Improving Adversarial Robustness of Deep Equilibrium Models with Explicit Regulations Along the Neural Dynamics

Zonghan Yang [1] **Peng Li** [2 3] **Tianyu Pang** [4] **Yang Liu** [1 2 3]

Abstract

Deep equilibrium (DEQ) models replace the multiple-layer stacking of conventional deep networks with a fixed-point iteration of a single-layer transformation. Having been demonstrated to be competitive in a variety of real-world scenarios, the adversarial robustness of general DEQs becomes increasingly crucial for their reliable deployment. Existing works improve the robustness of general DEQ models with the widely-used adversarial training (AT) framework, but they fail to exploit the structural uniquenesses of DEQ models. To this end, we interpret DEQs through the lens of neural dynamics and find that AT under-regulates intermediate states. Besides, the intermediate states typically provide predictions with a high prediction entropy. Informed by the correlation between the entropy of dynamical systems and their stability properties, we propose reducing prediction entropy by progressively updating inputs along the neural dynamics. During AT, we also utilize random intermediate states to compute the loss function. Our methods regulate the neural dynamics of DEQ models in this manner. Extensive experiments demonstrate that our methods substantially increase the robustness of DEQ models and even outperform the strong deep network baselines.

1. Introduction

Deep equilibrium (DEQ) models (Bai et al., 2019; 2020) are a type of novel neural architecture. Different from traditional deep networks with multiple stacked layers, DEQ models explicitly cast the forward propagation as a fixed-point iteration process with a single-layer transformation:

$$\mathbf{z}^\star = f_\theta(\mathbf{z}^\star; \mathbf{x}), \tag{1}$$

where f_θ is the transformation parameterized with θ, $\mathbf{x}$ is the input, and $\mathbf{z}^\star$ is the equilibrium solved by fixed-point solvers. While taking $\mathrm{O}(1)$ memory cost because of the single layer, DEQ models are validated to attain competitive performance compared with state-of-the-art traditional deep networks in different applications (Li et al., 2021; Lu et al., 2021; Huang et al., 2021; Bai et al., 2022; Pokle et al., 2022).

Promising in real-world practice, DEQ models necessitate adversarial robustness for their reliable deployment, which however remains underexplored. Most existing works that study robust DEQ models are dedicated to certifying the robustness (Revay et al., 2020; Jafarpour et al., 2021; Müller et al., 2021; Pabbaraju et al., 2021; Chen et al., 2021; Wei & Kolter, 2022) of monotone DEQ. Monotone DEQ (Winston & Kolter, 2020) is a type of DEQ model that enjoys equilibrium convergence guarantees but requires sophisticated layer and weight parameterization. In addition, the scalabil-

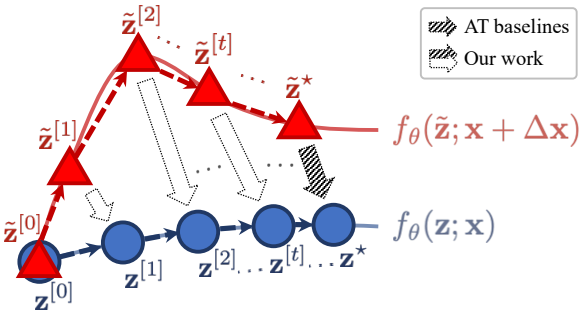

Figure 1. A *conceptual* illustration of the neural dynamics in DEQ models and their regulations. Given input $\mathbf{x}$, the neural dynamics are composed of intermediate states $\{\mathbf{z}^{[t]}\}$ at each forward fixed-point iteration in DEQ models. With input perturbation $\Delta\mathbf{x}$, the neural dynamics are shifted to $\{\tilde{\mathbf{z}}^{[t]}\}$ accordingly. Yang et al. (2022) train robust general DEQs with adversarial training (AT), which imposes regulation only on the equilibrium state by enforcing $\tilde{\mathbf{z}}^\star$ to give similar predictions as $\mathbf{z}^\star$ (the shaded arrow), leaving other intermediate states (*e.g.*, $\tilde{\mathbf{z}}^{[t]}$) along the neural dynamics under-regulated (the hollow arrows). In this work, we propose to impose explicit regulations along the entire DEQ neural dynamics.

[1]Dept. of Comp. Sci. & Tech., Institute for AI, Tsinghua University, Beijing, China [2]Institute for AI Industry Research (AIR), Tsinghua University, Beijing, China [3]Shanghai Artificial Intelligence Laboratory, Shanghai, China [4]Sea AI Lab, Singapore. Correspondence to: Peng Li <lipeng@air.tsinghua.edu.cn>, Yang Liu <liuyang2011@tsinghua.edu.cn>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

ity of robustness certification methods also limits the scope of these prior arts for practical use. In contrast, Gurumurthy et al. (2021) focus on general DEQs and study their empirical adversarial robustness by accelerating the gradient-based attacks. Yang et al. (2022) propose white-box robustness evaluation protocols for general DEQs and conduct fair comparisons between DEQs and traditional deep networks under the adversarial training (AT) framework (Madry et al., 2018). As reported in (Yang et al., 2022), however, the white-box robustness performance of general DEQs still falls behind their deep network counterparts. As adversarial training is a general technique that can be applied to all kinds of differentiable neural architectures, we ask the following question: *Is it possible to exploit the structural uniquenesses of DEQ models to further improve their adversarial robustness?*

Fortunately, the *neural dynamics* perspective for DEQ models brings insights into the problem. The neural dynamics perspective interprets the evolution of intermediate states in a neural model as a dynamical system (E, 2017; Li et al., 2017). This perspective is naturally suitable for DEQs, as their structure in Eq. (1) explicitly formulates the neural dynamics. From this perspective, robust neural models correspond to neural dynamics without a drastic shift in the terminal state given a perturbed initial state (Yan et al., 2019; Kang et al., 2021b), and AT enforces the *terminal state* of neural dynamics to give similar predictions whether the input is clean or perturbed (Zhang et al., 2019b). Shown in Figure 1, for DEQ models, AT does not *explicitly* regulate *intermediate states* along neural dynamics. However, Yang et al. (2022) shows that even a DEQ model is trained by AT, its intermediate states can still be attacked, leading to poor robustness performance. This finding implies the structural specialty of DEQs differentiating from deep networks, and paves the way for explicit regulations along the neural dynamics to improve their adversarial robustness.

In this work, we exploit the structural properties of DEQs to explicitly regulate their neural dynamics for improved robustness. Drawing inspiration from the entropy in dynamical systems and its implications on system stability and robustness, we propose to reduce prediction entropy by progressively updating the inputs along the DEQ neural dynamics. We also randomly select intermediate states along the neural dynamics for loss computation in adversarial training. In this way, our methods integrate explicit regulations along the neural dynamics of DEQ models, and boost the robustness of general DEQs: On the standard white-box robustness evaluation benchmark CIFAR-10 with perturbation range $\ell_\infty = 8/255$, our DEQs achieve significantly better performance in white-box adversarial robustness compared with the results in Yang et al. (2022), and even outperform the strong deep network baselines with benchmarked adversarial robustness results in Pang et al. (2021). We have also conducted several ablation studies to validate the effectiveness of our proposed methods. Our code is available at `https://github.com/minicheshire/DEQ-Regulating-Neural-Dynamics`.

2. Preliminaries

Deep equilibrium models are a class of emerging neural architecture (Bai et al., 2019; 2020). Of all deep networks, the closest resemblance to a DEQ model is an M-layer deep network with weight sharing and input injection. The forward propagation process of such a deep network would be

$$\mathbf{z}^{[m+1]} = f_\theta(\mathbf{z}^{[m]}; \mathbf{x}),\ \mathbf{z}^{[0]} = \mathbf{0}, \tag{2}$$

where $\mathbf{x} \in \mathbb{R}^l$ is the input, and $\mathbf{z}^{[m]} \in \mathbb{R}^d$ is the intermediate state after the m-th layer with $m = 0, \cdots, M-1$. $f_\theta : \mathbb{R}^{d\times l} \to \mathbb{R}^d$ forms the transformation at each layer, and θ is the weight shared across different layers of the deep network. When implementing this network in an automatic differentiation engine (*e.g.*, PyTorch (Paszke et al., 2019)), the $f_\theta(\cdot; \mathbf{x})$ transformation is sequentially compounded for M times, and all the intermediate states $\mathbf{z}^{[1]} \sim \mathbf{z}^{[M]}$ need to be stored. DEQ models seek the limit of Eq. (2) when the number of layers goes infinity: Assuming the convergence of the process, as $m \to \infty$, the state $\mathbf{z}^{[m]}$ converges to the equilibrium $\mathbf{z}^\star$ with $\mathbf{z}^\star = f_\theta(\mathbf{z}^\star; \mathbf{x})$, as stated in Eq. (1).

DEQ models cast the "infinite" forward process of Eq. (2) as a fixed-point iteration process to solve for the equilibrium $\mathbf{z}^\star$ in Eq. (1). While the most straightforward way to do this is exactly Eq. (2), in DEQs, advanced fixed-point solvers (*e.g.*, Broyden's method (Broyden, 1965)) are used to accelerate the iteration convergence. For simplicity, we abusively refer to $\mathbf{z}$'s as the intermediate states of DEQs in the following and discard Eq. (2). After N iterations of the forward solver, the $\mathbf{z}^{[N]}$ is returned and is numerically treated as $\mathbf{z}^\star = \mathbf{z}^{[N]}$.

Now we provide the formal definition for the neural dynamics of DEQ models, which are at the heart of our study. Neural dynamics reflect the evolution of the intermediate states in a neural model. For DEQ models, the neural dynamics consist of the sequence $\{\mathbf{z}^{[1]}, \cdots, \mathbf{z}^{[N]}\}$, which satisfies

$$\mathbf{z}^{[t+1]} = \mathrm{Solve}\Big(\mathbf{z} = f_\theta(\mathbf{z}; \mathbf{x});\ \mathbf{z}^{[\le t]}\Big) \tag{3}$$

for $0 \le t < N$ and $\mathbf{z}^{[0]} = \mathbf{0}$. Solve is the fixed-point solver for the forward process in DEQs, which is usually instantiated with the Broyden's method (Broyden, 1965). At iteration t, the solver Solve bases on $\mathbf{z}^{[t]}$ to compute the next intermediate state $\mathbf{z}^{[t+1]}$ for the fixed-point equation. While $\{\mathbf{z}^{[t]}\}$ are not stored in memory, Yang et al. (2022) demonstrate that the intermediate states $\mathbf{z}^{[t]} \neq \mathbf{z}^\star$ exhibit higher robustness than the equilibrium state $\mathbf{z}^\star$, and attacks can be constructed for the intermediate $\mathbf{z}^{[t]}$s. In our work, we explicitly regulate the behavior of all $\{\mathbf{z}^{[t]}\}$ along the neural dynamics in DEQ models to improve their robustness.

3. Methodology

In this section, we demonstrate our approaches that facilitate explicit regulations along the neural dynamics of DEQs to improve robustness. We start with a short overview with two observations about the structural properties of DEQs in Sec. 3.1. We then exploit the two uniquenesses of DEQs and propose two regulation methods in Secs. 3.2 and 3.3.

3.1. Overview

Suppose a trained DEQ model on an image classification task. Its weight θ is fixed, and the forward iteration number N is constant. From Eq. (3), it can be seen that the neural dynamics are fully decided by the input $\mathbf{x}$. How does the perturbation to $\mathbf{x}$ affect the neural dynamics in the DEQ model?

Assume that a clean input $\mathbf{x}$ induces $\{\mathbf{z}^{[t]}\}$, and a perturbed input $\mathbf{x}+\Delta\mathbf{x}$ induces $\{\tilde{\mathbf{z}}^{[t]}\}$. To get an intuitive understanding of the difference between them, we replace the Solve in Eq. (3) with the most straightforward unrolling for all $t = 1, \cdots, N$. For intermediate step t, we have

$$\mathbf{z}^{[t+1]} = f_\theta(\mathbf{z}^{[t]}; \mathbf{x}), \ \tilde{\mathbf{z}}^{[t+1]} = f_\theta(\tilde{\mathbf{z}}^{[t]}; \mathbf{x}+\Delta\mathbf{x}). \quad (4)$$

The difference between $\tilde{\mathbf{z}}^{[t+1]}$ and $\mathbf{z}^{[t+1]}$ reads

$$\begin{aligned} &\|\tilde{\mathbf{z}}^{[t+1]} - \mathbf{z}^{[t+1]}\| = \|f_\theta(\tilde{\mathbf{z}}^{[t]}; \mathbf{x}+\Delta\mathbf{x}) - f_\theta(\mathbf{z}^{[t]}; \mathbf{x})\| \\ =&\|f_\theta(\tilde{\mathbf{z}}^{[t]}; \mathbf{x}+\Delta\mathbf{x}) - f_\theta(\tilde{\mathbf{z}}^{[t]}; \mathbf{x}) \ + \ f_\theta(\tilde{\mathbf{z}}^{[t]}; \mathbf{x}) - f_\theta(\mathbf{z}^{[t]}; \mathbf{x})\| \\ \leq&\underbrace{\|f_\theta(\tilde{\mathbf{z}}^{[t]}; \mathbf{x}+\Delta\mathbf{x}) - f_\theta(\tilde{\mathbf{z}}^{[t]}; \mathbf{x})\|}_{\text{Perturbation from } \mathbf{x}} + \underbrace{\|f_\theta(\tilde{\mathbf{z}}^{[t]}; \mathbf{x}) - f_\theta(\mathbf{z}^{[t]}; \mathbf{x})\|}_{\text{Accumulation in } \mathbf{z}}. \end{aligned} \quad (5)$$

According to Eq. (5), the difference between $\tilde{\mathbf{z}}^{[t+1]}$ and $\mathbf{z}^{[t+1]}$ is inherited from those between $\tilde{\mathbf{z}}^{[t]}$ and $\mathbf{z}^{[t]}$, and further amplified by the perturbed input $\mathbf{x}+\Delta\mathbf{x}$. Fortunately, DEQs differentiate from traditional deep networks (He et al., 2016; Zagoruyko & Komodakis, 2016) in two structural uniquenesses: (i) The input $\mathbf{x}$ is involved in each iteration along the neural dynamics of DEQs. In contrast, conventional deep residual networks do not follow a layer-wise input-injection design. (ii) All of the intermediate states along the neural dynamics can be seamlessly sent into the classification head of the DEQ model for predictions. By comparison, for traditional deep networks like ResNets, the intermediate representations are often of different shapes from the input of the top classification layer. According to the two properties in DEQ models, we propose two methods for neural dynamics regulation in Secs. 3.2 and 3.3.

3.2. Input Entropy Reduction Along Neural Dynamics

3.2.1. Observations of Prediction Entropy

As noted in Sec. 3.1, the input $\mathbf{x}$ is applied along the neural dynamics in DEQ models, and $\mathbf{x}$ can be either clean or perturbed. A well-trained DEQ model exhibits neural dynamics that always obey a converged fixed-point iteration process with any inputs. Shown in Fig. 2-(a), for a DEQ model trained with AT, the relative error at the equilibrium state $\|f_\theta(\mathbf{z}^{[N]}; \mathbf{x}) - \mathbf{z}^{[N]}\|_2 / \|f_\theta(\mathbf{z}^{[N]}; \mathbf{x})\|_2$ follows similar distributions with either clean or perturbed input.

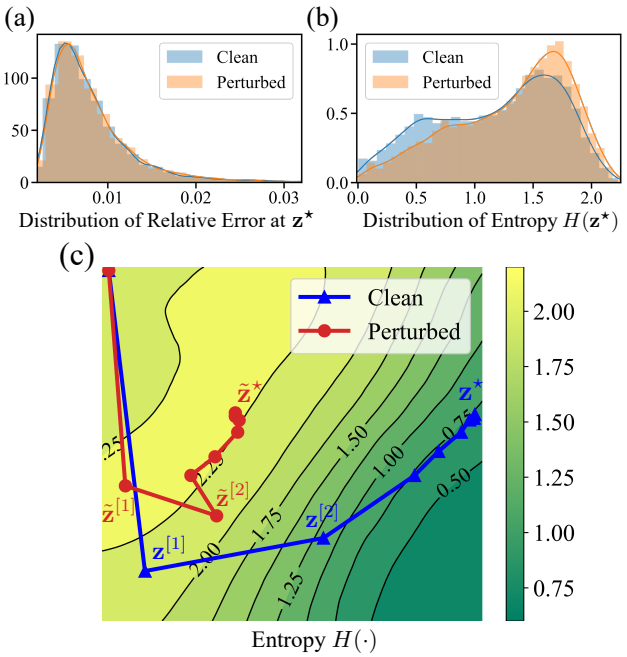

Figure 2. Observations on the prediction entropy of a DEQ model trained with AT under clean and perturbed inputs. **(a)** The distribution of relative error at the equilibrium state $\mathbf{z}^\star$ in the forward solver. Both clean and perturbed inputs lead to converged fixed-point neural dynamics with similar relative error distribution over the validation set. **(b)** The distribution of prediction entropy at the equilibrium state. Compared with (a), the clean inputs tend to induce equilibrium states that give smaller prediction entropy than the perturbed input counterparts. **(c)** An exampled visualization of the prediction entropy of each intermediate state along the neural dynamics. Compared to the clean input with $\mathbf{z}^{[t]}$s of diminishing prediction entropy and correct predictions, the perturbed input mistakes the prediction and results in $\tilde{\mathbf{z}}^{[t]}$s of high prediction entropy.

An adversarial example $\mathbf{x}+\Delta\mathbf{x}$ leads to the incorrect prediction of $\tilde{\mathbf{z}}^{[N]}$. However, the initial state $\mathbf{z}^{[0]}$ in DEQ models is always set to $\mathbf{0}$. This implies that with a perturbed $\mathbf{x}+\Delta\mathbf{x}$, the neural dynamics $\{\tilde{\mathbf{z}}^{[t]}\}$ diverge from the $\{\mathbf{z}^{[t]}\}$ under the clean $\mathbf{x}$ in predictions. While the DEQ model is determined in the fixed-point convergence of the neural dynamics, we investigate whether the intermediate states along the neural dynamics are "determined" in their predictions.

To characterize the "determination" of predictions, we adopt entropy as the measurement. On the one hand, a higher prediction entropy indicates a more flat probability distribution, with smaller probability differences among different classes. On the other hand, the theory of entropy in dynamical systems (Young, 2003) shows that a dynamical system with

higher entropy indicates larger Lyapunov exponents, therefore more inclined to be unstable (see a brief discussion about this in Appendix B). Inspired by this, we compute the prediction entropy of an intermediate state $\mathbf{z}^{[t]}$. Formally, the prediction entropy of $\mathbf{z}^{[t]}$ is defined as

$$H(\mathbf{z}^{[t]}) = -\sum_{j=1}^{C} p_j^{[t]} \log p_j^{[t]}, \tag{6}$$

where $\mathbf{p}^{[t]} = h_\Phi\big(\mathbf{z}^{[t]}\big)$ is the prediction logits vector with $\mathbf{p}^{[t]} = \left[p_j^{[t]}\right] \in \mathbb{R}^C$. $h_\Phi : \mathbb{R}^d \to \mathbb{R}^C$ is the classification head in the DEQ model with parameters Φ and C classes.

We use Eq. (6) to investigate the prediction entropy of the equilibrium state in DEQ models with clean or perturbed inputs. Surprisingly, shown in Fig. 2-(b), we find that the perturbed inputs result in the equilibrium states with higher prediction entropy than the clean inputs from a distributional perspective. It is inferred that the neural dynamics with high prediction entropy are prone to give incorrect predictions.

We further visualize an example of the prediction entropy along all the intermediate states of the neural dynamics in Fig. 2-(c), with clean $\mathbf{x}$ or perturbed $\mathbf{x} + \Delta\mathbf{x}$ as the input. With the clean input $\mathbf{x}$, the prediction entropy diminishes along the corresponding neural dynamics. In contrast, with the perturbed input $\mathbf{x} + \Delta\mathbf{x}$, the prediction entropy for each intermediate state remains high, and the resulting prediction is mistaken. To guide the perturbed dynamics toward the clean one, we propose to *reduce the prediction entropy by progressively updating the input along the neural dynamics*. In the next section, we provide its optimization framework.

3.2.2. Input Entropy Reduction Framework

Given a potentially perturbed input $\mathbf{x}$, the entropy reduction framework with progressive input updates is formalized as

$$\begin{aligned} \min_{\mathbf{u}^{[1]},\cdots,\mathbf{u}^{[N]}} \quad & H(\mathbf{z}^{[N]}), \\ \text{s.t.} \quad & \mathbf{z}^{[t+1]} = \text{Solve}\Big(\mathbf{z} = f_\theta(\mathbf{z}; \mathbf{x} + \mathbf{u}^{[t]});\ \mathbf{z}^{[\le t]}\Big), \\ & \mathbf{u}^{[t]} \in [-\epsilon, \epsilon]^l, \end{aligned} \tag{7}$$

where $t = 1, \cdots, N$, and $\{\mathbf{u}^{[t]}\}$ are the updates on the input $\mathbf{x}$ with range constraints. From (7), the $\{\mathbf{u}^{[t]}\}$ can be viewed as the controllers along the neural dynamics. Framework (7) then forms an optimal control problem with the aim of guiding the neural dynamics towards reduced entropy at the final state $\mathbf{z}^{[N]}$. As demonstrated in Li et al. (2017); Zhang et al. (2019a), by solving the problem with Pontryagin Maximum Principle (PMP) (Pontryagin et al., 1962), the gradient descent methods are derived to obtain the optimal controllers $\{\mathbf{u}^{[t]}\}$. We employ the iterative projected gradient descent framework to optimize for $\mathbf{u}^{[t]}$. Specifically, at iteration t, after $\mathbf{z}^{[t+1]}$ in the neural dynamics is obtained by Eq. (3) given $\mathbf{z}^{[\le t]}$ and $\mathbf{x}^{[t]}$, the input $\mathbf{x}^{[t]}$ is updated to

Algorithm 1 Entropy Reduction in Sec. 3.2

Input: Input $\mathbf{x}$, iteration number R and frequency T_f
Output: The equilibrium state $\mathbf{z}^\star = \mathbf{z}^{[N]}$

```
/* Test Phase */
z^[0] = 0, x^[0] = x
for t = 0, ··· , N − 1 do
    z^[t+1] = Solve(z = f_θ(z; x^[t]); z^[≤t])
    x^[t+1] = x^[t]
    if (t + 1) mod T_f = 0 then
        /* Reducing x^[t] Prediction Entropy for R Iters */
        x_0^[t] = x^[t]
        for i = 1, ··· , R do
            Update x_{i−1}^[t] into x_i^[t] with Eq. (8)
        end for
        x^[t+1] = x_R^[t]
        z^[t+1] = Solve(z = f_θ(z; x^[t+1]); z^[≤t])
    end if
end for
```

reduce prediction entropy at $\mathbf{z}^{[t+1]}$ for R iterations:

$$\mathbf{x}_i^{[t]} = \text{Proj}_{[-\epsilon,\epsilon]^l}\left(\mathbf{x}_{i-1}^{[t]} - \beta \nabla_{\mathbf{x}} H(f_\theta(\mathbf{z}^{[t+1]}; \mathbf{x}_{i-1}^{[t]}))\right), \tag{8}$$

with β as the step size, $i = 1, \cdots, R$, $\mathbf{x}_0^{[t]} = \mathbf{x}^{[t]}$, and $\mathbf{x}^{[t+1]} = \mathbf{x}_R^{[t]}$ forms $\mathbf{x} + \mathbf{u}^{[t+1]}$. After the updates of the input, the state $\mathbf{z}^{[t+1]}$ is re-calculated with $\mathbf{x}^{[t+1]}$ in Eq. (3). Solving for $\mathbf{u}^{[t]}$ for each t can be time-consuming. In the implementation, we intervene to optimize for $\mathbf{u}^{[t]}$ every T_f states along the neural dynamics, *i.e.*, requiring $\mathbf{u}^{[pT_f+q]} = \mathbf{u}^{[pT_f+1]}, \forall p, q \in \mathbb{N},\ p \ge 0,\ 1 < q \le T_f$.

The whole process of the input entropy reduction is shown in Algo. 1. In this way, the original neural dynamics are mounted to a regulated "orbit" from $\mathbf{z}^{[t]}$, which would crucially impact the states afterward and the predictions they give, and eventually result in reduced $H(\mathbf{z}^{[N]})$.

Our method is also related to the joint optimization for inputs and states in DEQ models (Gurumurthy et al., 2021). However, in this work, we do not couple the update of $\mathbf{z}^{[t+1]}$ with Eq. (8) and refrain from the calculation of the joint Jacobian for $\mathbf{x}$ and $\mathbf{z}$. We leave more efficient and effective regulations for neural dynamics in DEQs as future work.

3.3. Loss from Random Intermediate States

In addition to progressively updating the input during testing, we propose another technique for the explicit regulation of the neural dynamics in DEQ models. As shown in Eq. (5), the second term reflects the difference accumulated in $\tilde{\mathbf{z}}^{[t]}$ from $\mathbf{z}^{[t]}$. A straightforward approach to imposing explicit regulations on the intermediate state $\tilde{\mathbf{z}}^{[t]}$ is to calculate the adversarial loss using *random intermediate states* during AT.

Formally, for the vanilla AT baselines, the loss function L in

Table 1. Comparisons among the robustness performance (%) of deep networks (RESNET-18) (Pang et al., 2021) and the DEQ models of similar parameter counts (DEQ-LARGE) with our methods in different AT frameworks on CIFAR-10 test set. We leverage the early-state defense proposed in Yang et al. (2022) for all DEQ models, and use "ALL" to represent the minimum of the PGD attack and AutoAttack ("AA"). Our methods significantly improve the robustness of DEQs over (Yang et al., 2022), and even outperform the strong deep network baselines with the TRADES framework. ‡ indicates our implementation; * denotes that the results are brought from Pang et al. (2022).

AT FRAMEWORK	ARCHITECTURE	METHOD	CLEAN	PGD	AA	ALL
PGD-AT	RESNET-18	PANG ET AL. (2021)	**82.52**	53.58	48.51	48.51
	DEQ-LARGE	YANG ET AL. (2022)	79.67	47.12	48.37	47.12
		YANG ET AL. (2022)‡	77.89	49.45	47.58	47.58
		+ OURS (SEC. 3.2)	77.51	**51.62**	**49.31**	**49.31**
		+ OURS (SEC. 3.3)	78.93	48.18	48.09	48.09
		+ OURS (SECS. 3.2 & 3.3)	80.63	49.22	43.79	43.79
TRADES	RESNET-18	PANG ET AL. (2021)*	**81.47**	-	49.14	49.14
	DEQ-LARGE	YANG ET AL. (2022)‡	74.92	50.46	50.33	50.33
		+ OURS (SEC. 3.2)	73.80	51.41	50.52	50.52
		+ OURS (SEC. 3.3)	77.64	51.10	49.64	49.64
		+ OURS (SECS. 3.2 & 3.3)	78.89	**55.18**	**51.50**	**51.50**

the objective is calculated using only the equilibrium state:

$$\min_{\theta,\Phi} \max_{\Delta\mathbf{x}\in[-\epsilon,\epsilon]^l} L\left(h_\Phi\left(\tilde{\mathbf{z}}^{[N]}\right), y\right), \tag{9}$$

with the equilibrium state $\tilde{\mathbf{z}}^{[N]}$ satisfying Eq. (3) with $\mathbf{x}+\Delta\mathbf{x}$ as the input, and y is the ground-truth label for $\mathbf{x}$. Our method calculates Eq. (9) with random intermediates:

$$\min_{\theta,\Phi} \max_{\Delta\mathbf{x}\in[-\epsilon,\epsilon]^l} \mathbb{E}_{i\in\mathcal{U}[1,N]}\, L\left(h_\Phi\left(\tilde{\mathbf{z}}^{[i]}\right), y\right), \tag{10}$$

where we randomly select intermediate states $\tilde{\mathbf{z}}^{[i]}$ inside the forward fixed-point solver for loss computation. In this way, all the intermediates are imposed with explicit regulations without violating the $O(1)$ memory constraint of DEQ models. We thus expect their neural dynamics to be less deviated under attacks and exhibit higher robustness.

4. Experiments

Setup. We follow the settings in Yang et al. (2022) of the configurations of DEQ model architecture: the large-sized DEQ with its parameter count similar to ResNet-18. The number of iterations N in the forward solver is 8. For adversarial training frameworks, we use both PGD-AT (Madry et al., 2018) and TRADES (Zhang et al., 2019b). PGD-AT is used in the previous study on robust DEQ models (Madry et al., 2018), while the regularization term for robustness in TRADES shares similarity with Eq. (5). We experiment on CIFAR-10 (Krizhevsky & Hinton, 2009) with ℓ_∞ perturbation range $\epsilon = 8/255$. The default hyperparameter setting for the Sec. 3.2 method is $\beta = 2/255$, $R = 10$, and $T_f = 2$. The detailed settings are listed in Appendix A.

We follow to use the white-box robustness evaluation protocol proposed in Yang et al. (2022): We use the early-state defense by selecting the intermediate state with the highest accuracy under ready-made PGD-10 to compute for predictions. As the intermediate state is non-differentiable, we adopt the proposed intermediate unrolling method that estimates the gradients used to attack the state. Specifically, the gradients used in the attacks are calculated by unrolling an intermediate state $\mathbf{z}^{[i]}$ for K_a steps:

$$\mathbf{z}_a^{[i+j]} = (1-\lambda)\mathbf{z}_a^{[i+j-1]} + \lambda f_\theta(\mathbf{z}_a^{[i+j-1]}; \mathbf{x}), \tag{11}$$

with $\mathbf{z}_a^{[i]} = \mathbf{z}^{[i]}$ and $j = 1, \cdots, K_a$, and $\mathbf{z}_a^{[i+K_a]}$ is used to compute the loss and take the gradient. In our work, we provide a systematic evaluation by covering all $1 \le i \le N = 8$, $1 \le K_a \le 9$, and $\lambda \in \{0.5, 1\}$. Unless specified, all of the single robustness performance that is reported "under intermediate attacks" is the minimum accuracy over $8 \times 9 \times 2 = 144$ attacks in the form of Eq. (11).

4.1. Main Results

Table 1 shows the robustness comparisons among traditional deep networks ResNet-18 and the DEQ models with a similar amount of parameters (DEQ-Large). For the DEQ models, we use the original adversarial training framework and compose it with our methods. According to the results, Both the test-time input entropy reduction in Sec. 3.2 and the training-time loss computation with random intermediates in Sec. 3.3 improve DEQ model robustness over the vanilla AT baselines. Using our methods, the robustness performance of the DEQ-Large models significantly out-performs the DEQ-Large baselines in Yang et al. (2022), and even surpasses that of ResNet-18 (Pang et al., 2021).

Intermediate attacks are strong. As we conduct comprehensive intermediate-state attack experiments, the validated

Table 2. Comparisons between the ready-made PGD-10 at the equilibrium state $\mathbf{z}^{[N]}$ and the intermediate attack at state $\mathbf{z}^{[i]}$ (Eq. (11)). The intermediate attacks are always more effective, as they result in lower accuracy (%) of the model than the ready-made ones.

AT	METHOD	PGD AT WHICH $\mathbf{z}^{[t]}$ FINAL	INTER.	DIFF.
PGD-AT	YANG ET AL. (2022)	50.55	49.45	1.10
	+ SEC. 3.2	53.01	**51.62**	1.39
	+ SEC. 3.3	51.05	48.18	2.87
	+ SECS. 3.2 & 3.3	54.91	49.22	5.69
TRADES	YANG ET AL. (2022)	51.92	50.46	1.46
	+ SEC. 3.2	53.74	51.41	2.33
	+ SEC. 3.3	52.67	51.10	1.57
	+ SECS. 3.2 & 3.3	56.09	**55.18**	0.91

robustness is more reliable than only using ready-made attacks. Table 2 demonstrates that the intermediate-state PGD attacks always result in a larger decrease of white-box adversarial robustness than off-the-shelf attacks at the final state. According to Table 1, the effect of AutoAttack (Croce & Hein, 2020) is usually stronger than the PGD-10 attacks in TRADES experiments. This is opposite to the performance reported in Yang et al. (2022), as they argue that the AutoAttack will overfit to the inaccurate gradient estimations by unrolled intermediates and result in attacks weaker than PGD-10. However, in our work, we circumvent the overly inaccurate gradient estimations by scrutinizing all possible pairs of (i, K_a) in Eq. (11), leading to stronger adaptive-size PGD in AutoAttack. The composition of our methods yields the top robustness in the DEQ models trained with TRADES. The possible reason is that the regularization term in TRADES is more suitable for the regulations on Eq. (5). In the following sections, we conduct further evaluation and analysis with the TRADES-trained DEQ models.

4.2. Robustness Evaluation for Test-Time Defense

Sec. 3.2 describes the algorithm for prediction entropy reduction by iteratively updating the input along the neural dynamics. As the algorithm works at inference time, we follow the guidelines in Croce et al. (2022b) to evaluate its robustness. Based on the intermediate-state Eq. (11) as adaptive attacks for DEQ models, we transfer the attacks across different defense methods of ours ("TRADES Baseline", "TRADES + Sec. 3.2", "TRADES + Sec. 3.3", and "TRADES + Secs. 3.2 and 3.3"), and further employ the adaptive-size APGD-CE and the score-based Square (Andriushchenko et al., 2020) attacks. It is noted that our Sec. 3.2 defense operates on the neural dynamics, which lie along the forward pass of DEQ models only. Therefore, following the comments from Croce et al. (2022b) to Yoon et al. (2021), we do not equip BPDA (Athalye et al., 2018) with the attacks for our defense as it is unnecessary.

Table 3. The performance (%) of each type of our methods under the attacks transferred from other types of our methods with TRADES. We transfer all the intermediate attacks defined by Eq. (11) and report the minimum accuracy. Given one of our methods, it is shown that the attacks transferred from other defenses are always weaker than the attacks designed for the given method itself. The lowest robustness for each defense is shaded.

METHOD	PGD ATTACKS TRANSFERRED FROM BASE	B+S3.2	B+S3.3	B+S3.2&3.3
BASE	**50.46**	50.73	56.66	57.37
+ S3.2	51.58	**51.41**	55.60	56.09
+ S3.3	60.00	60.24	**51.10**	52.89
+ S3.2&3.3	59.87	60.06	55.19	**55.18**

Table 4. Robustness performance (%) of our methods with Secs. 3.2, 3.3 and TRADES under strong adaptive attacks. We choose the top (i, K_a) pairs in Eq. (11) that lead to significant accuracy decreases with PGD, and equip them with APGD-CE (Croce & Hein, 2020) and SQUARE (Andriushchenko et al., 2020). The results in Table 1 are reported with $(i, K_a) = (3, 5)$, as it forms the strongest attacks. * denotes evaluation with 1,000 test samples.

ATTACK	(i, K_a) IN EQ. (11) (3,5)	(3,4)	(3,6)	(3,7)	(4,5)	(5,5)	(6,5)
PGD	**55.18**	55.36	55.24	55.35	55.24	55.53	55.59
APGD	**53.29**	56.75	56.95	57.07	53.38	53.92	53.91
SQUARE*	67.00	67.23	67.28	67.13	**66.65**	67.00	66.75

Attack transferability. We transfer all the intermediate-state attacks among the four different defense methods based on TRADES to one another. The minimum robust accuracy for each setting is reported in Table 3. It is shown that for each defense method, its strongest attack is still constructed against the method itself. For a robustly-trained DEQ model (either by the TRADES framework or plus the Sec. 3.3 method), the adversarial examples have similar effects whether the Sec. 3.2 method is used or not.

Exploiting APGD-CE and Square attacks. We select the (i, K_a) pairs in Eq. (11), which form the intermediate-state attacks that trigger severe drops in accuracy with PGD-10. Specifically, $(i, K_a) = (3, 5)$ forms the strongest attacks, and the results in Table 1 are reported under this setting as well. We then implement these (i, K_a) settings in APGD-CE and Square attacks. Table 4 shows the effect of these attacks on our strongest defense "TRADES + Secs. 3.2 & 3.3". For Square attacks, due to the time limit, we evaluate the performance on 1,000 test samples. According to the results, the adaptive-size APGD-CE is stronger than PGD, and the defense method retains higher accuracy under Square attacks than PGD and APGD-CE attacks. These phenomena agree with the performance of deep networks (Croce et al., 2021). Finally, the robustness performances among different settings are similar, indicating the robustness of our defense to the configurations in Eq. (11) as well.

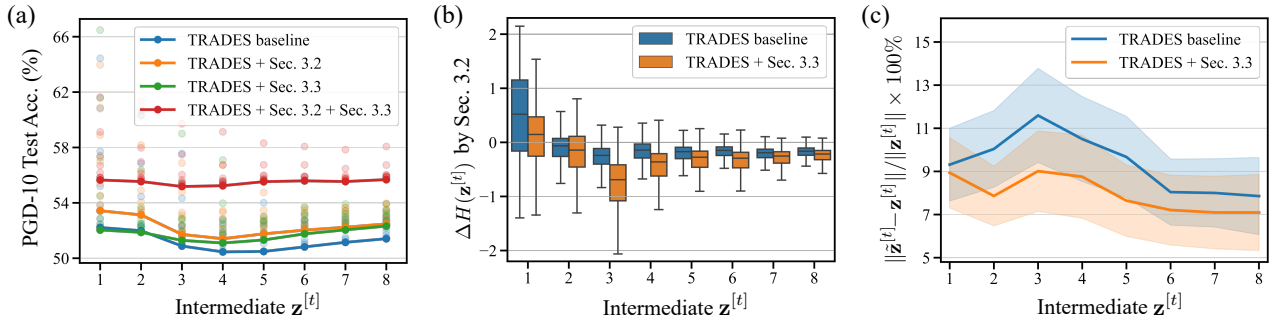

Figure 3. Comparisons among the original TRADES baseline and our methods along the neural dynamics $\mathbf{z}^{[t]}$ produced by the forward solver in DEQ models. **(a)** PGD-10 attacks with varied unrolling steps at different intermediate states along the neural dynamics. At each $\mathbf{z}^{[t]}$, the robustness under the strongest attack is plotted by a solid dot, with others by transparent dots. It is shown that the overall strongest attack is constructed by the state around the middle of the neural dynamics ($\mathbf{z}^{[3]}$ - $\mathbf{z}^{[5]}$). The composition of Secs. 3.2 and 3.3 forms the most robust DEQ model. **(b)** The effect of entropy reduction by Sec. 3.2. For DEQ models trained with both the TRADES baseline and the loss calculated by random intermediates (Sec. 3.3), the Sec. 3.2 method always manages to reduce the entropy along the neural dynamics, with the largest decrease in $H(\mathbf{z}^{[t]})$ at $\mathbf{z}^{[3]}$. The entropy reduction effect is more significant in the DEQ model trained with "TRADES + Sec. 3.3", which also accounts for the corresponding robustness improvement in Table 1. **(c)** The relative difference of $\tilde{\mathbf{z}}^{[t]}$ (with perturbed inputs) from $\mathbf{z}^{[t]}$ (with clean inputs). When trained with the method in Sec. 3.3, the DEQ model shows less-deviated neural dynamics under input perturbations than the "TRADES baseline", thus also demonstrating better robustness in Table 1.

5. Analysis and Discussion

5.1. Robustness Improvement Along Neural Dynamics

In this section, we conduct an in-depth analysis of the robustness improvement from our methods with TRADES by investigating the neural dynamics of the DEQ models. We use PGD-10 as it also reliably reflects model robustness in Table 1, while being faster than the adaptive-size attacks.

We first plot the PGD-10 robust accuracy for all the attacks in Eq. (11) at different intermediate states $\mathbf{z}^{[t]}$ with various unrolling steps K_a and $\lambda \in \{0.5, 1.0\}$. In Fig. 3-(a), we plot the lowest robust accuracy of the model under the attack constructed by unrolling $\mathbf{z}^{[t]}$ as a solid dot. The accuracy results under other attacks with $\mathbf{z}^{[t]}$ unrolling are depicted in transparency. Along the neural dynamics, it is observed that the strongest attack lies around the middle, namely, by unrolling the intermediate state of $\mathbf{z}^{[3]}$ - $\mathbf{z}^{[5]}$. The overall robustness performance of the model is determined by the lowest accuracy among the solid dots. Our composed method of Secs. 3.2 and 3.3 forms the strongest defense, as the lowest accuracy it obtains is much higher than all the other methods. Opposite to Yang et al. (2022), we find that $\lambda = 0.5$ in Eq. (11) results in attacks always stronger than $\lambda = 1.0$ (see detailed comparisons in Appendix C.1).

To validate the effectiveness of the Sec. 3.2 method, we quantize the entropy reduction $\Delta H(\mathbf{z}^{[t]})$ given each input from the validation set perturbed by the strongest intermediate PGD attacks. The distribution of $\Delta H(\mathbf{z}^{[t]})$ for each t is illustrated in Fig. 3-(b). As Sec. 3.2 updates the input along the neural dynamics, each $\mathbf{z}^{[t]}$ is correspondingly modified, thus leading to the difference in terms of its prediction entropy. According to Fig. 3-(b), $\Delta H(\mathbf{z}^{[t]}) < 0$ for each $t > 1$. This means the entropy at each $\mathbf{z}^{[t]}$ is reduced for $t > 1$, which verifies the effectiveness of our method. Specifically, the largest deterioration of $\Delta H(\mathbf{z}^{[t]})$ happens at $\mathbf{z}^{[3]}$, which is also around the middle of the neural dynamics.

Table 5. Robustness performance (%) of the test-time method in Sec. 3.2 with different settings of the frequency T_f and the iteration number R, with the underlying DEQ model trained with TRADES + the Sec. 3.3 method. For each pair of T_f and R, we evaluate the robustness accuracies using all PGD attacks along the neural dynamics of the DEQ model and report the minimum among them.

	$R = 1$	$R = 5$	$R = 10$	$R = 20$
$T_f = 1$	54.67	54.96	55.07	55.03
$T_f = 2$	54.15	55.05	55.18	**55.23**
$T_f = 4$	53.39	54.80	54.85	54.85
$T_f = 8$	52.85	52.80	52.83	52.81

Finally, we demonstrate the effect of Sec. 3.3 by plotting the relative difference of intermediate states ($\mathbf{z}^{[t]}$ and $\tilde{\mathbf{z}}^{[t]}$) along the neural dynamics given a clean input $\mathbf{x}$ and the perturbed one $\mathbf{x} + \Delta\mathbf{x}$. Shown in Fig. 3-(c), when trained with the loss computed with random intermediates, the DEQ model exhibits neural dynamics with the less relative difference among the clean and the perturbed inputs than the baseline. This analysis also accounts for the superiority of the Sec. 3.3 method in adversarial robustness, as shown in Table 1.

5.2. Effect of T_f and R in Sec. 3.2

In this section, we ablate the effect of T_f and R in the Sec. 3.2 method. During the input entropy reduction process, T_f controls the intervention frequency of input updates along the neural dynamics, and R denotes the iteration number

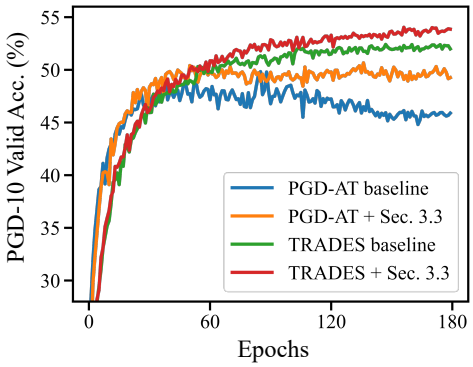

Figure 4. Robustness performance of different methods under ready-made PGD-10 on the CIFAR-10 validation set in DEQ training. The integration of Sec. 3.3 stabilizes the training process and yields consistent robustness improvement over the AT baselines.

within each intervention of input entropy reduction. According to the results in Table 5, a larger T_f leads to relatively lower robustness. This indicates that the Sec. 3.2 method would benefit from frequent input entropy reduction. Similarly, the method becomes more effective as R increases. As the time consumption is proportional to R and inversely proportional to T_f, we set $R = 20$ and $T_f = 2$ in our main experiments (see Appendix D for time cost analysis).

5.3. Effect of Sec. 3.3 During Training

In Fig. 4, we visualize the training process of each AT baseline and its improved version with the Sec. 3.3 method. The robust accuracy reported in Fig. 4 is evaluated at the final state $\mathbf{z}^\star$. It is witnessed that the Sec. 3.3 method always improves over the AT baselines in robust accuracy. The robust accuracy results obtained by TRADES are always higher than those obtained by PGD-AT through the training process. This agrees with the conclusions for deep networks (Zhang et al., 2019b; Croce et al., 2021). It is also observed that the PGD-AT framework leads to faster robustness overfitting, while this effect is not obvious in the TRADES experiments. Finally, it is noted that DEQ models are by default trained with Adam optimizer (Kingma & Ba, 2015), zero weight decay, and learning rate cosine decay (Loshchilov & Hutter, 2017). This differs from the common practice in training robust deep networks, where the optimizer is usually SGD, with weight decay and early stopping after the first time the learning rate is decayed (Zhang et al., 2019b; Rice et al., 2020). While these tricks have proven to be crucial in adversarial training (Pang et al., 2021), we failed to implement similar techniques in DEQ training: For example, our initial experiments show that when setting weight decay to be 5e-4, the loss becomes NaN after about 20 training epochs with PGD-AT. We leave more effective and efficient adversarial training for DEQ models as future work.

Table 6. Statistics of the P values calculated with all of the 144 types of intermediate state attacks according to Eq. (12).

AT	METHOD	AVG	MIN	MAX
PGD-AT	YANG ET AL. (2022)	78.26%	72.98%	85.53%
	+ SEC. 3.3	78.39%	68.79%	83.77%
TRADES	YANG ET AL. (2022)	81.46%	75.88%	87.99%
	+ SEC. 3.3	82.34%	76.63%	88.58%

Table 7. Statistics of the ΔH values calculated with all of the 144 types of intermediate state attacks according to Eq. (13).

AT	METHOD	AVG	MIN	MAX
PGD-AT	YANG ET AL. (2022)	-0.1948	-0.2133	-0.1156
	+ SEC. 3.3	-0.1027	-0.1337	-0.0612
TRADES	YANG ET AL. (2022)	-0.1174	-0.1368	-0.0769
	+ SEC. 3.3	-0.1179	-0.1445	-0.0836

5.4. Quantitative Analysis for Prediction Entropy

As demonstrated in Sec. 3.2, progressively reducing the predicted entropy of the input is beneficial to the regulation of its corresponding neural dynamics. While the improved performances (shown in different tables) have proved the effectiveness of the method, in this section, we conduct additional analysis to quantitatively compare the prediction entropies between a clean input and its perturbed counterpart.

We conduct the comparison using different types of attacks and adversarial training configurations. In our analysis, the adversarial inputs are generated by PGD-10 with the 144 different intermediate attacks described in the setup of Sec. 4. Under a certain attack, for each clean input $\mathbf{x}_j$ in the validation set $\mathcal{V}$ and its perturbed counterpart $\tilde{\mathbf{x}}_j$, their corresponding equilibrium states are denoted as $\mathbf{z}_j^\star$ and $\tilde{\mathbf{z}}_j^\star$.

We propose two metrics for the quantitative comparison of prediction entropy. For a certain attack, we calculate P, the percentage of clean inputs with prediction entropy lower than their perturbed counterparts. Formally,

$$P = \frac{1}{|\mathcal{V}|} \sum_j \mathbb{1}(H(\mathbf{z}_j) < H(\tilde{\mathbf{z}}_j^\star)) \times 100\%, \qquad (12)$$

where $H(\cdot)$ is the prediction entropy defined in Eq. (6).

We also calculate ΔH, the difference of the prediction entropy averaged in the validation set between the clean inputs and their perturbed counterparts:

$$\Delta H = \frac{1}{|\mathcal{V}|} \sum_j (H(\mathbf{z}_j) - H(\tilde{\mathbf{z}}_j^\star)). \qquad (13)$$

We calculate a pair of P and ΔH for each attack. For the 144 Ps and the 144 ΔHs, we list their average, their minimum, and their maximum value in Table 6 and Table 7. According to the P statistics in Table 6, an average of

over three-quarters of clean inputs have lower prediction entropy than their perturbed counterparts under all types of attacks. Furthermore, the ΔH statistics in Table 7 show that the averaged prediction entropy of clean inputs in the validation set is always less than that of the perturbed inputs. These two quantitative findings again verify the viability of our entropy reduction method in Sec. 3.2.

6. Related Work

6.1. Training-Time Adversarial Defense

Of all the training-time adversarial defense approaches, adversarial training has proven to be the most practical and effective technique for improving adversarial robustness (Athalye et al., 2018). However, AT only regulates the input-output behavior of neural models, leaving the internal neural dynamics under-supervised. The most related effort of explicit regulations along the entire neural dynamics is interval bound propagation (IBP) (Gowal et al., 2018; Huang et al., 2019; Zhang et al., 2020). IBP is a technique from the certificated robustness field that envelopes the neural dynamics of deep networks with layer-wise linear functions for robustness guarantees. However, the complicated training procedure and the lack of scalability hinder its practical use. In our work, we exploit the structural uniquenesses of DEQs to impose explicit regulations along their neural dynamics.

6.2. Test-Time Adversarial Defense

Recently, several test-time defense techniques have been proposed to exploit additional computes during inference time for robustness improvement (Pang et al., 2020; Shi et al., 2021; Mao et al., 2021; Wu et al., 2021; Alfarra et al., 2022). Croce et al. (2022a) conduct a thorough evaluation for test-time defenses. Our method is different from the previous works, as they focus on traditional deep networks, and usually calibrate only the output behavior of the model. In contrast, we progressively update the input along the forward pass to mount the neural dynamics of DEQ models to correct "orbits" in Sec. 3.2. The fundamental difference between our work and prior arts originates from the special design of DEQ models, as they directly cast the forward process as solving the fixed-point equation iteratively.

6.3. Dynamical System Perspective for Neural Models

E (2017) first proposes to interpret deep networks from a dynamical system perspective, which draws the connection between residual networks and the solution of an ODE with the forward Euler method. Since then, multiple types of novel neural models have been proposed, which directly model a dynamical system in their forward pass. Among them, neural ODEs (Chen et al., 2018) are integrated with continuous ODE, while DEQ models (Bai et al., 2019; 2020) are instantiated by discrete fixed-point iteration systems.

Several efforts have been made in designing robust neural ODEs by drawing inspiration from control theory (Zhang et al., 2019a; Yang et al., 2020; Kang et al., 2021a). By comparison, for DEQ models, Jacobian regularization is proposed in (Bai et al., 2021) by regulating only the equilibrium state to improve training stability instead of robustness. In our work, we explicitly regulate the entire neural dynamics of DEQ models to improve the adversarial robustness.

7. Conclusion

In this work, we propose to reduce the prediction entropy of intermediate states along the DEQ neural dynamics with progressive input updates. We also randomly select intermediate states to compute the loss function during adversarial training of DEQ models. Our work significantly outperforms previous works on improving DEQ robustness and even surpasses strong deep network baselines. Our work sheds light on explicitly regulating DEQ and other neural models from the perspective of neural dynamics.

In the future, we will continue to exploit the special properties (single layer, fixed-point structure, neural dynamics, etc.) of DEQ models to design tailored adversarial defense strategies. We will also investigate the relationship between our methods and the inexact/approximated gradient proposed for implicit models (Fung et al., 2022; Geng et al., 2021). We also leave the validation of our methods on larger benchmarks as future work.

Acknowledgment

We thank all of the anonymous reviewers for their constructive suggestions. This work was supported by the National Key R&D Program of China (2022ZD0160502) and the National Natural Science Foundation of China (No. 61925601, 62276152, 62236011).

References

Alfarra, M., Pérez, J. C., Thabet, A. K., Bibi, A., Torr, P. H. S., and Ghanem, B. Combating adversaries with anti-adversaries. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pp. 5992–6000. AAAI Press, 2022.

Andriushchenko, M., Croce, F., Flammarion, N., and Hein, M. Square attack: a query-efficient black-box adversarial attack via random search. In *European Conference on Computer Vision (ECCV)*, pp. 484–501. Springer, 2020.

Athalye, A., Carlini, N., and Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2018.

Bai, S., Kolter, J. Z., and Koltun, V. Deep Equilibrium Models. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 688–699, 2019.

Bai, S., Koltun, V., and Kolter, J. Z. Multiscale Deep Equilibrium Models. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Bai, S., Koltun, V., and Kolter, J. Z. Stabilizing Equilibrium Models by Jacobian Regularization. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 554–565. PMLR, 2021.

Bai, S., Geng, Z., Savani, Y., and Kolter, J. Z. Deep Equilibrium Optical Flow Estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Brin, M. and Katok, A. On local entropy. geometric dynamics (rio de janeiro, 1981)(lecture notes in mathematics, 1007), 1983.

Broyden, C. G. A Class of Methods for Solving Nonlinear Simultaneous Equations. *Mathematics of Computation*, 19:577–593, 1965.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, 2018.

Chen, T., Lasserre, J. B., Magron, V., and Pauwels, E. Semialgebraic Representation of Monotone Deep Equilibrium Models and Applications to Certification. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021.

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning (ICML)*, 2020.

Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., Mittal, P., and Hein, M. RobustBench: a standardized adversarial robustness benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

Croce, F., Gowal, S., Brunner, T., Shelhamer, E., Hein, M., and Cemgil, A. T. Evaluating the adversarial robustness of adaptive test-time defenses. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., and Sabato, S. (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 4421–4435. PMLR, 2022a. URL https://proceedings.mlr.press/v162/croce22a.html.

Croce, F., Gowal, S., Brunner, T., Shelhamer, E., Hein, M., and Cemgil, T. Evaluating the adversarial robustness of adaptive test-time defenses. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 4421–4435. PMLR, 17–23 Jul 2022b. URL https://proceedings.mlr.press/v162/croce22a.html.

E, W. A Proposal on Machine Learning via Dynamical Systems. *Communications in Mathematics and Statistics*, 5(1):1–11, 2017.

Fung, S. W., Heaton, H., Li, Q., McKenzie, D., Osher, S., and Yin, W. Jfb: Jacobian-free backpropagation for implicit networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 6648–6656, 2022.

Geng, Z., Zhang, X.-Y., Bai, S., Wang, Y., and Lin, Z. On Training Implicit Models. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021.

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T. A., and Kohli, P. On the effectiveness of interval bound propagation for training verifiably robust models. *CoRR*, abs/1810.12715, 2018.

Gurumurthy, S., Bai, S., Manchester, Z., and Kolter, J. Z. Joint inference and input optimization in equilibrium networks. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on*

Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.

Huang, P.-S., Stanforth, R., Welbl, J., Dyer, C., Yogatama, D., Gowal, S., Dvijotham, K., and Kohli, P. Achieving verified robustness to symbol substitutions via interval bound propagation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, November 2019.

Huang, Z., Bai, S., and Kolter, J. Z. $(\text{Implicit})^2$: Implicit Layers for Implicit Representations. In *Advances in Neural Information Processing Systems*, 2021.

Jafarpour, S., Abate, M., Davydov, A., Bullo, F., and Coogan, S. Robustness Certificates for Implicit Neural Networks: A Mixed Monotone Contractive Approach. *CoRR*, abs/2112.05310, 2021.

Kang, Q., Song, Y., Ding, Q., and Tay, W. P. Stable neural ODE with lyapunov-stable equilibrium points for defending against adversarial attacks. In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 14925–14937, 2021a.

Kang, Q., Song, Y., Ding, Q., and Tay, W. P. Stable Neural ODE with Lyapunov-Stable Equilibrium Points for Defending Against Adversarial Attacks. *Advances in Neural Information Processing Systems*, 34:14925–14937, 2021b.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Li, G., Müller, M., Ghanem, B., and Koltun, V. Training Graph Neural Networks with 1000 Layers. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6437–6449. PMLR, 2021.

Li, Q., Chen, L., Tai, C., and E, W. Maximum principle based algorithms for deep learning. *J. Mach. Learn. Res.*, 18:165:1–165:29, 2017. URL http://jmlr.org/papers/v18/17-653.html.

Loshchilov, I. and Hutter, F. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Lu, C., Chen, J., Li, C., Wang, Q., and Zhu, J. Implicit Normalizing Flows. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Lyapunov, A. M. The general problem of the stability of motion. *International journal of control*, 55(3):531–534, 1992.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

Mao, C., Chiquier, M., Wang, H., Yang, J., and Vondrick, C. Adversarial attacks are reversible with natural supervision. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pp. 641–651. IEEE, 2021.

Müller, M. N., Staab, R., Fischer, M., and Vechev, M. T. Effective Certification of Monotone Deep Equilibrium Models. *CoRR*, abs/2110.08260, 2021.

Pabbaraju, C., Winston, E., and Kolter, J. Z. Estimating Lipschitz constants of monotone deep equilibrium models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Pang, T., Xu, K., and Zhu, J. Mixup inference: Better exploiting mixup to defend adversarial attacks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ByxtC2VtPB.

Pang, T., Yang, X., Dong, Y., Su, H., and Zhu, J. Bag of tricks for adversarial training. In *International Conference on Learning Representations (ICLR)*, 2021.

Pang, T., Lin, M., Yang, X., Zhu, J., and Yan, S. Robustness and accuracy could be reconcilable by (Proper) definition. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 17258–17277. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/pang22a.html.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8024–8035, 2019.

Pesin, Y. B. Characteristic lyapunov exponents and smooth ergodic theory. *Russian Mathematical Surveys*, 32(4):55, aug 1977. doi: 10.1070/RM1977v032n04ABEH001639. URL https://dx.doi.org/10.1070/RM1977v032n04ABEH001639.

Pokle, A., Geng, Z., and Kolter, J. Z. Deep equilibrium approaches to diffusion models. In *Advances in Neural Information Processing Systems*, 2022.

Pontryagin, L. S., Mishchenko, E., Boltyanskii, V., and Gamkrelidze, R. The mathematical theory of optimal processes, 1962.

Revay, M., Wang, R., and Manchester, I. R. Lipschitz Bounded Equilibrium Networks. *CoRR*, abs/2010.01732, 2020.

Rice, L., Wong, E., and Kolter, J. Z. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning (ICML)*, 2020.

Shi, C., Holtz, C., and Mishne, G. Online adversarial purification based on self-supervised learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=_i3ASPp12WS.

Wei, C. and Kolter, J. Z. Certified robustness for deep equilibrium models via interval bound propagation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=y1PXylgrXZ.

Winston, E. and Kolter, J. Z. Monotone operator equilibrium networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Wu, B., Pan, H., Shen, L., Gu, J., Zhao, S., Li, Z., Cai, D., He, X., and Liu, W. Attacking adversarial attacks as A defense. *CoRR*, abs/2106.04938, 2021. URL https://arxiv.org/abs/2106.04938.

Yan, H., Du, J., Tan, V., and Feng, J. On robustness of neural ordinary differential equations. In *ICLR*, 2019.

Yang, Z., Liu, Y., Bao, C., and Shi, Z. Interpolation between residual and non-residual networks. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10736–10745. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/yang20g.html.

Yang, Z., Pang, T., and Liu, Y. A Closer Look at the Adversarial Robustness of Deep Equilibrium Models. In *Advances in Neural Information Processing Systems*, 2022.

Yoon, J., Hwang, S. J., and Lee, J. Adversarial purification with score-based generative models. In *International Conference on Machine Learning*, pp. 12062–12072. PMLR, 2021.

Young, L.-S. Entropy in dynamical systems. *Entropy*, 313: 114–127, 2003.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In *The British Machine Vision Conference (BMVC)*, 2016.

Zhang, D., Zhang, T., Lu, Y., Zhu, Z., and Dong, B. You Only Propagate Once: Painless Adversarial Training Using Maximal Principle. In *NeurIPS*, 2019a.

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning (ICML)*, 2019b.

Zhang, H., Chen, H., Xiao, C., Gowal, S., Stanforth, R., Li, B., Boning, D. S., and Hsieh, C. Towards stable and efficient training of verifiably robust neural networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

A. Experiment Details

We follow to adopt the DEQ-Large architecture used in Yang et al. (2022), which has similar parameter counts as ResNet-18. More specifically, the DEQ cell we use is the multiscale DEQ-Large with 4 scales. The numbers of head channels for each scale are 14, 28, 56, and 112. The numbers of channels for each scale are 32, 64, 128, and 256. The channel size of the final layer is 1680. We use Broyden's method as the black-box solver, with $N = 8$ forward solver iterations.

Following Bai et al. (2020; 2021), the model is trained with Adam optimizer: the initial learning rate is 0.001 with cosine decay; the Nesterov momentum is 0.98; the weight decay is 0. We follow Bai et al. (2020; 2021) and Yang et al. (2022) to set batch size as 96. We pretrain the DEQ models with the truncated deep networks (with standard training) for a good initialization of θ for 16000 steps, and then conduct adversarial training, until the total of 210 epochs training finishes. Following Yang et al. (2022), we use the unrolling-based phantom gradient to train the DEQ models with 5 unrolling steps (Geng et al., 2021). For PGD-AT training, the perturbation range is $\epsilon = 8/255$; the step size is $\alpha = 2/255$; the number of PGD steps during training is 10. Additionally, for TRADES training, the $1/\lambda = 6$.

We select the best model weight with the top robust accuracy on the CIFAR-10 validation set under ready-made PGD-10 (at the final state after unrolling). After obtaining the weight, we follow Yang et al. (2022) to leverage early-state defenses. In our experiments, we always use the last but one intermediate state as the early state for robustness evaluation. We leverage the unrolled intermediates method for intermediate attacks, as they form consistently stronger attacks than the simultaneous adjoint method. For the Sec. 3.2 method, we set $\beta = 2/255$, $T_f = 2$, and $R = 10$ in our main experiments. All experiments are conducted on a single NVIDIA 3090 GPU. Our code is available at `https://github.com/minicheshire/DEQ-Regulating-Neural-Dynamics`.

B. The Entropy in Dynamical Systems

In this section, we provide a brief introduction to the entropy in dynamical systems. This introduction heavily relies on Young (2003). We draw connections between the concepts of the entropy in dynamical systems with DEQ models.

Definition B.1. Let $(\mathcal{X}, d)$ be a metric space, and μ be a probability measure on it. Let $\alpha = \{X_1, \cdots, X_C\}$ be a finite partition of $\mathcal{X}$. Then the entropy of the partition is defined as

$$H(\alpha) = H(\{X_1, \cdots, X_C\}) = -\sum_{i=1}^{C} \mu(X_i) \log \mu(X_i). \tag{14}$$

Implications: Interpreting $\mathcal{X}$ as the space of $\mathbf{z}^{[t]}$ in DEQ models, the partition of $\mathcal{X}$ is naturally constructed by the C classes in a classification task. μ can therefore be instantiated by the classification head in DEQ models.

Definition B.2. Let (f, μ) be an ergodic discrete dynamical system, with $f : \mathcal{X} \to \mathcal{X}$ is the mapping from $\mathcal{X}$ to $\mathcal{X}$. We define the α-address of the n-orbit starting at $x \in \mathcal{X}$ as

$$\bigvee_{i=0}^{n-1} f^{-i}\alpha = \{x : x \in X_{i_0}, fx \in X_{i_1}, \cdots, f^{n-1}x \in X_{i_{n-1}}\} \tag{15}$$

for some $(i_0, i_1, \cdots, i_{n-1})$. The metric entropy of f with partition α is then defined as

$$h_\mu(f, \alpha) = \lim_{n\to\infty} \frac{1}{n} H(\bigvee_{i=0}^{n-1} f^{-i}\alpha). \tag{16}$$

Implications: The n-orbit of $\{x, fx, \cdots, f^{n-1}x\}$ coincides with the n-length neural dynamics of DEQ models we defined in Eq. 3. The α-address $\bigvee_{i=0}^{n-1} f^{-i}\alpha$ corresponds to all xs that induces an n-length orbit traversing different subspaces of the partitions with the mapping f. Of all xs that comprise the α-address of the n-orbit, $H(\bigvee_{i=0}^{n-1} f^{-i}\alpha)$ characterizes the entropy of the address and roughly reflects how "chaotic" the f is under the partition α in the space $\mathcal{X}$. While $h_\mu(f, \alpha)$ is defined by taking the limit of $n \to \infty$, $\lim_{n\to\infty} f^{n-1}x$ is exactly the equilibrium point of f starting with x. This suggests that $h_\mu(f, \alpha)$ is defined over the entire orbit and concerned about the equilibrium state behavior.

Theorem B.3. *For $x \in \mathcal{X}$, $n \in \mathbb{N}^+$, $\epsilon > 0$, define*

$$B(x, n, \epsilon) := \{y \in \mathcal{X} : d(f^i x, f^i y) < \epsilon, 0 \leq i < n\}. \tag{17}$$

With μ defined over partition α, we have

$$h_\mu(f, \alpha) = \limsup_{n \to \infty} -\frac{1}{n} \log \mu B(x, n, \epsilon). \tag{18}$$

Specifically, if $\mathcal{X}$ is a compact Riemann manifold with μ equivalent to the Riemannian measure, we have

$$h_\mu(f, \alpha) = \sum_i \max(\lambda_i, 0), \tag{19}$$

where λ_is are the Lyapunov exponents of (f, μ).

Proof. The proof is given by Pesin (1977) and Brin & Katok (1983). □

Implications: $B(x, n, \epsilon)$ indicates the "amount" of the neighbors of x that induce the trajectories close to that of x with f. This definition shares a similar idea with our derivation in Sec. 3.1, as we consider the deviation of the neural dynamics under clean or perturbed inputs. We want all the perturbed inputs to lie within $B(x, n, \epsilon)$ in Eq. 18, in this way all the orbits are regulated. This corresponds with an increase in $\mu B(x, n, \epsilon)$, and from Eq. 18 we realize that reducing the entropy $h_\mu(f, \alpha)$ achieves this. From Eq. (19) we know that reducing $h_\mu(f, \alpha)$ is also equivalent to reducing the sum of the Lyapunov exponents of the system. Smaller Lyapunov exponents imply more stable dynamical systems (Lyapunov, 1992), which is equivalent to more robust neural models as demonstrated by Yang et al. (2020); Kang et al. (2021a). Such correlations motivate us to observe and reduce prediction entropy along the neural dynamics in DEQ models.

C. Additional Ablation Studies

C.1. Effect of λ in Eq. (11) of Intermediate-State Attacks

In this section, we compare the effect of setting λ to be 1.0 or 0.5 in Eq. (11) as the intermediate-state attacks. We use the "TRADES+Secs.3.2 & 3.3" defense method for this study. Similar to Fig. 3-(a), we plot the lowest accuracy results under the attacks at each intermediate states along the neural dynamics in Fig. 5. It is obvious that $\lambda = 0.5$ builds consistently stronger attacks than $\lambda = 1.0$.

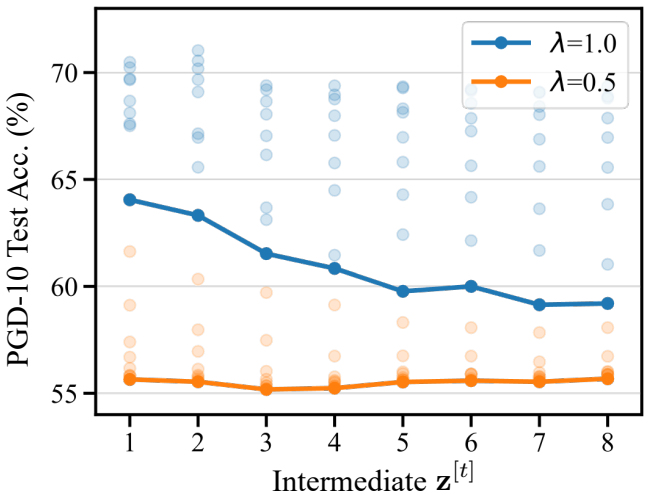

Figure 5. Performance (%) of the "TRADES+Secs.3.2 & 3.3" defense under all intermediate-state attacks along the neural dynamics, with $\lambda = 1.0$ and $\lambda = 0.5$. $\lambda = 0.5$ forms the intermediate-state attacks consistently stronger than those with $\lambda = 1.0$.

Table 8. The strongest attacks in Eq. (11) at each intermediate state along the neural dynamics.

$\lambda = 1.0$	(i, K_a)	$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(5,1)$	$(6,1)$	$(7,1)$	$(8,1)$
	PGD	64.05	63.32	61.53	60.84	59.77	60.00	59.14	59.20
$\lambda = 0.5$	(i, K_a)	$(1,8)$	$(2,7)$	$(3,5)$	$(4,5)$	$(5,5)$	$(6,5)$	$(7,4)$	$(8,5)$
	PGD	55.65	55.54	55.18	55.24	55.53	55.59	55.54	55.68

We further list the strongest attacks at each intermediate state along the neural dynamics in Table 8. For $\lambda = 1.0$, the best setting of K_a for each state $\mathbf{z}^{[t]}$ is 1. The overall strongest attack is formed by $(7, 1)$. It is noted that $\mathbf{z}^{[7]}$ is one iteration away from $\mathbf{z}^\star = \mathbf{z}^{[8]]}$. On the contrary, for $\lambda = 0.5$, the strongest attack is in the middle of the neural dynamics, with $(i, K_a) = (3, 5)$, and K_a is larger than 1 under $\lambda = 0.5$. It is inferred that $\lambda = 0.5$ constructs intermediate attacks with more accurate gradient estimation: the estimated gradients become more accurate as the unrolling step K_a becomes larger, so that the corresponding attacks are stronger.

C.2. Effect of the Jacobian Regularization Factor

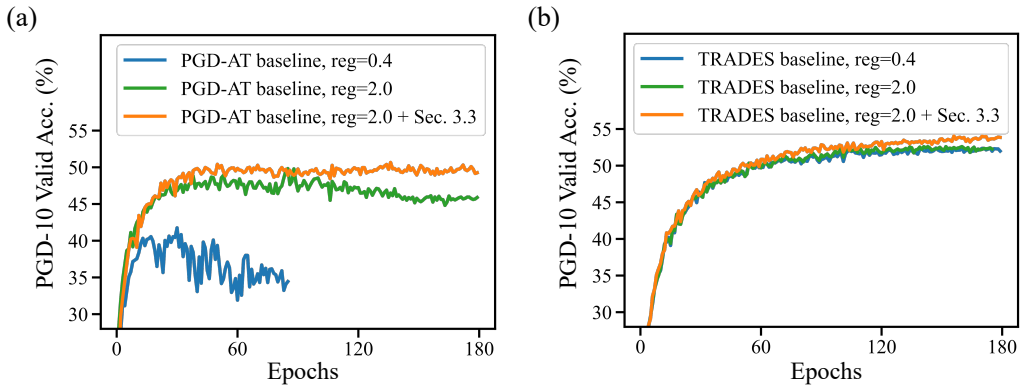

Figure 6. Comparisons between different Jacobian regularization factors during the training phase of baseline methods.

We also tried to train DEQ models by setting the Jacobian regularization factor proposed by Bai et al. (2021) as 0.4, following Yang et al. (2022). However, shown in Fig. 6, we found that under PGD-AT, the training becomes unstable with suboptimal robustness performance. After increasing the factor to 2.0, we found the performance during training is improved. We suspect that a larger Jacobian regularization factor is required to stabilize the training PGD-AT. In contrast, when trained with TRADES, we found that the empirical difference between setting the factor to be 0.4 or 2.0 is marginal. It is therefore inferred that TRADES might implicitly impose more regularization during the adversarial training process, which might also account for the superiority of TRADES experiments.

D. Running Time Analysis

In this section, we compare the time cost of our methods with the vanilla adversarial training baseline for DEQ models. We first compare the training-time methods: the vanilla baseline and the Section 3.3 method. The comparison is shown in Table 9.

Table 9. Training speed comparison (Samples/s) between the vanilla adversarial training baseline (Yang et al., 2022) and the Sec. 3.3 method.

AT FRAMEWORK	METHOD	TRAINING SPEED (SAMPLES/S)
PGD-AT	Yang et al. (2022)	30.2
	+ OURS (SEC. 3.3)	**39.9**
TRADES	Yang et al. (2022)	24.0
	+ OURS (SEC. 3.3)	**30.6**

According to Table 9, our Sec.3.3 method is faster than the vanilla adversarial training. This is because we use random intermediate states for loss computation in Sec. 3.3. In this way, the fixed-point solver in the forward process runs fewer than N iterations. In contrast, the solver always runs for N iterations in the baseline.

Next, we compare the running speed among different settings in the Sec. 3.2 method during inference.

Table 10. Inference speed comparison (Samples/s) with different settings in the Sec. 3.2 method during inference. $R = 0$ indicates the baseline method without Sec. 3.2.

	$R = 0$	$R = 1$	$R = 5$	$R = 10$	$R = 20$
$T_f = 1$	94.9	32.1	11.8	8.6	6.5
$T_f = 2$		42.6	20.5	11.7	7.5
$T_f = 4$		48.6	29.9	18.6	12.8
$T_f = 8$		49.6	33.1	26.6	17.6

Reported in Table 10, the time consumption of Sec. 3.2 method is roughly proportional to R and inversely proportional to T_f. This coincides with our Algorithm 1. Comparing Table 10 with Table 5, we set $R = 10$ and $T_f = 2$ to achieve the trade-off between inference speed and robustness performance.