points using two different random seeds. In the figure panels, we report the training behaviors with different network sizes $m_c = 1, 2,$ and 4; the number of convolutional filters per convolutional layer is $8 \times m_c$ and the number of neurons per fully connected hidden layer is $128 \times m_c$. As can be seen, with the Semeion data set, the networks of all sizes achieved zero error with $Q_T = 0$ for all T. With the random data set,



(a) Fully-connected network



(b) Convolutional network: seed #1



(c) Convolutional network: seed #2

Figure 1: Training error and the value of Q_T over time steps T. The legend of panel b is shown in panel c. The value of Q_T measures the number of Y_ℓ not satisfying the condition of $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$. This figure validates our theoretical understanding that the bound on the global optimality gap decreases at any iteration when Q_T does not increase at the iteration—that is, when the Q_T value is flat. The minimum eigenvalue of the matrix $\mathbf{M}(\theta^t) = \frac{\partial \text{vec}(f_X(\theta))}{\partial \theta}(\frac{\partial \text{vec}(f_X(\theta))}{\partial \theta})^\top$ is zero at all iterations in Figures 1b and 1c for all cases with $m_c = 1$.

the deep networks yielded the zero training error whenever Q_T is not linearly increasing over the time or, equivalently, whenever the condition of $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^T))}{\partial \theta^T})$ holds sufficiently many steps T. This is consistent with our theory.

Finally, we also confirmed that gradient representation $\frac{\partial f(x,\theta^t)}{\partial \theta^t}$ changed significantly from the initial one $\frac{\partial f(x,\theta^0)}{\partial \theta^0}$ in our experiments. That is, the

Table 1: The Change of the Gradient Representation during Training, $\|\mathbf{M}(\theta^{\hat{T}}) - \mathbf{M}(\theta^0)\|_F^2$, Where $\mathbf{M}(\theta) := \frac{\partial \text{vec}(f_X(\theta))}{\partial \theta}(\frac{\partial \text{vec}(f_X(\theta))}{\partial \theta})^\top$ and $\hat{T}$ Is the Last Time Step.

a. Fully-Connected Network

Data Set	
Two moons	2.09×10^{11}
Sine wave	3.95×10^{9}

b. Convolutional Network

	$m_c = 4$		$m_c = 2$		$m_c = 1$	
Data Set	seed#1	seed#2	seed#1	seed#2	seed#1	seed#2
Semeion	8.09×10^{12}	5.19×10^{12}	9.82×10^{12}	3.97×10^{12}	2.97×10^{12}	5.41×10^{12}
Random	3.73×10^{12}	1.64×10^{12}	3.43×10^{7}	4.86×10^{12}	1.40×10^{7}	8.57×10^{11}

values of $\|\mathbf{M}(\theta^T) - \mathbf{M}(\theta^0)\|_F^2$ were significantly large and tended to increase as T increases, where the matrix $\mathbf{M}(\theta) \in \mathbb{R}^{nm_y \times nm_y}$ is defined by $\mathbf{M}(\theta) = \frac{\partial \text{vec}(f_X(\theta))}{\partial \theta}(\frac{\partial \text{vec}(f_X(\theta))}{\partial \theta})^\top$. Table 1 summarizes the values of $\|\mathbf{M}(\theta^T) - \mathbf{M}(\theta^0)\|_F^2$ at the end of the training.

2.3 Implications. In section 2.1.3, we showed that an uncommon model structure $f(x, \theta) = \theta^4 - 10\theta^2 + 6\theta + 100$ does not satisfy assumption 1, and assumption 1 is not required for global convergence if the minimum eigenvalue is nonzero. However, in practice, we typically use machine learning models that satisfy assumption 1 instead of the model $f(x, \theta) = \theta^4 - 10\theta^2 + 6\theta + 100$, and the minimum eigenvalue is zero in many cases. In this context, theorem 1 provides the justification for common practice in nonlinear representation learning. Furthermore, theorem 1i contributes to the literature by identifying the common model structure assumption (assumption 1) and the data-architecture alignment condition (definition 1) as the novel and practical conditions to ensure the global convergence even when the minimum eigenvalue becomes zero. Moreover, theorem 1ii shows that this condition is not arbitrary in the sense that it is also necessary to obtain the globally optimal models. Furthermore, the data-architecture alignment condition is strictly more general than the condition of the minimum eigenvalue being nonzero, in the sense that the latter implies the former but not vice versa.

Our new theoretical understanding based on the data-architecture alignment condition can explain and deepen the previously known empirical observation that increasing network size tends to improve training behaviors. Indeed, the size of networks seems to correlate well with the training error to a certain degree in Figure 1b. However, the size and the training error do not correlate well in Figure 1c. Our new theoretical understanding explains that the training behaviors correlate more directly with the data-architecture alignment condition of $\mathrm{vec}(Y_\ell) \in \mathrm{Col}(\frac{\partial \mathrm{vec}(f_X(\theta^t))}{\partial \theta^t})$ instead. The seeming correlation with the network size is indirect and caused by another correlation between the network size and the condition of $\mathrm{vec}(Y_\ell) \in \mathrm{Col}(\frac{\partial \mathrm{vec}(f_X(\theta^t))}{\partial \theta^t})$. That is, the condition of $\mathrm{vec}(Y_\ell) \in \mathrm{Col}(\frac{\partial \mathrm{vec}(f_X(\theta^t))}{\partial \theta^t})$ more likely tends to hold when the network size is larger because the matrix $\frac{\partial \mathrm{vec}(f_X(\theta^t))}{\partial \theta^t}$ is of size $nm_y \times d$ where d is the number of parameters: that is, by increasing d, we can increase the column space $\mathrm{Col}(\frac{\partial \mathrm{vec}(f_X(\theta^t))}{\partial \theta^t})$ to increase the chance of satisfying the condition of $\mathrm{vec}(Y_\ell) \in \mathrm{Col}(\frac{\partial \mathrm{vec}(f_X(\theta^t))}{\partial \theta^t})$.

Note that the minimum eigenvalue of the matrix $\mathbf{M}(\theta^t) = \frac{\partial \mathrm{vec}(f_X(\theta))}{\partial \theta}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial \theta})^\top$ is zero at all iterations in Figures 1b and 1c for all cases of $m_c = 1$. Thus, Figures 1b and 1c also illustrate the fact that while having the zero minimum eigenvalue of the matrix $\mathbf{M}(\theta^t)$, the dynamics can achieve the global convergence under the data-architecture alignment condition. Moreover, because the multilayer neural network in the lazy training regime (Kawaguchi & Sun, 2021) achieves zero training errors for *all* data sets, Figure 1 additionally illustrates that our theoretical and empirical results apply to the models outside of the lazy training regime and can distinguish "good" data sets from "bad" data sets given a learning algorithm.

In sum, our new theoretical understanding has the ability to explain and distinguish the successful case and failure case based on the data-architecture alignment condition for the common machine learning models. Because the data-architecture alignment condition is dependent on data and architecture, theorem 1, along with our experimental results, shows why and when the global convergence in nonlinear representation learning is achieved based on the relationship between the data (X, Y) and architecture f. This new understanding is used in section 3 to derive a practical algorithm and is expected to be a basis for many future algorithms.

2.4 Details and Formalization of Theorem 1. This section presents the precise mathematical statements that formalize the informal description of theorem 1. In the following sections, we devide the formal version of theorem 1 into theorem 2, theorem 3, and proposition 1.

2.4.1 Preliminaries. Let $(\theta^t)_{t=0}^\infty$ be the sequence defined by $\theta^{t+1} = \theta^t - \alpha^t \bar{g}^t$ with an initial parameter vector θ^0, a learning rate α^t, and an update vector $\bar{g}^t$. The analysis in this section relies on the following assumption on the update vector $\bar{g}^t$:

Assumption 2. There exist $\bar{c}, \underline{c} > 0$ such that $\underline{c}\|\nabla\mathcal{L}(\theta^t)\|^2 \leq \nabla\mathcal{L}(\theta^t)^\top \bar{g}^t$ and $\|\bar{g}^t\|^2 \leq \bar{c}\|\nabla\mathcal{L}(\theta^t)\|^2$ for any $t \geq 0$.

Assumption 2 is satisfied by using $\bar{g}^t = D^t\nabla\mathcal{L}(\theta^t)$, where D^t is any positive-definite symmetric matrix with eigenvalues in the interval $[\underline{c}, \sqrt{\bar{c}}]$.

If we set $D^t = I$, we have gradient descent, and assumption 2 is satisfied with $\underline{c} = \bar{c} = 1$. This section also uses the standard assumption of differentiability and Lipschitz continuity:

Assumption 3. For every $i \in [n]$, the function $\ell_i : q \mapsto \ell(q, y_i)$ is differentiable and convex, the map $f_i : \theta \mapsto f(x_i, \theta)$ is differentiable, and $\|\nabla\mathcal{L}(\theta) - \nabla\mathcal{L}(\theta')\| \leq L\|\theta - \theta'\|$ for all θ, θ' in the domain of $\mathcal{L}$ for some $L \geq 0$.

The assumptions on the loss function in assumption 3 are satisfied by using standard loss functions, including the squared loss, logistic loss, and cross-entropy loss. Although the objective function $\mathcal{L}$ is nonconvex and non-invex, the function $q \mapsto \ell(q, y_i)$ is typically convex.

For any matrix $Y^* = (y_1^*, y_2^*, \ldots, y_n^*)^\top \in \mathbb{R}^{n \times m_y}$, we define

$$\mathcal{L}^*(Y^*) = \frac{1}{n}\sum_{i=1}^{n} \ell\left(y_i^*, y_i\right). \tag{2.6}$$

For example, for the squared loss ℓ, the value of $\mathcal{L}^*(Y_\ell)$ is at most the global minimum value of $\mathcal{L}$ as

$$\mathcal{L}^*(Y_\ell) \leq \mathcal{L}(\theta),\ \forall\theta \in \mathbb{R}^d, \tag{2.7}$$

since $\mathcal{L}^*(Y_\ell) = \frac{1}{n}\sum_{i=1}^{n} \|y_i - y_i\|_2^2 = 0 \leq \mathcal{L}(\theta)\ \forall\theta \in \mathbb{R}^d$. This letter also uses the notation of $[k] = \{1, 2, \ldots, k\}$ for any $k \in \mathbb{N}^+$ and $\|\cdot\| = \|\cdot\|_2$ (Euclidean norm). Finally, we note that for any $\eta \in \mathbb{R}$, the condition of $\mathrm{vec}(Y_\ell) \in \mathrm{Col}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta})$ is necessary to learn a near-global optimal model $f_X(\theta) = \eta Y_\ell$:

Proposition 1. *Suppose assumption 1 holds. If* $\mathrm{vec}(Y_\ell) \notin \mathrm{Col}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial\theta})$*, then* $f_X(\theta) \neq \eta Y_\ell$ *for any* $\eta \in \mathbb{R}$*.*

Proof. All proofs of this letter are presented in appendix A in the supplementary information. □

2.4.2 Global Optimality at the Limit Point. The following theorem shows that every limit point $\hat{\theta}$ of the sequence $(\theta^t)_t$ achieves a loss value $\mathcal{L}(\hat{\theta})$ no worse than $\inf_{\eta\in\mathbb{R}} \mathcal{L}^*(\eta Y^*)$ for any Y^* such that $\mathrm{vec}(Y^*) \in \mathrm{Col}(\frac{\partial \mathrm{vec}(f_X(\theta^t))}{\partial\theta^t})$ for all $t \in [\tau, \infty)$ with some $\tau \geq 0$:

Theorem 2. *Suppose assumptions 1 to 3 hold. Assume that the learning rate sequence* $(\alpha^t)_t$ *satisfies either (i)* $\epsilon \leq \alpha^t \leq \frac{\underline{c}(2-\epsilon)}{L\bar{c}}$ *for some* $\epsilon > 0$ *or (ii)* $\lim_{t\to\infty} \alpha^t = 0$ *and* $\sum_{t=0}^{\infty} \alpha^t = \infty$*. Then for any* $Y^* \in \mathbb{R}^{n\times m_y}$*, if there exists* $\tau \geq 0$ *such that* $\mathrm{vec}(Y^*) \in \mathrm{Col}(\frac{\partial \mathrm{vec}(f_X(\theta^t))}{\partial\theta^t})$ *for all* $t \in [\tau, \infty)$*, every limit point* $\hat{\theta}$ *of the sequence* $(\theta^t)_t$ *satisfies*

$$\mathcal{L}(\hat{\theta}) \leq \mathcal{L}^*(\eta Y^*), \quad \forall\eta \in \mathbb{R}. \tag{2.8}$$

For example, for the squared loss $\ell(q, y) = \|q - y\|^2$, theorem 2 implies that every limit point $\hat{\theta}$ of the sequence $(\theta^t)_t$ is a global minimum as

$$\mathcal{L}(\hat{\theta}) \leq \mathcal{L}(\theta), \quad \forall \theta \in \mathbb{R}^d, \tag{2.9}$$

if $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$ for $t \in [\tau, \infty)$ with some $\tau \geq 0$. This is because $\mathcal{L}(\hat{\theta}) \leq \mathcal{L}^*(Y_\ell) \leq \mathcal{L}(\theta)\ \forall \theta \in \mathbb{R}^d$ from theorem 2 and equation 2.7.

In practice, one can easily satisfy all the assumptions in theorem 2 except for the condition that $\text{vec}(Y^*) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$ for all $t \in [\tau, \infty)$. Accordingly, we now weaken this condition by analyzing optimality at each iteration so that the condition is verifiable in experiments.

2.4.3 Global Optimality Gap at Each Iteration. The following theorem states that under standard settings, the sequence $(\theta^t)_{t \in \mathcal{T}}$ converges to a loss value no worse than $\inf_{\eta \in \mathbb{R}} \mathcal{L}^*(\eta Y^*)$ at the rate of $O(1/\sqrt{|\mathcal{T}|})$ for any $\mathcal{T}$ and Y^* such that $\text{vec}(Y^*) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$ for $t \in \mathcal{T}$:

Theorem 3. *Suppose assumptions 1 and 3 hold. Let* $(\alpha^t, \bar{g}^t) = (\frac{2\alpha}{L}, \nabla \mathcal{L}(\theta^t))$ *with an arbitrary* $\alpha \in (0, 1)$. *Then for any* $\mathcal{T} \subseteq \mathbb{N}_0$ *and* $Y^* \in \mathbb{R}^{n \times m_y}$ *such that* $\text{vec}(Y^*) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$ *for all* $t \in \mathcal{T}$, *it holds that*

$$\min_{t \in \mathcal{T}} \mathcal{L}(\theta^t) \leq \mathcal{L}^*(\eta Y^*) + \frac{1}{\sqrt{|\mathcal{T}|}} \sqrt{\frac{L \zeta_\eta \mathcal{L}(\theta^{t_0})}{2\alpha(1-\alpha)}}, \tag{2.10}$$

for any $\eta \in \mathbb{R}$, *where* $t_0 = \min\{t : t \in \mathcal{T}\}$, $\zeta_\eta := 4 \max_{t \in \mathcal{T}} \max(\|\nu(\theta^t)\|^2, \|\hat{\beta}(\theta^t, \eta)\|^2)$, $\hat{\beta}(\theta, \eta) := \eta((\frac{\partial \text{vec}(f_X(\theta))}{\partial \theta})^\top \frac{\partial \text{vec}(f_X(\theta))}{\partial \theta})^\dagger (\frac{\partial \text{vec}(f_X(\theta))}{\partial \theta})^\top \text{vec}(Y^*)$, *and* $\nu(\theta)_k = \theta_k$ *for all* $k \in S$ *and* $\nu(\theta)_k = 0$ *for all* $k \notin S$ *with the set* S *being defined in assumption 1.*

For the squared loss ℓ, theorem 3 implies the following for any $T \geq 1$: for any $\mathcal{T} \subseteq [T]$ such that $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$ for all $t \in \mathcal{T}$, we have

$$\min_{t \in [T]} \mathcal{L}(\theta^t) \leq \inf_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta) + O(1/\sqrt{|\mathcal{T}|}). \tag{2.11}$$

This is because $\mathcal{L}^*(Y_\ell) \leq \inf_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta)$ from equations 2.7 and $\min_{t \in [T]} \mathcal{L}(\theta^t) \leq \min_{t \in \mathcal{T}} \mathcal{L}(\theta^t)$ from $\mathcal{T} \subseteq [T]$.

Similarly, for the binary and multiclass cross-entropy losses, theorem 3 implies the following for any $T \geq 1$: for any $\mathcal{T} \subseteq [T]$ such that $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$ for all $t \in \mathcal{T}$, we have that for any $\eta \in \mathbb{R}$,

$$\min_{t \in [T]} \mathcal{L}(\theta^t) \leq \mathcal{L}^*(\eta Y_\ell) + O(\eta^2/\sqrt{|\mathcal{T}|}). \tag{2.12}$$

Given any desired $\epsilon > 0$, since $\mathcal{L}^*(\eta Y_\ell) \to 0$ as $\eta \to \infty$, setting η to be sufficiently large obtains the desired ϵ value as $\min_{t \in \mathcal{T}} \mathcal{L}(\theta^t) \leq \epsilon$ in equation 2.12 as $\sqrt{|\mathcal{T}|} \to \infty$.

3 Application to the Design of Training Framework

The results in the previous section show that the bound on the global optimality gap decreases per iteration whenever the data-architecture alignment condition holds. Using this theoretical understanding, in this section, we propose a new training framework with prior guarantees while learning hierarchical nonlinear representations without assuming the data-architecture alignment condition. As a result, we made significant improvements over the most closely related study on global convergence guarantees (Kawaguchi & Sun, 2021). In particular, whereas the related study requires a wide layer with a width larger than n, our results reduce the requirement to a layer with a width larger than $\sqrt{n}$. For example, the MNIST data set has $n = 60{,}000$ and hence previous studies require 60,000 neurons at a layer, whereas we only require $\sqrt{60{,}000} \approx 245$ neurons at a layer. Our requirement is consistent and satisfied by the models used in practice that typically have from 256 to 1024 neurons for some layers.

We begin in section 3.1 with additional notations and then present the training framework in section 3.2 and convergence analysis in section 3.3. We conclude in section 3.4 by providing empirical evidence to support our theory.

3.1 Additional Notations. We denote by $\theta_{(l)} \in \mathbb{R}^{d_l}$ the vector of all the trainable parameters at the lth layer for $l = 1, \ldots, H$ where H is the depth or the number of trainable layers (i.e., one plus the number of hidden layers). That is, the Hth layer is the last layer containing the trainable parameter vector $\theta_{(H)}$ at the last layer. For any pair (l, l') such that $1 \leq l \leq l' \leq H$, we define $\theta_{(l:l')} = [\theta_{(l)}^\top, \ldots, \theta_{(l')}^\top]^\top \in \mathbb{R}^{d_{l:l'}}$—for example, $\theta_{(1:H)} = \theta$ and $\theta_{(l:l')} = \theta_{(l)}$ if $l = l'$. We consider a family of training algorithms that update the parameter vector θ as follows: for each $l = 1, \ldots, H$,

$$\theta_{(l)}^{t+1} = \theta_{(l)}^t - g_{(l)}^t, \qquad g_{(l)}^t \sim \mathcal{G}_{(l)}(\theta^t, t), \tag{3.1}$$

where the function $\mathcal{G}_{(l)}$ outputs a distribution over the vector $g_{(l)}^t$ and differs for different training algorithms. For example, for (minibatch) stochastic gradient descent (SGD), $g_{(l)}^t$ represents a product of a learning rate and a stochastic gradient with respect to $\theta_{(l)}$ at the time t. We define $\mathcal{G} = (\mathcal{G}_{(1)}, \ldots, \mathcal{G}_{(H)})$ to represent a training algorithm.

For an arbitrary matrix $M \in \mathbb{R}^{m \times m'}$, we let M_{*j} be its jth column vector in $\mathbb{R}^m$, M_{i*} be its ith row vector in $\mathbb{R}^{m'}$, and rank(M) be its matrix rank. We define $M \circ M'$ to be the Hadamard product of any matrices M and M'.

For any vector $v \in \mathbb{R}^m$, we let $\mathrm{diag}(v) \in \mathbb{R}^{m \times m}$ be the diagonal matrix with $\mathrm{diag}(v)_{ii} = v_i$ for $i \in [m]$. We denote by I_m the $m \times m$ identity matrix.

3.2 Exploration-Exploitation Wrapper. In this section, we introduce the exploration-exploitation (EE) wrapper $\mathcal{A}$. The EE wrapper $\mathcal{A}$ is not a standalone training algorithm. Instead, it takes any training algorithm $\mathcal{G}$ as its input and runs the algorithm $\mathcal{G}$ in a particular way to guarantee global convergence. We note that the exploitation phase in the EE wrapper does not optimize the last layer; instead, it optimizes hidden layers, whereas the exploration phase optimizes all layers. The EE wrapper allows us to learn the representation $\frac{\partial f(x,\theta^t)}{\partial \theta^t}$ that differs significantly from the initial representation $\frac{\partial f(x,\theta^0)}{\partial \theta^0}$ without making assumptions on the minimum eigenvalue of the matrix $\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial \theta}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial \theta})^\top$ by leveraging the data-architecture alignment condition. The data-architecture alignment condition is ensured by the safe-exploration condition (defined in section 3.3.1), which is time independent and holds in practical common architectures (as demonstrated in section 3.4).

3.2.1 Main Mechanisms. Algorithm 1 outlines the EE wrapper $\mathcal{A}$. During the exploration phase in lines 3 to 7 of algorithm 1, the EE wrapper $\mathcal{A}$ freely explores hierarchical nonlinear representations to be learned without any restrictions. Then, during the exploitation phase in lines 8 to 12, it starts exploiting the current knowledge to ensure $\mathrm{vec}(Y_\ell) \in \mathrm{Col}(\frac{\partial \mathrm{vec}(f_X(\theta^t))}{\partial \theta^t})$ for all t to guarantee global convergence. The value of τ is the hyperparameter that controls the time when it transitions from the exploration phase to the exploitation phase.

In the exploitation phase, the wrapper $\mathcal{A}$ only optimizes the parameter vector $\theta^t_{(H-1)}$ at the $(H-1)$th hidden layer, instead of the parameter vector $\theta^t_{(H)}$ at the last layer or the Hth layer. Despite this, the EE wrapper $\mathcal{A}$ is proved to converge to global minima of all layers in $\mathbb{R}^d$. The exploitation phase still allows us to significantly change the representations as $\mathbf{M}(\theta^t) \not\approx \mathbf{M}(\theta^\tau)$ for $t > \tau$. This is because we optimize the hidden layers instead of the last layer without any significant overparameterizations.

The exploitation phase uses an arbitrary optimizer $\tilde{\mathcal{G}}$ with the update vector $\tilde{g}^t \sim \tilde{\mathcal{G}}_{(H-1)}(\theta^t, t)$ with $\tilde{g}^t = \alpha^t \hat{g}^t \in \mathbb{R}^{d_{H-1}}$. During the two phases, we can use the same optimizers (e.g., SGD for both $\mathcal{G}$ and $\tilde{\mathcal{G}}$) or different optimizers (e.g., SGD for $\mathcal{G}$ and L-BFGS for $\tilde{\mathcal{G}}$).

3.2.2 Model Modification. This section defines the details of the model modification at line 2 of algorithm 1. Given any base network $\bar{f}$, the wrapper $\mathcal{A}$ first checks whether the last two layers of the given network $\bar{f}$ are fully connected. If not, one or two fully connected last layers are added such that the output of the network $\bar{f}$ can be written by $\bar{f}(x, \bar{\theta}) =$

Algorithm 1: $\mathcal{A}$: Exploration-Exploitation (EE) Wrapper.

```
1:  Inputs: a training algorithm $\mathcal{G}$ and a base model $\bar{f}$.
2:  Modify the base model $\bar{f}$ to the model $f$
3:  Exploration phase:
4:  Initialize the parameter vector $\theta^0$ of the model $f$
5:      for $t = 0, 1, \ldots, \tau - 1$ do
6:          $\theta_{(l)}^{t+1} = \theta_{(l)}^{t} - g_{(l)}^{t},\ \ g_{(l)}^{t} \sim \mathcal{G}_{(l)}(\theta^t, t),\ \forall l \in [H]$
7:      end for
8:  Exploitation phase:
9:      Set $\theta^\tau = \theta^{\tau-1} + \varepsilon\delta$ where $\delta \sim \mathcal{N}_d(0, I)$
10:     Replace $\sigma_j$ by $\sigma_{R^{(j)}}$ with $R^{(j)} = \theta^{\tau}_{(H-1,j)}$
11:     for $t = \tau, \tau + 1, \ldots$ do
12:         $\theta_{(H-1)}^{t+1} = \theta_{(H-1)}^{t} - \tilde{g}^{t},\ \ \tilde{g}^{t} \sim \tilde{\mathcal{G}}_{(H-1)}(\theta^t, t)$
13:     end for
```

$\bar{W}^{(H)}\bar{\sigma}(\bar{W}^{(H-1)}z(x, \theta_{(1:H-2)}))$. Here, $z(x, \theta_{(1:H-2)})$ is the output of the $(H-2)$th layer, and the function z is arbitrary and can represent various deep networks. Moreover, $\bar{\sigma}$ is a nonlinear activation function, and $\bar{W}^{(H-1)}$ and $\bar{W}^{(H)}$ are the weight matrices of the last two layers. The wrapper $\mathcal{A}$ then modifies these last two layers as follows. In the case of $m_y = 1$, the model $\bar{f}$ is modified to

$$f(x, \theta) = W^{(H)}\sigma(W^{(H-1)}, z(x, \theta_{(1:H-2)})), \tag{3.2}$$

where $W^{(H-1)} \in \mathbb{R}^{m_H \times m_{H-1}}$ and $W^{(H)} \in \mathbb{R}^{m_y \times m_H}$ are the weight matrices of the last two layers. The nonlinear activation σ is defined by $\sigma(q, q') = \tilde{\sigma}(qq') \circ (qq')$, where $\tilde{\sigma}$ is some nonlinear function. For example, we can set $\tilde{\sigma}(q) = \frac{1}{1+e^{-\varsigma' q}}$ (sigmoid) with any hyperparameter $\varsigma' > 0$, for which it holds that as $\varsigma' \to \infty$,

$$f(x, \theta) \to W^{(H)}\mathrm{relu}(W^{(H-1)}z(x, \theta_{(1:H-2)})). \tag{3.3}$$

We generalize equation 3.2 to the case of $m_y \geq 2$ as

$$f(x,\theta)_j = W^{(H)}_{j*}\sigma_j(W^{(H-1,j)}, z(x,\theta_{(1:H-2)})), \tag{3.4}$$

for $j \in [m_y]$ where $W^{(H-1,j)} \in \mathbb{R}^{m_H \times m_{H-1}}$ is the weight matrix at the $(H-1)$th layer and $\sigma_j = \sigma$ until line 10. At line 10, the wrapper $\mathcal{A}$ replaces σ_j by $\sigma_{R^{(j)}}$ where $\sigma_{R^{(j)}}(q, q') = \tilde{\sigma}(R^{(j)}q') \circ (qq')$ with $R^{(j)} = \theta^{\tau}_{(H-1,j)}$ and $\theta_{(H-1,j)} = (W^{(H-1,j)})^\top$. To consider the bias term, we include the constant neuron to the output of the $(H-1)$th layer as $z(x,\theta_{(1:H-2)}) = [\bar{z}(x,\theta_{(1:H-2)})^\top, 1]^\top \in \mathbb{R}^{m_{H-1}}$, where $\bar{z}(x,\theta_{(1:H-2)})$ is the output without the constant neuron.

3.3 Convergence Analysis. In this section, we establish global convergence of the EE wrapper $\mathcal{A}$ without using assumptions from the previous section. Let τ be an arbitrary positive integer and ε be an arbitrary positive real number. Let $(\theta^t)_{t=0}^{\infty}$ be a sequence generated by the EE wrapper $\mathcal{A}$. We define $\hat{\mathcal{L}}(\theta_{(H-1)}) = \mathcal{L}(\theta^{\tau}_{(1:H-2)}, \theta_{(H-1)}, \theta^{\tau}_{(H)})$ and $B_{\bar{\epsilon}} = \min_{\theta_{(H-1)} \in \Theta_{\bar{\epsilon}}} \|\theta_{(H-1)} - \theta^{\tau}_{(H-1)}\|$ where $\Theta_{\bar{\epsilon}} = \operatorname{argmin}_{\theta_{(H-1)}} \max(\hat{\mathcal{L}}(\theta_{(H-1)}), \bar{\epsilon})$ for any $\bar{\epsilon} \geq 0$.

3.3.1 Safe-Exploration Condition. The mathematical analysis in this section relies on the safe-exploration condition, which allows us to safely explore deep nonlinear representations in the exploration phase without getting stuck in the states of $\operatorname{vec}(Y_\ell) \notin \operatorname{Col}(\frac{\partial \operatorname{vec}(f_X(\theta^t))}{\partial \theta^t})$. The safe-exploration condition is verifiable, time-independent, data-dependent, and architecture-dependent. The verifiability and time independence make the assumption strong enough to provide prior guarantees before training. The data dependence and architecture dependence make the assumption weak enough to be applicable for a wide range of practical settings.

For any $q \in \mathbb{R}^{m_{H-1} \times m_H}$, we define the matrix-valued function $\phi(q, \theta_{(1:H-2)}) \in \mathbb{R}^{n \times m_H m_{H-1}}$ by

$$\phi(q,\theta_{(1:H-2)}) = \begin{bmatrix} \tilde{\sigma}(z_1^\top q_{*1}) z_1^\top & \cdots & \tilde{\sigma}(z_1^\top q_{*m_H}) z_1^\top \\ \vdots & \ddots & \vdots \\ \tilde{\sigma}(z_n^\top q_{*1}) z_n^\top & \cdots & \tilde{\sigma}(z_n^\top q_{*m_H}) z_n^\top \end{bmatrix},$$

where $z_i = z(x_i, \theta_{(1:H-2)}) \in \mathbb{R}^{m_{H-1}}$ and $\tilde{\sigma}(z_i^\top q_{*k}) z_i^\top \in \mathbb{R}^{1 \times m_{H-1}}$ for all $i \in [n]$ and $k \in [m_H]$. Using this function, the safe exploration condition is formally stated as:

Assumption 4 (Safe-Exploration Condition). There exist a $q \in \mathbb{R}^{m_{H-1} \times m_H}$ and a $\theta_{(1:H-2)} \in \mathbb{R}^{d_{1:H-2}}$ such that $\operatorname{rank}(\phi(q, \theta_{(1:H-2)})) = n$.

The safe-exploration condition asks for only the existence of one parameter vector in the network architecture such that $\operatorname{rank}(\phi(q, \theta_{(1:H-2)})) = n$. It is not about the training trajectory $(\theta^t)_t$. Since the matrix $\phi(q, \theta_{(1:H-2)})$ is of size $n \times m_H m_{H-1}$, the safe-exploration condition does not require any wide layer of size $m_H \geq n$ or $m_{H-1} \geq n$. Instead, it requires a layer of size $m_H m_{H-1} \geq n$. This is a significant improvement over the most closely related study (Kawaguchi & Sun, 2021) where the wide layer of size $m_H \geq n$ was required. Note that having $m_H m_{H-1} \geq n$ does not imply the safe-exploration condition. Instead, $m_H m_{H-1} \geq n$ is a necessary condition to satisfy the safe-exploration condition, whereas $m_H \geq n$ or $m_{H-1} \geq n$ was a necessary condition to satisfy assumptions in previous papers, including the most closely related study (Kawaguchi & Sun, 2021). The safe-exploration condition is verified in experiments in section 3.4.

3.3.2 Additional Assumptions. We also use the following assumptions:

Assumption 5. For any $i \in [n]$, the function $\ell_i : q \mapsto \ell(q, y_i)$ is differentiable, and $\|\nabla \ell_i(q) - \nabla \ell_i(q')\| \leq L_\ell \|q - q'\|$ for all $q, q' \in \mathbb{R}$.

Assumption 6. For each $i \in [n]$, the functions $\theta_{(1:H-2)} \mapsto z(x_i, \theta_{(1:H-2)})$ and $q \mapsto \tilde{\sigma}(q)$ are real analytic.

Assumption 5 is satisfied by using standard loss functions such as the squared loss $\ell(q, y) = \|q - y\|^2$ and cross-entropy loss $\ell(q, y) = -\sum_{k=1}^{d_y} y_k \log \frac{\exp(q_k)}{\sum_{k'} \exp(q_{k'})}$. The assumptions of the invexity and convexity of the function $q \mapsto \ell(q, y_i)$ in sections 3.3.3 and 3.3.4 also hold for these standard loss functions. Using L_ℓ in assumption 5, we define $\hat{L} = \frac{L_\ell}{n}\|Z\|^2$, where $Z \in \mathbb{R}^n$ is defined by $Z_i = \max_{j \in [m_y]} \|[\mathrm{diag}(\theta^\tau_{(H,j)}) \otimes I_{m_{H-1}}](\phi(\theta^\tau_{(H-1,j)}, \theta^\tau_{(1:H-2)})_{i*})^\top\|$ with $\theta_{(H,j)} = (W^{(H)}_{j*})^\top$.

Assumption 6 is satisfied by using any analytic activation function such as sigmoid, hyperbolic tangents, and softplus activations $q \mapsto \ln(1 + \exp(\varsigma q))/\varsigma$ with any hyperparameter $\varsigma > 0$. This is because a composition of real analytic functions is real analytic, and the following are all real analytic functions in $\theta_{(1:H-2)}$: the convolution, affine map, average pooling, skip connection, and batch normalization. Therefore, the assumptions can be satisfied by using a wide range of machine learning models, including deep neural networks with convolution, skip connection, and batch normalization. Moreover, the softplus activation can approximate the ReLU activation for any desired accuracy, that is, $\ln(1 + \exp(\varsigma q))/\varsigma \to \mathrm{relu}(q)$ as $\varsigma \to \infty$, where ReLU represents the ReLU activation.

3.3.3 Global Optimality at the Limit Point. The following theorem proves the global optimality at limit points of the EE wrapper with a wide range of optimizers, including gradient descent and modified Newton methods:

Theorem 4. *Suppose assumptions 4 to 6 hold and that the function $\ell_i : q \mapsto \ell(q, y_i)$ is invex for any $i \in [n]$. Assume that there exist $\bar{c}, \underline{c} > 0$ such that $\underline{c}\|\nabla \hat{\mathcal{L}}(\theta^t_{(H-1)})\|^2 \leq \nabla \hat{\mathcal{L}}(\theta^t_{(H-1)})^\top \hat{g}^t$ and $\|\hat{g}^t\|^2 \leq \bar{c}\|\nabla \hat{\mathcal{L}}(\theta^t_{(H-1)})\|^2$ for any $t \geq \tau$.*

Assume that the learning rate sequence $(\alpha^t)_{t\geq\tau}$ *satisfies either (i)* $\epsilon \leq \alpha^t \leq \frac{\underline{c}(2-\epsilon)}{\hat{L}\bar{c}}$ *for some* $\epsilon > 0$ *or (ii)* $\lim_{t\to\infty} \alpha^t = 0$ *and* $\sum_{t=\tau}^{\infty} \alpha^t = \infty$*. Then with probability one, every limit point* $\hat{\theta}$ *of the sequence* $(\theta^t)_t$ *is a global minimum of* $\mathcal{L}$ *as* $\mathcal{L}(\hat{\theta}) \leq \mathcal{L}(\theta)$ *for all* $\theta \in \mathbb{R}^d$*.*

3.3.4 Global Optimality Gap at Each Iteration. We now present global convergence guarantees of the EE wrapper $\mathcal{A}$ with gradient decent and SGD:

Theorem 5. *Suppose assumptions 4 to 6 hold and that the function* $\ell_i : q \mapsto \ell(q, y_i)$ *is convex for any* $i \in [n]$*. Then, with probability one, the following two statements hold:*

(i) (Gradient descent) if $\hat{g}^t = \nabla\hat{\mathcal{L}}(\theta^t_{(H-1)})$ *and* $\alpha^t = \frac{1}{\hat{L}}$ *for* $t \geq \tau$*, then for any* $\bar{\epsilon} \geq 0$ *and* $t > \tau$*,*

$$\mathcal{L}(\theta^t) \leq \inf_{\theta\in\mathbb{R}^d} \max(\mathcal{L}(\theta), \bar{\epsilon}) + \frac{B_{\bar{\epsilon}}^2 \hat{L}}{2(t-\tau)}.$$

(ii) (SGD) if $\mathbb{E}[\hat{g}^t | \theta^t] = \nabla\hat{\mathcal{L}}(\theta^t_{(H-1)})$ *(almost surely) with* $\mathbb{E}[\|\hat{g}^t\|^2] \leq G^2$*, and if* $\alpha^t \geq 0$*,* $\sum_{t=\tau}^{\infty}(\alpha^t)^2 < \infty$ *and* $\sum_{t=\tau}^{\infty} \alpha^t = \infty$ *for* $t \geq \tau$*, then for any* $\bar{\epsilon} \geq 0$ *and* $t > \tau$*,*

$$\mathbb{E}[\mathcal{L}(\theta^{t^*})] \leq \inf_{\theta\in\mathbb{R}^d} \max(\mathcal{L}(\theta), \bar{\epsilon}) + \frac{B_{\bar{\epsilon}}^2 + G^2 \sum_{k=\tau}^{t} \alpha_k^2}{2\sum_{k=\tau}^{t} \alpha_k} \tag{3.5}$$

where $t^* \in \operatorname{argmin}_{k\in\{\tau,\tau+1,\ldots,t\}} \mathcal{L}(\theta^k)$*.*

In theorem 5ii, with $\alpha^t \sim O(1/\sqrt{t})$, the optimality gap becomes

$$\mathbb{E}[\mathcal{L}(\theta^{t^*})] - \inf_{\theta\in\mathbb{R}^d} \max(\mathcal{L}(\theta), \bar{\epsilon}) = \tilde{O}(1/\sqrt{t}).$$

3.4 Experiments. This section presents empirical evidence to support our theory and what is predicted by a well-known hypothesis. We note that there is no related work or algorithm that can guarantee global convergence in the setting of our experiments where the model has convolutions, skip connections, and batch normalizations without any wide layer (of the width larger than n). Moreover, unlike any previous studies that propose new methods, our training framework works by modifying any given method.

3.4.1 Sine Wave Data Set. We have seen in section 2.2 that gradient descent gets stuck at suboptimal points for the sine wave data set. Using the same setting as that in section 2.2 with $\varepsilon = 0.01$, $\tau = 2000$, and $\tilde{\mathcal{G}} = \mathcal{G}$, we confirm in Figure 2 that the EE wrapper $\mathcal{A}$ can modify gradient descent to avoid suboptimal points and converge to global minima as predicted by our

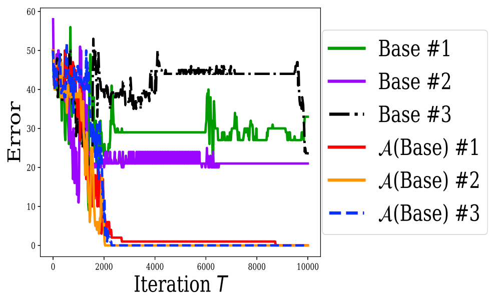


Figure 2: Training errors for three random trials.

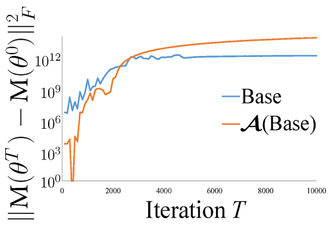


Figure 3: Changes of the representations where $\mathbf{M}(\theta) := \frac{\partial \mathrm{vec}(f_X(\theta))}{\partial \theta}(\frac{\partial \mathrm{vec}(f_X(\theta))}{\partial \theta})^\top$.

theory. Figure 3 shows the value of the change of the gradient representation, $\|\mathbf{M}(\theta^T) - \mathbf{M}(\theta^0)\|_F^2$, for each time step T. As can be seen, the values of $\|\mathbf{M}(\theta^T) - \mathbf{M}(\theta^0)\|_F^2$ are large for both methods. Notably, the EE wrapper $\mathcal{A}$ of the base case significantly increases the value of $\|\mathbf{M}(\theta^T) - \mathbf{M}(\theta^0)\|_F^2$ even in the exploitation phase after $\tau = 2000$ as we are optimizing the hidden layer. See appendix B in the supplementary information for more details of the experiments for the sine wave data set.

3.4.2 Image Data Sets. The standard convolutional ResNet with 18 layers (He, Zhang, Ren, & Sun, 2016) is used as the base model $\bar{f}$. We use ResNet-18 for the illustration of our theory because it is used in practice and it has convolution, skip connections, and batch normalization without any width larger than the number of data points. This setting is not covered by any of the previous theories for global convergence. We set the activation to be the softplus function $q \mapsto \ln(1 + \exp(\varsigma q))/\varsigma$ with $\varsigma = 100$ for all layers of the base ResNet. This approximates the ReLU activation well, as shown in appendix C in the supplementary information. We employ the cross-entropy loss and $\tilde{\sigma}(q) = \frac{1}{1+e^{-q}}$. We use a standard algorithm, SGD, with its standard hyperparameter setting for the training algorithm $\mathcal{G}$ with $\tilde{\mathcal{G}} = \mathcal{G}$—that is, we let the minibatch size be 64, the weight decay rate be 10^{-5}, the momentum coefficient be 0.9, the learning rate be $\alpha^t = 0.1$, and the last epoch

Table 2: Verification of the Safe-Exploration Condition (Assumption 4).

Data Set	n	m_{H-1}	m_H	Assumption 4
MNIST	60,000	513	234	Verified
CIFAR-10	50,000	513	195	Verified
CIFAR-100	50,000	513	195	Verified
Semeion	1,000	513	4	Verified
KMNIST	60,000	513	234	Verified
SVHN	73,257	513	286	Verified

Note: $m_H = \lceil 2(n/m_{H-1}) \rceil$ where n is the number of training data, m_H is the width of the last hidden layer, and m_{H-1} is the width of the penultimate hidden layer.

$\hat{T}$ be 200 (with data augmentation) and 100 (without data augmentation). The hyperparameters ε and $\tau = \tau_0 \hat{T}$ were selected from $\varepsilon \in \{10^{-3}, 10^{-5}\}$ and $\tau_0 \in \{0.4, 0.6, 0.8\}$ by only using training data. That is, we randomly divided each training data set (100%) into a smaller training data set (80%), and a validation data set (20%) for a grid search over the hyperparameters. See appendix B in the supplementary information for the results of the grid search and details of the experimental setting. This standard setting satisfies assumptions 5 and 6, leaving assumption 4 to be verified.

Verification of Assumption 4. Table 2 summarizes the verification results of the safe-exploration condition. Because the condition only requires an existence of a pair (θ, q) satisfying the condition, we verified it by using a randomly sampled q from the standard normal distribution and a θ returned by a common initialization scheme (He et al., 2015). As $m_{H-1} = 513$ ($512 +$ the constant neuron for the bias term) for the standard ResNet, we set $m_H = \lceil 2(n/m_{H-1}) \rceil$ throughout all the experiments with the ResNet. For each data set, the rank condition was verified twice by the two standard methods: one from Press, Teukolsky, Vetterling, and Flannery (2007) and another from Golub and Van Loan (1996).

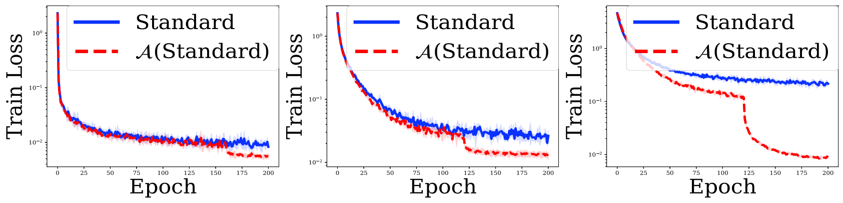
Test Performance. One well-known hypothesis is that the success of deep-learning methods partially comes from its ability to automatically learn deep nonlinear representations suitable for making accurate predictions from data (LeCun et al., 2015). As the EE wrapper $\mathcal{A}$ keeps this ability of representation learning, the hypothesis suggests that the test performance of the EE wrapper $\mathcal{A}$ of a standard method is approximately comparable to that of the standard method. Unlike typical experimental studies, our objective here is to confirm this prediction instead of showing improvements over a previous method. We empirically confirmed the prediction in Tables 3 and 4 where the numbers indicate the mean test errors (and standard deviations are in parentheses) over five random trials. As expected, the values

Table 3: Test Error (%): Data Augmentation.

Data Set	Standard	$\mathcal{A}$(Standard)
MNIST	0.40 (0.05)	0.30 (0.05)
CIFAR-10	7.80 (0.50)	7.14 (0.12)
CIFAR-100	32.26 (0.15)	28.38 (0.42)
Semeion	2.59 (0.57)	2.56 (0.55)
KMNIST	1.48 (0.07)	1.36 (0.11)
SVHN	4.67 (0.05)	4.43 (0.11)

Table 4: Test Error (%): No Data Augmentation.

Data Set	Standard	$\mathcal{A}$(Standard)
MNIST	0.52 (0.16)	0.49 (0.02)
CIFAR-10	15.15 (0.87)	14.56 (0.38)
CIFAR-100	54.99 (2.29)	46.13 (1.80)



(a) MNIST (b) CIFAR-10 (c) CIFAR-100

Figure 4: Training Loss with Data Augmentation.

of $\|\mathbf{M}(\theta^{\hat{T}}) - \mathbf{M}(\theta^0)\|_2^2$ were also large—for example, 4.64×10^{12} for the standard method and 3.43×10^{12} for wrapper $\mathcal{A}$ of the method with the Semeion data set.

Training Behavior. Figure 4 shows that the EE wrapper $\mathcal{A}$ can improve training loss values of the standard SGD algorithm in the exploitation phase without changing its hyperparameters because $\tilde{\mathcal{G}} = \mathcal{G}$ in these experiments. In the figure, the plotted lines indicate the mean values over five random trials, and the shaded regions show error bars with 1 standard deviation.

Computational Time. The EE wrapper $\mathcal{A}$ runs the standard SGD $\mathcal{G}$ in the exploration phase and the SGD $\tilde{\mathcal{G}} = \mathcal{G}$ only on the subset of the weights $\theta_{(H-1)}$ in the exploitation phase. Thus, the computational time of the EE wrapper $\mathcal{A}$ is similar to that of the SGD in the exploration phase, and it

Table 5: Total Wall-Clock Time in a Local GPU Workstation.

Data Set	Standard	$\mathcal{A}$(Standard)
Semeion	364.60 (0.94)	356.82 (0.67)
CIFAR-10	3616.92 (10.57)	3604.5 (6.80)

Table 6: Test Errors (%) of $\mathcal{A}$(Standard) with $\tilde{\mathcal{G}}$ = L-BFGS.

(a) with data augmentation				(b) without data augmentation			
ε \ τ_0	0.4	0.6	0.8	ε \ τ_0	0.4	0.6	0.8
10^{-3}	0.26	0.38	0.37	10^{-3}	0.36	0.43	0.42
10^{-5}	0.37	0.32	0.37	10^{-5}	0.42	0.35	0.35

tends to be faster than the SGD in the exploitation phase. To confirm this, we measure computational time with the Semeion and CIFAR-10 data sets under the same computational resources (e.g., without running other jobs in parallel) in a local workstation for each method. The mean wall-clock time (in seconds) over five random trials is summarized in Table 5, where the numbers in parentheses are standard deviations. It shows that the EE wrapper $\mathcal{A}$ is slightly faster than the standard method, as expected.

Effect of Learning Rate and Optimizer. We also conducted experiments on the effects of learning rates and optimizers using the MNIST data set with data augmentation. Using the best learning rate from {0.2, 0.1, 0.01, 0.001} for each method (with $\tilde{\mathcal{G}} = \mathcal{G} =$ SGD), the mean test errors (%) over five random trials were 0.33 (0.03) for the standard base method and 0.27 (0.03) for the $\mathcal{A}$ wrapper of the standard base method (the numbers in parentheses are standard deviations). Moreover, Table 6 reports the preliminary results on the effect of optimizers with $\tilde{\mathcal{G}}$ being set to the limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (L-BFGS) (with $\mathcal{G}$ = the standard SGD). By comparing Tables 3 and 6, we can see that using a different optimizer in the exploitation phase can potentially lead to performance improvements. A comprehensive study of this phenomenon is left to future work.

4 Conclusion

Despite the nonlinearity of the dynamics and the noninvexity of the objective, we have rigorously proved convergence of training dynamics to global minima for nonlinear representation learning. Our results apply to a wide range of machine learning models, allowing both underparameterization and overparameterization. For example, our results are applicable to the case where the minimum eigenvalue of the matrix $\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})^\top$ is zero for all $t \geq 0$. Under the common model structure assumption, models that cannot achieve zero error for all data sets (except some "good" data sets) are shown to achieve global optimality with zero error exactly when the dynamics satisfy the data-architecture alignment condition. Our results provide guidance for choosing and designing model structure and algorithms via the common model structure assumption and data-architecture alignment condition.

The key limitation in our analysis is the differentiability of the function f. For multilayer neural networks, this is satisfied by using standard activation functions, such as softplus, sigmoid, and hyperbolic tangents. Whereas softplus can approximate ReLU arbitrarily well, the direct treatment of ReLU in nonlinear representation learning is left to future work.

Our theoretical results and numerical observations uncover novel mathematical properties and provide a basis for future work. For example, we have shown global convergence under the data-architecture alignment condition $\text{vec}(Y_\ell) \in \text{Col}(\frac{\partial \text{vec}(f_X(\theta^t))}{\partial \theta^t})$. The EE wrapper $\mathcal{A}$ is only one way to ensure this condition. There are many other ways to ensure the data-architecture alignment condition, and each way can result in a new algorithm with guarantees.

References

Bartlett, P. L., Helmbold, D. P., & Long, P. M. (2019). Gradient descent with identity initialization efficiently learns positive-definite linear transformations by deep residual networks. *Neural Computation*, *31*(3), 477–502. 10.1162/neco_a_01164, PubMed: 30645179

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828. 10.1109/TPAMI.2013.50, PubMed: 23787338

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2007). Greedy layerwise training of deep networks. In B. Schölkopf, J. Platt, & T. Hoffman (Eds.), *Advances in neural information processing systems*, *19* (p. 153). Red Hook. NY: Curran.

Bordes, A., Glorot, X., Weston, J., & Bengio, Y. (2012). Joint learning of words and meaning representations for open-text semantic parsing. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, *22* (pp. 127–135).

Ciregan, D., Meier, U., & Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3642–3649). Piscataway, NJ: IEEE.

Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., & Ha, D. (2019). Deep learning for classical Japanese literature. In *NeurIPS Creativity Workshop 2019*.

Dahl, G., Ranzato, M., Mohamed, A.-r., & Hinton, G. E. (2010). Phone recognition with the mean-covariance restricted Boltzmann machine. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, & A. Culotta (Eds.), *Advances in neural information processing systems, 23* (pp. 469–477). Red Hook, NY: Curran.

Dahl, G. E., Yu, D., Deng, L., & Acero, A. (2011). Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, *20*(1), 30–42. 10.1109/TASL.2011.2134090

Deng, L., Seltzer, M. L., Yu, D., Acero, A., Mohamed, A.-r., & Hinton, G. (2010). Binary coding of speech spectrograms using a deep auto-encoder. In *Proceedings of the Eleventh Annual Conference of the International Speech Communication Association*. Red Hook, NY: Curran.

Dong, C., Loy, C. C., He, K., & Tang, X. (2014). Learning a deep convolutional network for image super-resolution. In *Proceedings of the European Conference on Computer Vision* (pp. 184–199). Berlin: Springer.

Gatys, L. A., Ecker, A. S., & Bethge, M. (2016). Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2414–2423). Piscataway, NJ: IEEE.

Glorot, X., Bordes, A., & Bengio, Y. (2011). Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the International Conference on Machine Learning*. New York: ACM.

Golub, G. H., & Van Loan, C. F. (1996). *Matrix computations*. Baltimore, MD: Johns Hopkins University Press.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1026–1034). Piscataway, NJ: IEEE.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Identity mappings in deep residual networks. In *Proceedings of the European Conference on Computer Vision* (pp. 630–645). Berlin: Springer.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., . . . Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(6), 82–97. 10.1109/MSP.2012.2205597

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), 1527–1554. 10.1162/neco.2006.18.7.1527, PubMed: 16764513

Kawaguchi, K. (2016). Deep learning without poor local minima. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in neural information processing systems, 29* (pp. 586–594). Red Hook, NY: Curran.

Kawaguchi, K. (2021). On the theory of implicit deep learning: Global convergence with implicit layers. In *Proceedings of the International Conference on Learning Representations*.

Kawaguchi, K., Kaelbling, L. P., & Lozano-Pérez, T. (2015). Bayesian optimization with exponential convergence. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in neural information processing systems, 28* (pp. 2809–2817). Red Hook. NY: Curran.

Kawaguchi, K., Maruyama, Y., & Zheng, X. (2016). Global continuous optimization with error bound and fast convergence. *Journal of Artificial Intelligence Research*, *56*, 153–195. 10.1613/jair.4742

Kawaguchi, K., & Sun, Q. (2021). A recipe for global convergence guarantee in deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, *35* (pp. 8074–8082). Palo Alto, CA: AAAI.

Krizhevsky, A., & Hinton, G. (2009). *Learning multiple layers of features from tiny images* (Technical report). Citeseer.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems*, *25* (pp. 1097–1105). Red Hook. NY: Curran.

Laurent, T., & Brecht, J. (2018). Deep linear networks with arbitrary loss: All local minima are global. In *Proceedings of the International Conference on Machine Learning* (pp. 2902–2907).

Le, H.-S., Oparin, I., Allauzen, A., Gauvain, J.-L., & Yvon, F. (2012). Structured output layer neural network language models for speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, *21*(1), 197–206.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. 10.1038/nature14539, PubMed: 26017442

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, *86*(11), 2278–2324. 10.1109/5.726791

Luan, F., Paris, S., Shechtman, E., & Bala, K. (2017). Deep photo style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4990–4998). Piscataway, NJ: IEEE.

Mityagin, B. (2015). *The zero set of a real analytic function.* arXiv:1512.07276.

Mohamed, A.-r., Dahl, G. E., & Hinton, G. (2011). Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech, and Language Processing*, *20*(1), 14–22. 10.1109/TASL.2011.2109382

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., . . . Chintala, S. (2019a). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems*, *32* (pp. 8026–8037). Red Hook. NY: Curran.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., . . . Chintala, S. (2019b). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems*, *32* (pp. 8024–8035). Red Hook. NY: Curran.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, . . . Duchesnay, E., (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Poggio, T., Kawaguchi, K., Liao, Q., Miranda, B., Rosasco, L., Boix, X., Hidary, J., & Mhaskar, H. (2017). *Theory of deep learning III: Explaining the non-overfitting puzzle*. arXiv:1801.00173.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical recipes: The art of scientific computing*. (3rd ed.). Cambridge: Cambridge University Press.

Rifai, S., Dauphin, Y. N., Vincent, P., Bengio, Y., & Muller, X. (2011). The manifold tangent classifier. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems*, *24* (pp. 2294–2302). Red Hook. NY: Curran.

Saxe, A. M., McClelland, J. L., & Ganguli, S. (2014). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *Proceedings of the International Conference on Learning Representations*.

Schwenk, H., Rousseau, A., & Attik, M. (2012). Large, pruned or continuous space language models on a GPU for statistical machine translation. In *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT* (pp. 11–19). Stroudsburg, PA: Association for Computational Linguistics.

Seide, F., Li, G., & Yu, D. (2011). Conversational speech transcription using context-dependent deep neural networks. In *Proceedings of the Twelfth Annual Conference of the International Speech Communication Association*. New York: ACM.

Socher, R., Huang, E. H., Pennington, J., Ng, A. Y., & Manning, C. D. (2011). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems*, *24* (pp. 801–809). Red Hook, NY: Curran.

Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., & Manning, C. D. (2011). Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing* (pp. 151–161). Stroudsburg, PA: Association for Computational Linguistics.

Srl, B. T., & Brescia, I. (1994). *Semeion handwritten digit data set*. Rome, Italy: Semeion Research Center of Sciences of Communication.

Turner, C. R., Fuggetta, A., Lavazza, L., & Wolf, A. L. (1999). A conceptual basis for feature engineering. *Journal of Systems and Software*, *49*(1), 3–15. 10.1016/S0164-1212(99)00062-X

Xu, K., Zhang, M., Jegelka, S., & Kawaguchi, K. (2021). Optimization of graph neural networks: Implicit acceleration by skip connections and more depth. In *Proceedings of the International Conference on Machine Learning*.

Zheng, A., & Casari, A. (2018). *Feature engineering for machine learning: Principles and techniques for data scientists*. Sebastopol, CA: O'Reilly Media.

Zou, D., Long, P. M., & Gu, Q. (2020). On the global convergence of training deep linear ResNets. In *Proceedings of the International Conference on Learning Representations*.

Received July 16, 2021; accepted November 24, 2021.