Networks and Heterogeneous Media

https://www.aimspress.com/journal/nhm

NHM, 21(1): 170–182.
DOI: 10.3934/nhm.2026007
Received: 24 July 2025
Revised: 22 December 2025
Accepted: 05 January 2026
Published: 27 January 2026

Research article

Random weights of DNNs and emergence of fixed points

L. Berlyand[1], **O. Krupchytskyi**[1] **and V. Slavin**[2,*]

[1] Department of Mathematics, Pennsylvania State University, University Park, Pennsylvania 16802, USA

[2] B. Verkin Institute for Low Temperature Physics and Engineering of the National Academy of Sciences of Ukraine, Nauky Ave., 47, Kharkiv 61103, Ukraine

* **Correspondence:** Email: slavin@ilt.kharkov.ua.

Abstract: We perform a numerical study of autoencoder deep neural networks (DNNs) when the input and the output vectors have the same dimension. Our focus is on fixed points (FPs) arising in these DNNs. We show that the existence and the number of these FPs depend on the distribution of randomly initialized DNNs' weight matrices. We first consider initialization with the identically and independently distributed (i.i.d.) light-tailed distributions of weights (e.g., Gaussian) and show existence of a single stable FP for a wide class of DNN architectures. In contrast, for heavy-tailed distributions (e.g., Cauchy), which typically appear after the training of DNNs, a number of stable FPs emerge. We observe an intriguing non-monotone dependence of the number of FPs on the DNN's depth. Finally, we link our result for untrained DNNs to the trained ones by showing that a number of FPs emerge after training of DNNs with light-tailed initialization.

Keywords: autoencoder deep neural network; random initialization; fixed points; stability and basin of attraction

1. Introduction

In recent years, a variety of new technologies based on deep neural networks (DNNs), also known as artificial neural networks (ANNs) have been developed. AI-based technologies have been successfully used in physics, medicine, business and everyday life (see e.g., [1]). The two key theoretical directions in DNN theory are the development of novel (i) types of DNNs and (ii) training algorithms.

One of the most important applications of DNNs is the processing of visual information [2,3]. Image transformation (also known as image-to-image translation) involves a transformation of the original image into another image according to the goals, such as, enlarging the pictures without losing the quality. Another important example is self-mapping transformation or autoencoder DNNs. Such DNNs

are used e.g., for image restoration where the restored image is a fixed point (FP) of the DNN [4]. The proximity of a DNN's output vector to an FP can be used as a stopping criterion for DNNs' training.

Note that the FPs of DNNs have many applications beyond image-to-image transformation. In the modeling of the brain, FPs appear in the time evolution of networks [5–9], whereas the networks considered here are static. In addition, most of the Firing model studies deals with nonrandom weight matrices. Other prominent examples are Hopfield networks [10, 11], where FPs are used for memory modeling [12]. The Hopfield model is also used in quantum physics, where FPs describe phase transitions [13]. Note that in Hopfield networks, the FPs of loss function are considered, while we study the FPs of DNNs.

In this work, using numerical methods, we study the dependence of the properties of FPs on random distributions of i.i.d. weight matrices and on the network architecture.

2. The model: Image-to-image transformation and FPs

We consider a fully-connected feedforward network where layer-to-layer transformation is a composition of the affine map with the nonlinear activation function [14]. The output vector $\boldsymbol{x}^{l+1}$ of the l-th layer of the DNN is

$$\boldsymbol{x}^{l+1} = \boldsymbol{\Phi}^l(\boldsymbol{x}^l) = \varphi(\mathbf{W}^l\boldsymbol{x}^l + \mathbf{b}^l), \tag{2.1}$$

where $\boldsymbol{W}^l$ is a real-valued $n_{l+1} \times n_l$ weight matrix, $\boldsymbol{b}^l \in \mathbb{R}^{n_l}$ is a bias vector, and the function φ is the nonlinear activation function, (see e.g., [14,15]). For simplicity of presentation, we consider the square weight matrices of hidden layers, so that $n_l = N$, $l = 1, 2, \ldots, L-2$.

DNN is then a function $\boldsymbol{\Phi}$ that maps the input vector $\boldsymbol{x}^0$ into the output vector $\boldsymbol{x}^l$.

$$\boldsymbol{\Phi}(\boldsymbol{x}^0) = (\boldsymbol{\Phi}^{L-1} \circ \boldsymbol{\Phi}^{L-2} \circ \cdots \circ \boldsymbol{\Phi}^1 \circ \boldsymbol{\Phi}^0)(\boldsymbol{x}^0) = \boldsymbol{x}^L. \tag{2.2}$$

The FPs are defined for autoencoder types of networks $\boldsymbol{\Phi}$, when the input and the output vectors have the same dimension, $n_0 = n_L$. The function $\boldsymbol{\Phi}$ is parametrized by weights and biases that hereafter will be denoted by α; i.e., $\boldsymbol{\Phi} = \boldsymbol{\Phi}(\boldsymbol{x}, \alpha)$.

Let us consider the problem of encoding and decoding of a single picture [16,17]. Let $\boldsymbol{x}$ correspond to the original picture. For its encoding, we use the follwoing DNN $\boldsymbol{\Phi}_c$: $\mathbb{R}^{n_0} \to \mathbb{R}^{n_1}$:

$$\boldsymbol{\Phi}_c(\boldsymbol{x}) = \boldsymbol{y},$$

where $\boldsymbol{y} \in \mathbb{R}^{n_1}$ is the *encoded picture* (n_1 is the size of $\boldsymbol{y}$; in autoencoder DNNs [16], $n_1 < n_0$). For picture decoding, we use another DNN $\boldsymbol{\Phi}_d : \mathbb{R}^{n_1} \to \mathbb{R}^{n_0}$. Let $\boldsymbol{\Phi}_d(\boldsymbol{y}) = \boldsymbol{z}$, where $\boldsymbol{z} \in \mathbb{R}^{n_0}$ is the *decoded (restored) picture*. Let DNN $\boldsymbol{\Phi} : \mathbb{R}^{n_0} \to \mathbb{R}^{n_0}$ be the following composition:

$$\boldsymbol{\Phi}(\boldsymbol{x}) = (\boldsymbol{\Phi}_d \circ \boldsymbol{\Phi}_c)(\boldsymbol{x}) = \boldsymbol{z}. \tag{2.3}$$

Autoencoders were originally designed to learn how to represent image data $\boldsymbol{x}_i$, $i = 1, 2, \ldots, K$ in a compressed format $\boldsymbol{y}_i$ and then restore the compressed data $\boldsymbol{y}_i$ to $\boldsymbol{z}_i$. The goal of training in this case is $\boldsymbol{x}_i = \boldsymbol{z}_i$. Therefore, the mean square loss is

$$\mathcal{L}(\alpha) = \sum_{i=1}^{K} \|\boldsymbol{\Phi}(\boldsymbol{x}_i, \alpha) - \boldsymbol{x}_i\|^2 \ . \tag{2.4}$$

The images $\boldsymbol{x}_i$ in Eq (2.3) are FPs [18].

$$\boldsymbol{\Phi}(\boldsymbol{x}_i) = \boldsymbol{x}_i. \tag{2.5}$$

Here, we consider the method of picture encoding/decoding based on autoencoders. The method can be used, for example, for employees' access control. In this case, an employee's photo can be encoded using DNN $\boldsymbol{\Phi}_c$ and then securely transmitted via the network to access the server for decoding using $\boldsymbol{\Phi}_d$ and for access control. Let K be a number of employees. To perform training, we start with the input vectors $\boldsymbol{x} \in T_k$, $k = 1, 2, \ldots, K$. Here, T_k is the training set that contains the photos of the k-th employee. One of these photos, $\boldsymbol{x}_k^* \in T_k$, can be considered to be the "true" photo of the employee stored on the access control server for identification. The other photos in T_k are different photos of the same employee (c.f., various fingerprints in a touch id, only one works). The FPs $\boldsymbol{\Phi}(\boldsymbol{x}_k^*) = \boldsymbol{x}_k^*$ are obtained via training with the mean square loss

$$\mathcal{L}(\alpha) = \sum_{k=1}^{K} \sum_{\boldsymbol{x} \in T_k} \left\| \boldsymbol{\Phi}(\boldsymbol{x}, \alpha) - \boldsymbol{x}_k^* \right\|^2, \tag{2.6}$$

The FPs $\boldsymbol{x}_k^*$ differ essentially from Eq (2.5) because of the more complex structure of loss (c.f. Eqs (2.4) and (2.6)). As a result, it becomes possible to distinguish the "true" photo of one employee $\boldsymbol{x}_k^*$ from the "true" photo of another employee $\boldsymbol{x}_{k'}^*$ and to distinguish a photo of the real employee from a "fake" photo of the employee. If photo $\boldsymbol{x}$ is in the basin of attraction of $\boldsymbol{x}_k^*$, $k = 1, 2, \ldots K$, then $\boldsymbol{x}$ is a real photo of the k-th employee; otherwise, it is a fake photo.

Note 1. These FPs are also distinguish from FPs arising in the special case of a singl-layer, deterministic, non-negative DNN [19].

Note 2. Instead of Eq (2.6), one can use, say, the cross-entropy loss function [14, 20]. There are no explicit formulas for $\boldsymbol{\Phi}_c$ and $\boldsymbol{\Phi}_d$ in this procedure. In order to restore a picture, one has to know all the weights obtained in training. This is a significant protection against hacking.

3. Light- vs heavy-tailed distributions and DNN's training

We now explain how "heavy-tailed" distributions arise in DNNs. Typical initialization of weights and biases is done with the light-tailed (subexponential) distributions, e.g., Gaussian. Such initializations are widely used for training via stochastic gradient descent (SGD) (see [20–26]). Note that there are many modifications of SGD training based on random matrix theory (RMT) approaches aimed at improving DNNs' performance; for example, Marchenko-Pastur pruning of singular values of random weight matrices enhances DNN's accuracy while reducing the noise [27, 28]. Numerical studies in the seminal work showed that the initialization of the weight matrices by a "light-tailed" distribution *becomes "heavy-tailed"* in the course of training. This phenomenon is known as the heavy-tailed self-regularization [29]. Moreover, recently it was shown that the input-output Jacobian of a trained DNN also has heavy-tailed empirical spectral distributions [30–33]. Heavy-tailed self-regularization allows us to use the tools of RMT for studying the FP properties of untrained and trained DNNs.

In the model of image encoding/decoding, a FP corresponds to a "true" photo of employee, $\boldsymbol{x}_k^*$, and the transition from a light-tailed to a heavy-tailed distribution during training will lead to a drastic change in the number of these FPs, their stability, and the shapes/sizes of the basins of attractions.

4. Fixed points and their basins of attraction

Here we describe our numerical calculations of FPs in untrained DNNs. For simplicity of presentation, the dimension of input/output vectors $\boldsymbol{x}$ is taken as $n = 2$. The space of the input vectors $\boldsymbol{x}$ was chosen as a square, namely $\Omega = [-1, 1] \times [-1, 1] \subset \mathbb{R}^2$. This choice of Ω seems to be reasonable because the range of values of most of activation functions φ is $[-1, 1]$. This square was partitioned using a grid with a step $\delta = 0.05$. The grid points are

$$\boldsymbol{x}_{j,l} = \begin{cases} x = -1 + \delta j, j = 0, 1, \ldots, \lfloor 2/\delta \rfloor \\ y = -1 + \delta l, l = 0, 1, \ldots, \lfloor 2/\delta \rfloor \end{cases}, \tag{4.1}$$

where $\lfloor \ldots \rfloor$ denotes an integer part. For each $\boldsymbol{x}_{j,l}$, we run iterative procedure:

$$\boldsymbol{x}^{m+1} = \boldsymbol{\Phi}(\boldsymbol{x}^m), \quad m = 1, 2, 3, \ldots, \tag{4.2}$$

where $\boldsymbol{x}^1 = \boldsymbol{x}_{j,l}$. For the contraction mapping $\boldsymbol{\Phi}$ on a domain Ω, Banach's fixed-point theorem guarantees convergence to a FP $\boldsymbol{x}^*$

$$\lim_{m \to \infty} \boldsymbol{x}^{m+1} = \boldsymbol{\Phi}(\boldsymbol{x}^m) = \boldsymbol{x}^*, \quad \boldsymbol{x}^1 \in \Omega. \tag{4.3}$$

The contraction property was checked numerically, and the existence of the limit (4.3) was checked via the Cauchy criterion $|\boldsymbol{x}^{m+1} - \boldsymbol{x}^m| < \varepsilon$, $m < N_0$. In our calculations, $\varepsilon = 10^{-5}$ and $N_0 = 50$. If the limit exists, then $\boldsymbol{x}^m$ is the numerical approximation of the FP $\boldsymbol{x}^*$ corresponding to the starting grid point $\boldsymbol{x}^1 = \boldsymbol{x}_{j,l}$ defined in Eq (4.1) (for the details, see Section 5).

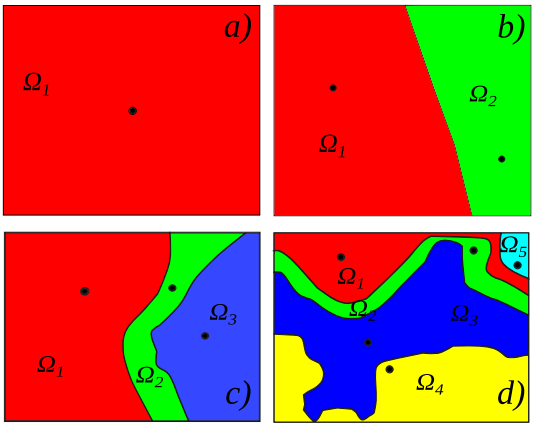

Figure 1. The results of numerical simulations for input/output vectors with a size $n = 2$ (a) for a normal distribution of the matrix elements and bias vector components. The number of layers is $L = 2$. The same result was found for the Cauchy distribution and $L = 20$. (b) Cauchy distribution for $L = 2$. (c) Cauchy distribution for $L = 3$. (d) Cauchy distribution for $L = 5$. The black circles are FPs. Different colors correspond to different basins of attraction Ω_k.

If the domain Ω contains $Q > 1$ FPs and Q basins of attraction $\Omega_k \in \Omega$, $k = 1, 2, \ldots, Q$, then for all grid points $\boldsymbol{x}^1 = \boldsymbol{x}_{j,l} \in \Omega_k$, the limit (4.3) provides a numerical approximation of the FP $\boldsymbol{x}^*_k$.

We start with an untrained DNN with depth $L = 2$. The matrix entries and the bias vector's components in Eq (2.1) are randomly initialized with the normal distribution $N(0, \sigma_l)$, $\sigma_l = (n_l)^{-1}$ $l = 0, 1$, where $n_l = \{2, 100\}$ are the layers' widths (i.e., the weight matrices sizes, $n_{l+1} \times n_l$, are 100×2 and 2×100). The unique fixed point $\boldsymbol{x} = 0$ exists (i.e., $Q = 1$) and the corresponding basin of attraction is the entire of Ω. This result can be interpreted as follows: such untrained DNNs can not identify "true" photos.

Next, using the approach based on heavy-tailed self-regularization (see [29, 32]), we model the trained DNN by an untrained DNN initialized by the Cauchy distribution centered at the origin with the scale $\gamma_l = (n_l)^{-1}$. The results are presented in Figure 1b–d. Figure 1b corresponds to the same architecture as that in Figure 1a ($L = 2$, $n_l = \{2, 100, 2\}$), but we see two FPs, $Q = 2$. Figure 1c corresponds to $L = 3$, $n_l = \{2, 100, 100\}$, and $Q = 3$. In Figure 1d, we present the results of the calculations for $L = 5$, $n_l = \{2, 100, \ldots, 100\}$, and $Q = 5$. It is important that a *further increase in depth L* leads to the *decrease* in Q, and the result for $L = 20$ is the same as that for $L = 2$ and a normal distribution — the only FP, $Q = 1$. Due to the "weak similarity" effect [32], the choice of activation function φ does not change the number of FPs.

Note that the number of FPs, and the shapes/sizes of their basins of attraction Ω_k ($k = 1, 2, \ldots, Q$) are still random because of the finite size of the matrices. An interesting open question is the existence of a deterministic limit of Q and Ω_k as $n_l \to \infty$.

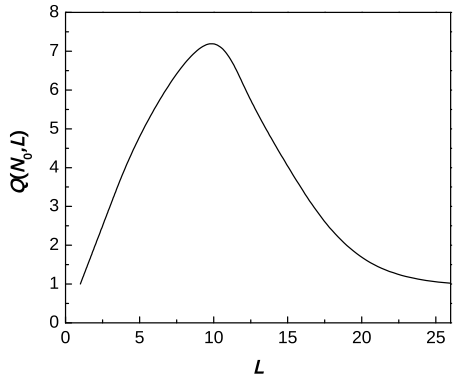

Figure 2. The dependence of the most frequently appearing data (mode) of the number of FPs $Q(N_0, L)$ on the number of layers L with layer widths $N_0 = 100$. The weights and biases initialized by random variables with a Cauchy distribution. Q first grows with the DNN depth L, but then decreases.

Finally, we numerically studied the dependence of the number of FPs on the DNN's depth L. Since this value is random for any finite matrix size N_0, we studied the dependence of the most frequently appearing data (mode) $Q(N_0, L)$ and observed surprising non-monotone dependence. The dependence of $Q(N_0, L)$ on L for DNNs with layer widths $N_0 = 100$ and with the weights and biases initialized via Cauchy distribution is presented in Figure 2.

The nature of such non-monotone behavior of $Q(N_0, L)$ is intuitively clear. For fixed matrix sizes N_0, increasing the number of layers L leads to an increase in the total number of parameters of DNN,

i.e., the total number of weight matrices entries and bias vectors components. This, in turn, allows the DNN to distinguish a larger number of input vectors $\mathbf{x}^0$, e.g., the employee's photo in our example.

At the same time, an increase in L decreases the influence of $\mathbf{x}^0$ on the output vectors $\mathbf{x}^L = \mathbf{\Phi}(\mathbf{x}^0)$. In other words, the matrix entries of the input-output Jacobian

$$\mathbf{J}_{i,j} = \frac{\partial \mathbf{\Phi}(\mathbf{x}^0)_i}{\partial \mathbf{x}^0_j} \tag{4.4}$$

tend to 0, as $L \to \infty$. For light-tailed distributions of weights and biases, this behavior of Eq (4.4) follows from the equations describing the empirical spectra distribution (ESD) $\rho(x)$ of the singular values of Eq (4.4) [30, 32]. Indeed, for $L \to \infty$, the ESD $\rho(x) = \delta(x)$, where $\delta(x)$ is the Dirac delta-function. It means that with probability 1, the derivatives in Eq (4.4) are equal to zero. Therefore, the influence of input $\mathbf{x}^0$ on output $\mathbf{x}^L$ is absent and DNN cannot distinguish different input vectors. Hence, only one FP can exist.

The interplay of these two opposite tendencies is responsible for the non-monotonic dependence of $Q(N_0, L)$. Numerical simulation carried out in [32] give us reason to expect that the similar behavior of the Jacobian at $L \to \infty$ holds for heavy-tailed distributions.

5. Contraction mapping of DNNs

In Section 4 we show numerically that for the light-tailed distribution (the Gaussian distribution in our calculations), there is a limit $\boldsymbol{x}^*$ in the iterative procedure Eq (4.2) for all initial values $\boldsymbol{x}^1$ from Ω (see Eq (4.1)). In Section 4, we show that $\mathbf{x}^*$ is an FP of a DNN $\mathbf{\Phi}$, in this section we investigate the stability and basins of attraction of such FPs. This is done by establishing numerically the contraction property of $\mathbf{\Phi}$, c.f. the converses to the Banach's FP theorem [34, 35]. Here, we show that for the light-tailed distribution of weight matrices, the mapping $\mathbf{\Phi}$ is a contraction. Moreover, we study the dependence of the contraction property of the DNN on the number of layers L, the size of the weight matrices (the number of columns, N), the choice of activation function φ, and the variance of the distribution of the weight matrices σ^2.

To this end, we choose the variance σ^2 in the form

$$\sigma^2 = N^{-2\beta}, \tag{5.1}$$

and compute the contraction constant g, defined as

$$g = \max_{\boldsymbol{x}_{j,l} \neq \boldsymbol{x}_{j',l'}} \frac{|\mathbf{\Phi}(\boldsymbol{x}_{j,l}, \alpha) - \mathbf{\Phi}(\boldsymbol{x}_{j',l'}, \alpha)|}{|\boldsymbol{x}_{j,l} - \boldsymbol{x}_{j',l'}|}, \tag{5.2}$$

where grid points $\boldsymbol{x}_{j,l}$ are given in Eq (4.1). Observe that g depends on β via the DNN's parameters α (weights and biases).

For a three-layer ($L = 3$) DNN with Gaussian initialization of the weight matrices $\mathbf{W}$ of a size $N = 400$ and the tanh activation function this dependence is presented in Figure 3a. We see that the contraction mapping property ($g < 1$) depends crucially on β. Our simulations show the existence of a critical value $\beta_{cr} \approx 1/2$, so that for $\beta \geq \beta_{cr}$, the function $\mathbf{\Phi}$ is a contraction on Ω for the tanh activation

function. Note that the numerical value of β_{cr} agrees with the analytical results for *odd activation functions*, e.g., tanh, hardtanh.

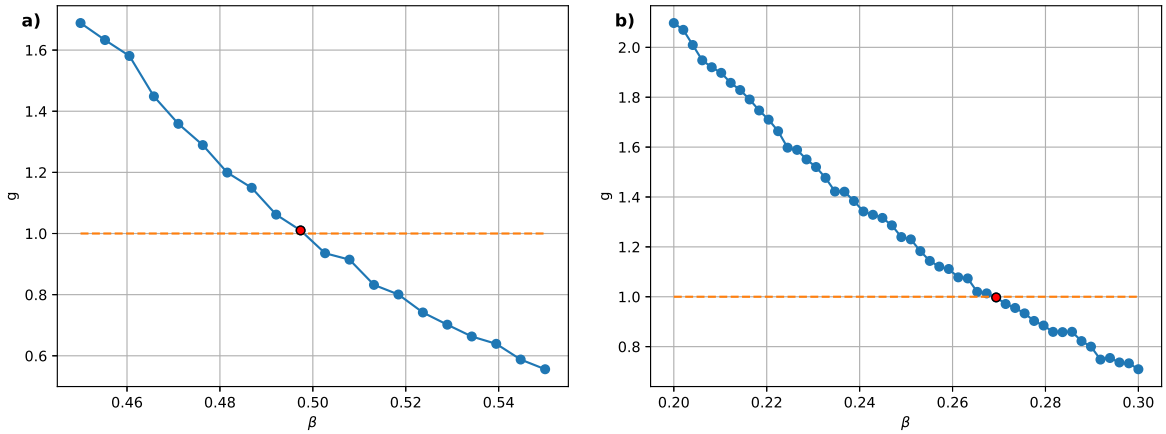

Figure 3. The dependence of the contraction mapping parameter g (5.2) on the parameter β characterizing the variance σ^2 (5.1) for Gaussian initialization of the weight matrix. The activation function is tanh in (a) with $\beta_{cr} \approx 1/2$ and sigmoid in (b) with $\beta_{cr} \approx 0.27$.

At the same time, for the three-layer DNN and the sigmoid activation function

$$\varphi(x) = \frac{1}{1+\exp(-x)} = \frac{1}{2}\left(\tanh(x/2)+1\right), \tag{5.3}$$

the critical value is $\beta_{cr} \approx 0.27$ (see Figure 3b). The difference between the two values of β_{cr} is due to the fact that the sigmoid is not an odd function and, in particular, $\varphi(0) \neq 0$ and φ acts a horizontal shift by 1; see Eq (5.3).

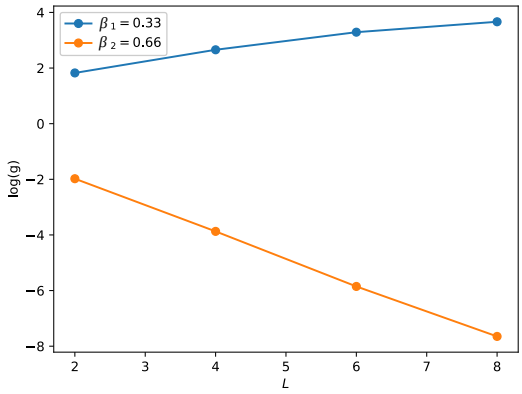

Figure 4. The dependence of the contraction constant g on the number of DNN's layers L for different values of $\beta_1 < \beta_{cr}$ and $\beta_2 > \beta_{cr}$.

The dependence of the contraction constant g on the number of layers L is presented in Figure 4. Linear dependence in the semi-log scale $log(g)$ vs. L means the power dependence of the contraction constant g on the number of layers L and reflects the composition structure of the DNN in Eq (2.2).

Indeed, if g_0 is a contraction constant of a single layer, the contraction constant g of the entire DNN is given by

$$g = (g_0)^L. \tag{5.4}$$

Our computations for the heavy-tailed Cauchy distribution also show that on each basin of attraction of $\mathbf{\Phi}$, the contraction constant is $g < 1$.

6. Emergence of FPs in trained DNNs

So far, we have studied the relation between randomness and FPs in untrained DNNs. However, training is the key ingredient in DNNs' applications. Therefore, in this section, we address similar issues for trained DNNs and emphasize the similarities and differences. The main difference is that the number of FPs in trained DNNs does depend on the DNN's depth L and it is determined by the training set T (e.g., the number of "true" photos in the above-mentioned example).

On the other hand, training results in the formation of a number of FPs similar to a transition from light-tailed to heavy-tailed distribution. The randomness in trained DNNs tends to vanish [36] and direct application of the RMT is difficult [37–39]. However, the similarities between a trained DNN and an untrained one via heavy-tailed self-regularization allows us to use RMT tools for trained DNNs. Therefore, investigations of untrained DNNs with random initialization of the weight matrices and investigations of trained DNNs complement each other.

We use the same architecture for trained DNNs as for untrained ones. The number of layers is $L = 3$, $n_l = \{2, 100, 100\}$, and activation function $\varphi(x)$ is hardtanh. The random initialization of the DNN's parameters is done via the Gaussian distribution $N(0, \sigma^2)$, where $\sigma^2 = 1/n_l$. DNN training is performed for a toy model of encoding/decoding of employees' photos. In this model, the DNN's input/output photos are represented by two-dimensional vectors ($\boldsymbol{x}^0, \boldsymbol{x}^L \subset \mathbb{R}^2$). It means that each employee's photo is represented by a point (the upper part of Figure 5). The photos of the current employee are the points $\boldsymbol{x}$ inside the corresponding circle T_k, i.e., $\boldsymbol{x} \in T_k$. These points form the training set for the k-th employee ($k = 1, 2, \ldots, K$). The number of the circles is the number of employees, $K = 5$. The centers of each circle are marked by solid black circle $\boldsymbol{x}_k^*$ and represent "true" photos. The rest of points inside a circle represent the "false" photos of the same employee. The loss function is chosen in the form of Eq (2.6).

After training, we search for the FPs of the DNN. Similar to the case of untrained DNNs, we run the iterative procedure (4.2) for each starting point $\boldsymbol{x}^1$ of the set $\boldsymbol{x}_{j,l} \subset \Omega$ (see Eq (4.1)). If the process converges, then the corresponding FP is marked as $*$ (see the lower part of Figure 5). Numerically, we see that the positions of FPs coincide with the "true" photos $\boldsymbol{x}_k^*$. The different colors of the subdomains in the lower part of Figure 5 correspond to different FPs and their basins of attractions Ω_k. In particular, $T_k \subset \Omega_k$; that is, each basin of attraction is larger then the corresponding training set. This means that this DNN can identify photos outside the training sets.

We briefly summarize the results of this section.

- The contraction mapping property of the DNN depends crucially on the parameter β in Eq (5.1). There exists a critical value β_{cr} of the scaling exponent for the variance $\sigma = N^{-\beta}$ which separates the areas of contraction and non-contraction mappings.
- We observe the following universality property: $\beta_{cr} \approx 1/2$ for all odd activation functions φ.

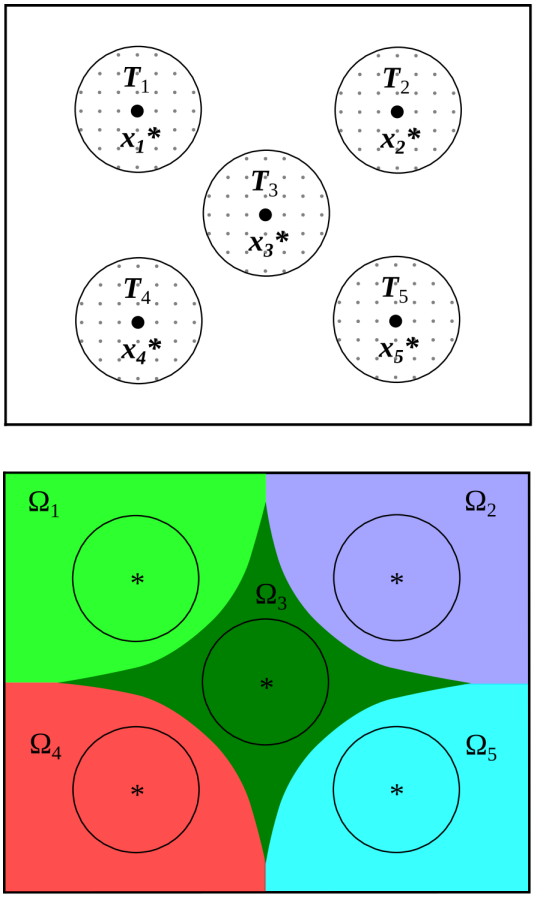

Figure 5. The simplified model of employee's photo encoding/decoding. The upper part corresponds to the untrained DNN. Each employee's photo is represented by point in the corresponding circle. The number of circles is the number of employers ($K = 5$). The solid black circles $\boldsymbol{x}_k^*$ correspond to the "true" photos. The lower part is the result of the numerical calculation for trained DNN. The asterisks are the FPs. We see numerically that the positions of the FP "*" coincide with $\boldsymbol{x}_k^*$. The filled areas are the corresponding basins of attraction Ω_k.

7. Conclusions

We studied the relation between random distributions of weights and the properties of FPs in autoencoder DNNs. We first considered *untrained* DNNs with random initialization of weight matrices with a light-tailed probability distribution, e.g., Gaussian. For such DNNs, we show the existence of the unique FP for DNNs with an arbitrary depth L (the number of layers) and an arbitrary width N (the size of square weight matrices), for a wide class of S-shaped odd activation functions. In the context of the image encoding/decoding problem, it means that for all images, there is a unique "true" image, i.e., the DNN cannot distinguish images. In contrast, for heavy-tailed DNNs (e.g., the Cauchy distribution), we show the existence of many FPs and, therefore, these DNNs are capable of identifying the "true" images. Our study showed the surprising nonmonotone dependence of the number of fixed points in the DNN, $Q(N_0, L)$, on the DNN's depth L (Figure 2).

Next, we studied the influence of the DNN architecture on the contraction property of the DNN function $\mathbf{\Phi}$ and, therefore, on the formation of FPs. We showed, that for light-tailed initialization, this property depends on the scaling exponent β in the variance $\sigma^2 = N^{-2\beta}$. Moreover, there exists the critical value β_{cr}, such that if $\beta \geq \beta_{cr}$, then the function $\mathbf{\Phi}$ is a contraction of *all* input vectors. Then Banach's FP theorem yields existence of the unique FP. Moreover, our simulations show that this FP is stable.

For heavy-tailed initialization of the weights, the contraction property of $\mathbf{\Phi}$ depends on the input vectors. This leads to existence of several FPs, so the set of input vectors is partitioned into basins of attractions corresponding to each FP.

Finally, we studied the properties of *trained* DNNs. Because of the self-regularization phenomenon [29], training leads to the formation of heavy-tailed distribution of the weights for any distributions at initialization. Hence, our results on the heavy-tailed distribution for *untrained* DNNs imply the *emergence of a number of FPs in the course of training*. We trained an autoencoder DNN for a simple case of two-dimensional input/output vectors (see Figure 5) and observed the formation of several FPs.

In conclusion, we note that our results can be useful in the practical design of autoencoders, because we provide the quantitative dependence of the number of FPs on the DNN's architecture, e.g., the number of layers, L. In particular, the non-monotone dependence of the number of FPs $Q(N_0, L)$ suggests the optimal autoencoder architecture.

Use of AI tools declaration

The authors did not use Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The work of L.B. and O.K. was partially supported by the National Science Foundation (NSF) Grant IMPRESS-U: N2401227. The work of V.S. was supported by the grant "International Multilateral Partnerships for Resilient Education and Science System in Ukraine" IMPRESS-U: N7114 funded by the US National Academy of Science and Office of Naval Research-Global. The authors are grateful to Mikhail Genkin, Vladimir Itskov, Tim Laux, and Ievgenii Afanasiev for many discussions and useful suggestions. The authors are also grateful to Zelong Li help with the numerical example in Section 6 and for the help in creating Figure 5.

Conflict of interest

L. Berlyand is an editorial board member for *Networks and Heterogeneous Media* and was not involved in the editorial review or the decision to publish this article. All authors declare that they have no competing interests.

Author contributions

All authors contributed equally to this work.

References

1. Y. Yan, S. Yang, Y. Wang, J. Zhao, F. Shen, Review neural networks about image transformation based on IGC learning framework with annotated information, preprint, arXiv:2206.10155, 2022. https://doi.org/10.48550/arXiv.2206.10155
2. S. Kaji, S. Kida, Overview of image-to-image translation by use of deep neural networks: Denoising, super-resolution, modality conversion, and reconstruction in medical imaging, preprint, arXiv:1905.08603, 2019. https://doi.org/10.48550/arXiv.1905.08603
3. W. Hong, T. Chen, M. Lu, S. Pu, Z. Ma, Efficient neural image decoding via fixed-point inference, in *IEEE Transactions on Circuits and Systems for Video Technology*, **31** (2021), 3618–3630. https://doi.org/10.1109/TCSVT.2020.3040367
4. C. Mou, Q. Wang, J. Zhang, Deep generalized unfolding networks for image restoration, preprint, arXiv:2204.13348, 2022. https://doi.org/10.48550/arXiv.2204.13348
5. D. Ferster, K. D. Miller, Neural mechanisms of orientation selectivity in the visual cortex, *Annual Rev. Neurosci.*, **23** (2000), 441–471. https://doi.org/10.1146/annurev.neuro.23.1.441
6. H. Ozeki, I. M. Finn, E. S. Schaffer, K. D. Miller, D. Ferster, Inhibitory stabilization of the cortical network underlies visual surround suppression, *Neuron*, **62** (2009), 578–592. https://doi.org/10.1016/j.neuron.2009.03.028
7. D. B. Rubin, S. D. Van Hooser, K. D. Miller, The stabilized supralinear network: A unifying circuit motif underlying multi-input integration in sensory cortex, *Neuron*, **85** (2015), 402–417. https://doi.org/10.1016/j.neuron.2014.12.026
8. C. Ebsch, R. Rosenbaum, Imbalanced amplification: A mechanism of amplification and suppression from local imbalance of excitation and inhibition in cortical circuits, *PLoS Comput. Biol.*, **14** (2018), e1006048. https://doi.org/10.1371/journal.pcbi.1006048
9. C. Curto, J. Geneson, K. Morrison, Fixed points of competitive threshold-linear networks. *Neural Comput.*, **31** (2019), 94–155. https://doi.org/10.1162/neco_a_01151
10. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, *Proc. Natl. Acad. Sci.*, **79** (1982), 2554. https://doi.org/10.1073/pnas.79.8.2554
11. J. Hopfield, Neurons with graded response have collective computational properties like those of two-state neurons, *Proc. Natl. Acad. Sci.*, **81** (1984), 3088–3092. https://doi.org/10.1073/pnas.81.10.3088
12. D. Krotov, J. Hopfield, Large associative memory problem in neurobiology and machine learning, preprint, arXiv:2008.06996, 2020. https://doi.org/10.48550/arXiv.2008.06996
13. T. Kimura, K. Kato, Analysis of discrete modern hopfield networks in open quantum system, preprint, arXiv:2411.02883, 2024. https://doi.org/10.1103/3966-8xs2
14. L. Berlyand, P. E. Jabin, *Mathematics of Deep Learning: An Introduction*, Walter de Gruyter GmbH & Co KG, (2023), 132. https://doi.org/10.1515/9783111025551
15. I. Goodfellow, Y. Bengio, A. Courville, Deep learning, in *Adaptive Computation and Machine Learning Series*, MIT Press, (2016), 785. Available from: http://www.deeplearningbook.org.

16. D. P. Kingma, M. Welling, An introduction to variational autoencoders, preprint, arXiv:1906.02691v3, 2019. https://doi.org/10.1561/2200000056

17. J. Wang, R. Cao, N. J. Brandmeir, X. Li, S. Wang, Face identity coding in the deep neural network and primate brain, *Commun. Biol.*, **5** (2022), 611. https://doi.org/10.1038/s42003-022-03557-9

18. P. Baldi, K. Hornik, Neural networks and principal component analysis: Learning from examples without local minima, *Neural Networks*, **2** (1989), 53–58. https://doi.org/10.1016/0893-6080(89)90014-2

19. T. J. Piotrowski, R. L. G. Cavalcante, M. Gabor, Fixed points of nonnegative neural networks, preprint, arXiv:2106.16239v9, 2021. https://doi.org/10.48550/arXiv.2106.16239

20. N. Buduma, N. Buduma, J. Papa, *Fundamentals of Deep Learning*, O'Reilly Media, Inc., 2nd edition, 2022, 387. Available from: https://www.oreilly.com/library/view/fundamentals-of-deep/9781492082170/.

21. Y. Bahri, J. Kadmon, J. Pennington, S. Schoenholz, J. Sohl-Dickstein, S. Ganguli, Statistical mechanics of deep learning, *Annu. Rev. Condens. Matter Phys.*, **11** (2020), 501–528. https://doi.org/10.1146/annurev-conmatphys-031119-050745

22. C. Gallicchio, S. Scardapane, Deep randomized neural networks, in *Recent Trends in Learning From Data. Studies in Computational Intelligence*, (eds. L. Oneto, N. Navarin, A. Sperduti, and D. Anguita), Springer, Cham, **896** (2020). https://doi.org/10.1007/978-3-030-43883-8_3

23. R. Giryes, G. Sapiro, A. M. Bronstein, Deep neural networks with random Gaussian weights: A universal classification strategy, *IEEE Trans. Signal Process.* **64** (2016), 3444–3457. https://doi.org/10.1109/TSP.2016.2546221

24. Z. Ling, X. He, R. C. Qiu, Spectrum concentration in deep residual learning: A free probability approach, preprint, arXiv:1807.11694, 2018. https://doi.org/10.48550/arXiv.1807.11694

25. A. G. de G. Matthews, J. Hron, M. Rowland, R. E. Turner, Z. Ghahramani, Gaussian process behaviour in wide deep neural networks, preprint, arXiv:1804.11271, 2018. https://doi.org/10.48550/arXiv.1804.11271

26. G. Yang, Tensor programs III: Neural matrix laws, preprint, arXiv:2009.10685v1, 2020. https://doi.org/10.48550/arXiv.2009.10685

27. V. Marchenko, L. Pastur, The eigenvalue distribution in some ensembles of random matrices, *Math. USSR Sbornik*, **1** (1967), 457–483. https://doi.org/10.1070/SM1967v001n04ABEH001994

28. L. Berlyand, E. Sandier, Y. Shmalo, L. Zhang, Enhancing accuracy in deep learning using random matrix theory, *J. Mach. Learn.*, **3** (2024), 347–412. https://doi.org/10.4208/jml.231220

29. C. H. Martin, M. W. Mahoney, Implicit self-regularization in deep neural networks: Evidence from random matrix theory and implications for learning, *J. Mach. Learn. Res.*, **22** (2021), 1–73. https://doi.org/10.5555/3546258.3546423

30. J. Pennington, S. S. Schoenholz, S. Ganguli, The emergence of spectral universality in deep networks, preprint, arXiv:1802.09979v1, 2018. https://doi.org/10.48550/arXiv.1802.09979

31. N. Belrose, A. Scherlis, Understanding gradient descent through the training Jacobian, preprint, arXiv:2412.07003v2, 2024. https://doi.org/10.48550/arXiv.2412.07003

32. L. Pastur, V. Slavin, On random matrices arising in deep neural networks: General I.I.D. case, *Random Matrices: Theory Appl.*, **12** (2023), 2250046. https://doi.org/10.1142/s2010326322500460

33. J. Hoffman, D. A. Roberts, S. Yaida, Robust learning with Jacobian regularization, preprint, arXiv:1908.02729v1, 2019. https://doi.org/10.48550/arXiv.1908.02729

34. C. Bessaga, On the converse of the Banach "fixed-point principle", *Colloq. Math.*, **7** (1959), 41–43. https://doi.org/10.4064/cm-7-1-41-43

35. J. Jachymski, I. Jóźwik, M. Terepeta, The Banach fixed point theorem: Selected topics from its hundred-year history, *Rev. Real Acad. Cienc. Exactas Fis. Nat. Ser. A-Mat.*, **118** (2024), 140. https://doi.org/10.1007/s13398-024-01636-6

36. Y. Shmalo, J. Jenkins, O. Krupchytskyi, Deep learning weight pruning with RMT-SVD: Increasing accuracy and reducing overfitting, preprint, arXiv:2303.08986v1, 2023. https://doi.org/10.48550/arXiv.2303.08986

37. T. Shcherbina, On universality of local edge regime for the deformed Gaussian unitary ensemble, *J. Stat. Phys.*, **143** (2011), 455–481. https://doi.org/10.1007/s10955-011-0196-9

38. J. G. Russo, Deformed Cauchy random matrix ensembles and large N phase transitions, *J. High Energy Phys.*, **14** (2020), 1. https://doi.org/10.1007/JHEP11(2020)014

39. M. Hisakado, T. Kaneko, Deformation of Marchenko–Pastur distribution for the correlated time series, preprint, arXiv:2305.12632v2, 2023. https://doi.org/10.48550/arXiv.2305.12632

AIMS Press