Neural Computing and Applications (2021) 33:3385–3401
https://doi.org/10.1007/s00521-020-05182-1

ORIGINAL ARTICLE

"SPOCU": scaled polynomial constant unit activation function

Jozef Kiseľák[1,2] · Ying Lu[3] · Ján Švihra[4] · Peter Szépe[5] · Milan Stehlík[2,6,7]

Received: 15 February 2019 / Accepted: 11 July 2020 / Published online: 25 July 2020

Abstract
We address the following problem: given a set of complex images or a large database, the numerical and computational complexity and quality of approximation for neural network may drastically differ from one activation function to another. A general novel methodology, scaled polynomial constant unit activation function "SPOCU," is introduced and shown to work satisfactorily on a variety of problems. Moreover, we show that SPOCU can overcome already introduced activation functions with good properties, e.g., SELU and ReLU, on generic problems. In order to explain the good properties of SPOCU, we provide several theoretical and practical motivations, including tissue growth model and memristive cellular nonlinear networks. We also provide estimation strategy for SPOCU parameters and its relation to generation of random type of Sierpinski carpet, related to the [*pppq*] model. One of the attractive properties of SPOCU is its genuine normalization of the output of layers. We illustrate SPOCU methodology on cancer discrimination, including mammary and prostate cancer and data from Wisconsin Diagnostic Breast Cancer dataset. Moreover, we compared SPOCU with SELU and ReLU on large dataset MNIST, which justifies usefulness of SPOCU by its very good performance.

Keywords Activation function · SPOCU · SELU · ReLU · Cancer discrimination · Percolation

Mathematics Subject Classification 60K35 · 82B43 · 92B20

1 Introduction

Neural computing and activations in neural networks are multidisciplinary topics, which range from neuroscience to theoretical statistical physics. Also, importantly enough, one of the most important theoretical and practical topics in neural networks and neural computing is choice of the appropriate activation function. Activation function and its nonlinear ability represent core of the neural networks, both deep and shallow, with various architectures. In standard problems with a reasonable nonlinearity, the sigmoidal function is well justified; however, one faces the inverse problem of specification of the underlying

✉ Milan Stehlík
Milan.Stehlik@jku.at

Jozef Kiseľák
jozef.kiselak@upjs.sk

Ying Lu
ylu1@stanford.edu

Ján Švihra
svihra@jfmed.uniba.sk

Peter Szépe
peter.szepe@atlas.sk

[1] Institute of Mathematics, Faculty of Science, P.J.Šafárik University in Košice, Kosice, Slovak Republic

[2] Linz Institute of Technology (LIT) and Department of Applied Statistics, Johannes Kepler University in Linz, Linz, Austria

[3] School of Medicine, Stanford University, Stanford, USA

[4] Department of Urology, Jessenius Faculty of Medicine, Comenius University Bratislava, 03659 Martin, Slovak Republic

[5] Department of Pathological Anatomy, University Hospital Martin, Kollárova 2, 036 59 Martin, Slovak Republic

[6] Department of Statistics, University of Valparaíso, Valparaíso, Chile

[7] Department of Statistics and Actuarial Science, The University of Iowa, Iowa City, USA

parameters; otherwise, network may perform slowly. Recently, several papers related to making a more adequate activation function have appeared, mainly comparing activation functions on large datasets. To illustrate some of important contributions, the influence of the activation function in the convolutional neural network (CNN) model is studied in [24], improving a ReLU activation by construction of a novel surrogate. Theoretical analysis about gradient instability as well as the fundamental explanation for the exploding/vanishing gradient and the performances of different activation functions are given in [13].

The main contribution of this paper is the construction and testing of novel scaled polynomial constant unit (SPOCU) activation function. Such a novel activation function relates to complexity patterns through phenomenon of percolation, and thus, it can overcome already introduced activation functions, e.g., SELU and ReLU. In statistical physics and mathematics, percolation theory describes the behavior of a network when nodes or links are removed, or complex patterns are embedded to learning process. Thus, SPOCU well contributes to fill the gap in the theories and it is "picking up" the appropriate properties for activation function directly from training of classification on complex patterns, e.g., cancer images. Such efforts can contribute to many applications. One example is the generation of the artificial networks applied in the field of the Touring pattern generators networks, since they require the two-layer coupling (two "diffusion" mechanisms having different diffusion coefficients) of resistor couplings. Turing patterns are nonlinear phenomena which appear in the reaction–diffusion systems based on the memristive cell. For Turing patterns in the simplest memristive cellular nonlinear networks (MCNNs), see [2], where it is proposed a new MCNN model consisting in a two-dimensional array of cells made by only two components: a linear passive capacitor and a nonlinear active memristor. In fact, cellular nonlinear networks are perfectly suited to fit the structure of reaction–diffusion systems, since they can be used to map partial differential equations [8]. MCNNs are intrinsically related to percolation; link to this relation is outlined in Sect. 3; for more, see, e.g., Chapter 18 in [6].

The paper is organized as follows. In Sect. 2.1, we introduce random fractal construction useful for our purposes. We give an overview of the problem of generating of random type of Sierpiński carpet, related to the [*pppq*] model introduced by [9]. We emphasize importance of using Kronecker product, which gives a simple and quick possibility to generate random fractals. We also present several useful results concerning fractal geometry and "average" fractal dimension. In Sect. 3, we deal with percolation threshold, which is an important instrument for SPOCU. By using of direct approach and logistic regression, we derive estimations for this critical value in the case of specific random models. In Sect. 4, we use basic generator from random Sierpiński carpet as a new activation function SPOCU for self-normalizing neural network. We study general activation functions, but with main focus on SPOCU. As normalizing the output of layers is known to be a very efficient way to improve the performance of neural networks, we can conclude that SPOCU activation function behaves very well. Further we provide theoretical justification of the fact that SPOCU outperforms both SELU and ReLU in several desirable properties, necessary for the correct classification of complex images. The SPOCU also gets uniformly better performance with respect to generic properties on large MNIST database. This is well illustrated in the last Sect. 6 on image-based discrimination for cancer diagnostics. Therein we apply the developed methodology to cancer discrimination problems; namely, we consider image-based discrimination for mammary cancer versus mastopathy and benign prostatic hyperplasia or normal prostate versus prostate cancer. A comparison of SPOCU, ReLU and SELU on the Wisconsin Diagnostic Breast Cancer (WDBC) dataset justifies SPOCU qualities. Technicalities and proofs are given in "Appendix A."

2 Random fractals, Kronecker product and fractal dimension

Many of fractal constructs have random analogues; see Chapter 15 in [6]. Here, we start with the natural motivation of the random Sierpiński carpet (SC). We introduce simple computation of matrix expressing the form of fractals, including random SC.

2.1 Random Sierpiński carpet

The Sierpiński carpet (SC) is the fractal, which can be constructed by taking square $[0,1]^2$, dividing it into nine equal squares of side length 1/3 and removing the central square. This procedure is then repeated for each of the eight remaining squares and iterated infinitely many times. The carpet is the resulting fractal and has Hausdorff dimension $d_f = \ln 8/\ln 3$. We can equivalently construct this fractal by using string rewriting beginning with a cell 1 and iterating the rules

$$\left\{ 0 \rightarrow \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, 1 \rightarrow \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right\}.$$

Here, cell 0 indicates the absence of generated square and cell 1 indicates its presence. Its random variation (random SC) can be defined by using of iterating the rules

$$\left\{0 \to \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, 1 \to \begin{bmatrix} \mathcal{B}_{1,p_n} & \mathcal{B}_{2,p_n} & \mathcal{B}_{3,p_n} \\ \mathcal{B}_{4,p_n} & 0 & \mathcal{B}_{5,p_n} \\ \mathcal{B}_{6,p_n} & \mathcal{B}_{7,p_n} & \mathcal{B}_{8,p_n} \end{bmatrix}\right\} \tag{1}$$

beginning again with a cell 1, whereas $\mathcal{B}_{i,p_n}, i = 1, \ldots, 8$ are mutually independent random variables generated in nth iteration by the uniformly distributed random variable $U_{i,n}$ and prescribed value $p_n \in [0, 1]$ as

$$\mathcal{B}_{i,p_n} := \begin{cases} 1, & \text{if } U_n > p_n, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, the random numbers result from a Bernoulli distribution by usage of the inversion method. Notice that $[pppq]$ model, [9], is included to such construction. This notation naturally generalizes it into $[p\ldots pq]$ model and also $[p_1\ldots p_{n-1}p_n]$ model.

Remark 2.1 Notice also that in [5, Example 1.1] the authors consider also a generalization of Mandelbrot's percolation process, but this approach is different from our (depending on the set of indices generating zeros and ones). With probability p, they generate zero 3×3 matrix or matrix with zero element in the middle with probability $1 - p$.

2.2 Fractals induced by Kronecker product

The Kronecker product (also direct matrix product) is a special case of the tensor product. It is denoted by $\otimes$ and is an operation on two matrices of arbitrary size resulting in a block matrix; see "Appendix A." The multiple Kronecker product is defined in a recursive fashion as $\bigotimes_{i=1}^{r} A_i := \left(\bigotimes_{i=1}^{r-1} A_i\right) \otimes A_r$, where $\bigotimes_{i=1}^{1} A_i = A_1$. Some fractals, see examples in "Appendix A.2," can be represented by multiple Kronecker product, [20, chap. 9]. In [19], two methods for generating images of (approximations to) fractals and fractal-like sets are given: iterated Kronecker products and iterated matrix-valued homomorphisms. For random alternative, we use the notation

$$\mathbf{M}_r^{p_1,\ldots,p_r} := \bigotimes_{i=1}^{r} \mathbf{X}_{p_i},$$

where r is natural number and

$$\mathbf{M}_r^{p_1,\ldots,p_\infty} := \bigotimes_{i=1}^{\infty} \mathbf{X}_{p_i},$$

if it is infinity and $\mathbf{X}_{p_i}$ means matrix involving values generated by $\mathcal{B}_{\cdot,p_i}$. (This can be naturally generalized for suitable random variable.) Obviously $\mathbf{M}_r^{p_1,\ldots,p_r}$ can be understood as an approximation of $\mathbf{M}_r^{p_1,\ldots,p_\infty}$. From a dynamical point of view, random fractal can be understood as follows

$$\mathbf{Y}_0 := 1 \tag{2}$$

$$\mathbf{Y}_n := \mathbf{Y}_{n-1} \bigotimes \mathbf{X}_{p_n}, n \geq 1. \tag{3}$$

Since we assume mutual independence, we have

$$\mathbb{E}[\mathbf{Y}_n] = \mathbb{E}\left[\mathbf{Y}_{n-1} \bigotimes \mathbf{X}_{p_n}\right] = \mathbb{E}[\mathbf{Y}_{n-1}] \bigotimes \mathbb{E}\left[\mathbf{X}_{p_n}\right] = \bigotimes_{j=0}^{n} \mathbf{E}\left[\mathbf{X}_{p_j}\right];$$

see [7]. We also define matrix $\mathbf{M}_r^p := \mathbf{M}_r^{p,\ldots,p}$, i.e., $\mathbf{M}_r^p = \bigotimes_{i=1}^{r} \mathbf{X}_{p_i}, p_i = p$ for all $i \in \{1, \ldots, r\}$.

Lemma in "Appendix A.5" gives us the mean value of retained elements since we know that in the deterministic case the SC has exactly 2^{3n} elements in $n-$th iteration. Therefore, the joint pdf of elements of $\mathbf{Y}_n$ is $2^{3n}\tilde{p}_n$ and we have $N_n = 2^{3n}\tilde{p}_n$ retained elements in average, which reduces to $(8p)^n$ for case $p_j = p$. From that, we see the balance threshold $p_b < \frac{1}{8}$ for which convergence to zero in average is obtained. (Similar argumentation can be done for modified SC with 9 random elements, and in the results, 2^{3n} has to be replaced by 3^{3n}.)

2.3 Geometry and dimension

Now we describe the model geometrically, similarly as [3]. Let $A_0 = [0, 1]^2$, and for $1 \leq i, j \leq 3$, let $B_{ij} = \left[\frac{i-1}{3}, \frac{i}{3}\right] \times \left[\frac{j-1}{3}, \frac{j}{3}\right]$. Moreover, we let

$$A_1 = \bigcup_{\substack{i,j \\ Y_1^{ij} = 1}} B_{ij}.$$

To define A_2, we repeat the last construction. More generally, we let $B_{ij}^n = \left[\frac{i-1}{3^n}, \frac{i}{3^n}\right] \times \left[\frac{j-1}{3^n}, \frac{j}{3^n}\right]$ for $1 \leq i, j \leq 3^n$ and

$$A_n = \bigcup_{\substack{i,j \\ Y_n^{ij} = 1}} B_{ij}^n.$$

Clearly $\{A_k\}_{k\in\mathbb{N}}$ is a decreasing sequence of compact sets so the limit $A_\infty = \bigcap_{n\in\mathbb{N}} A_n$. The next theorem follows directly from the average number of retained elements N_n and the fact that we obtain a branching process in which each particle has on the average $8p$ offspring.

Theorem 2.2 *If p_j does not change in time ($p_j = p$), then $A_\infty \neq \emptyset$ with positive probability if and only if $p > \frac{1}{8}$.*

This is closely related to the estimation of fractal dimension; see [27]. Here, we deal with "average" fractal dimension or fractal dimension in mean. We already have N_n the average count of retained elements in A_n and denote as s_n the scale in nth step, i.e., in our case $s_n = 3^n$. Now using least square fitting on data $\{\ln s_i, \ln N_i\}$, we obtain the slope of the curve

$$\mathrm{l}_n = \frac{n\sum_{i=1}^{n}(\ln s_i \ln N_i) - \sum_{i=1}^{n}\ln s_i \sum_{i=1}^{n}\ln N_i}{n\sum_{i=1}^{n}\ln^2 s_i - \left(\sum_{i=1}^{n}\ln s_i\right)^2}, \qquad (4)$$

whereas $\max\{0, \mathrm{sl}_n\}$ is the nth approximation of fractal dimension.

Theorem 2.3 *If $A_\infty \neq \emptyset$, then*

$$\dim_B(A_\infty) = \dim_H(A_\infty) = \max \left\{ 0, \frac{\ln 8}{\ln 3} + \frac{6}{\ln 3}\lim_{n\to\infty} \frac{\sum_{i=1}^{n}\left((2i-n-1)\sum_{j=1}^{i}\ln p_j\right)}{n(n-1)(n+1)} \right\}.$$

For constant case $p_j = p$, we have $P(i) = ip$, and thus, $\dim_B(A_\infty) = \dim_H(A_\infty) = \max\left\{\frac{\ln(8p)}{\ln 3}, 0\right\}$; see also Fig. 2. Naturally for $p = 1$, this is in coincidence with the standard (deterministic) Sierpiński carpet. See Fig. 1 where the results of $[p_1p_2p_3p_4]$ model are plotted. Now denote as $\mathbf{p}$ sequence (vector) of probabilities $\{p_j\}_{j\in\mathbb{N}}$. Notice that it need not to be a probability vector since we do not require $\sum_{j=1}^{\infty} p_j = 1$. Consider now the case "Appendix A.4," then

$$\dim_H(A_\infty) = \max\left\{\frac{\ln(8\sqrt{PQ})}{\ln 3}, 0\right\}.$$

Fig. 2 plots the condition $PQ > \frac{1}{64}$,, i.e., a hyperbola, which determines whether A_∞ has a positive measure. This means that if P is small, Q can "save" the situation.

Theorem 2.4 *For any vector of probabilities* $\mathbf{p}$

$$0 \le \dim_B(A_\infty) \le d_f = \frac{\ln 8}{\ln 3}.$$

Moreover, for every value from this interval there exist a vector of probabilities $\mathbf{p}$.

Examples in "Appendix A.3" show that constant vector and nonconstant vector may lead to a similar fractal dimension. Now we introduce assertion related to expected matrix.

Theorem 2.5

$$\|\mathbf{M}_r^{p_1,\ldots,p_r}\|_F = \sqrt{\prod_{i=1}^{r} \mathrm{tr}\left(\mathbf{X}_{p_i}^{*}\mathbf{X}_{p_i}\right)}.$$

For specific case of $\mathcal{B}_{j,p_i}$ and under the assumption of independence, we have

$$Z = \|\mathbf{M}_r^{p_1,\ldots,p_r}\|_F^2 = \prod_{i=1}^{r}\sum_{j=1}^{8}\mathcal{B}_{j,p_i}^2 = \prod_{i=1}^{r}\sum_{j=1}^{8}\mathcal{B}_{j,p_i} = \prod_{i=1}^{r}\mathrm{Bin}(8, p_i).$$

Therefore, $\mathbb{E}[Z] = \prod_{i=1}^{r} 8p_i$,; in particular, it is $(8p)^r$ for $p_i = p$ (which means that for at least small p_i the probability of Z value is close to zero). We have estimated this probability $\mathcal{P}_{all}$ of event that matrix $\mathbf{M}_r^p$ is zero matrix (empty) and probability $\mathcal{P}_{col}$ of event that at least one column of matrix $\mathbf{M}_r^p$ is zero. Probability $\mathcal{P}_{all}$, i.e., that Z is zero value is given by inclusion–exclusion principle. Both are specific percolations, especially the latter, since we can reach the bottom. The results are summarized in Table 6.

3 Percolation threshold

Percolation itself is a physically well-motivated phenomenon and it is intrinsically related to memristor models (see [22]); thus, it can be well involved in neural modeling of memristive cellular nonlinear networks (MCNNs). The combination of percolation theory and Monte Carlo simulation provides a possible solution to model the ion migration and electron transport in an amorphous system even from the hardware perspective. The most known mathematical percolation model is to take some regular lattice, e.g., a square lattice $N \times N$, and make it into a random network by randomly "occupying" sites (vertices) or bonds (edges) with a statistically independent random variables. For example, each of the lattice sites is either occupied (with probability p) or vacant (with probability $1-p$). This is commonly known as the site percolation problem. There exists also a related bond percolation problem. It can be posed in terms of whether or not the edges between neighboring sites are open or closed. It is

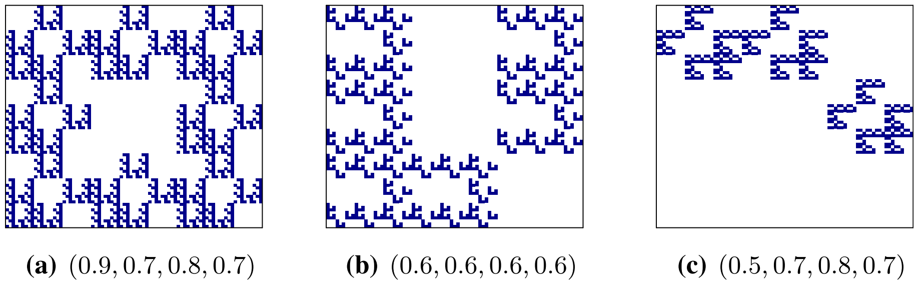
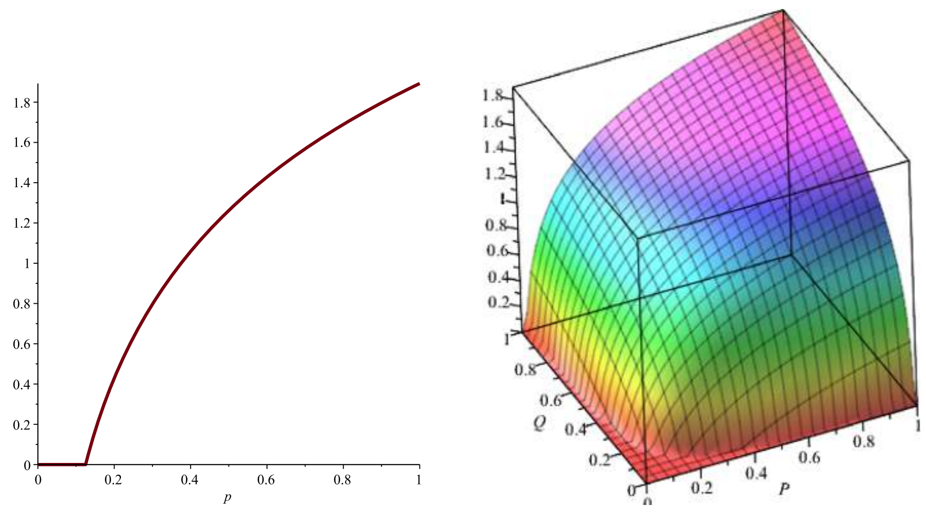

(a) $(0.9, 0.7, 0.8, 0.7)$ **(b)** $(0.6, 0.6, 0.6, 0.6)$ **(c)** $(0.5, 0.7, 0.8, 0.7)$

Fig. 1 Random SC for $[p_1p_2p_3p_4]$ model

Fig. 2 Fractal dimension of A_∞



(a) $\dim_H(A_\infty)$ depending on p. **(b)** $\dim_H(A_\infty)$ depending on P and Q.

well known that there is a well-defined range of p for which the probability of percolation decreases rapidly from one to zero. It is centered on a critical value p_c called percolation threshold. Percolation clusters become self-similar precisely at the threshold density p_c for sufficiently large length scales, entailing the following asymptotic power laws: $M(L) \sim L^{df}$ at $p = p_c$ and for large probe sizes, $L \to \infty$, i.e., fractal dimension d_f (e.g., Hausdorff dimension $\dim_H$) describes the scaling of the mass M of a critical cluster within a distance L (it characterizes how the mass $M(L)$ changes with the linear size L of the system); see [21]. This follows from the following idea: If we consider a smaller part of a system of linear size $bL(b<1)$, then $M(bL)$ is decreased by a factor of b^d, i.e., $M(bL) = b^d M(L)$. For example, for Sierpinski Gasket we have functional equation $M(L/2) = M(L)/3$ yielding solution $M(L) = AL_f^d$ with $d_f = \ln(3)/\ln(2)$.

Depending on the method for obtaining the random network, usually one distinguishes between the site percolation threshold and the bond percolation threshold. More general systems may have several probabilities p_1, p_2, etc., and the transition is characterized by a critical surface or a manifold. In the classical systems, it is assumed that the occupation of a site or bond is completely random. This is the so-called Bernoulli percolation. Here, we want to emphasize that random SC does not fall under them. In 1974, Mandelbrot [14] introduced a process in $[0, 1]^2$ which he called “canonical curdling.” It is nothing else than the model from Remark “Appendix A.1” with $\mathcal{B}_{p_{ij,n}} = \mathcal{B}_p$. In paper [3], the authors study the connectivity or “percolation” properties of such sets. They showed there is a probability $p_c \in (0, 1)$ so that if $p < p_c$ then the set is “dustlike,” whereas if $p \geq p_c$ opposing sides are connected with positive probability. To be precise, we introduce here the exact definition of p_c. Let

$$B_n = \{x \in A_n : x \text{can be connected to} [0, 1] \times \{0\} \text{and} [0, 1] \times \{1\} \text{by paths in} A_n\},$$

$B_\infty = \bigcap_{n\in\mathbb{N}} B_n$ and let $\Omega = \{B_\infty \neq \emptyset\}$. Notice that when Ω occurs there is a up-to-down crossing of $[0, 1]^2$. Finally, $p_c = \inf_p\{\mathbb{P}(\Omega)\}$. For example, from [3] we have that Mandelbrot percolation is $p_c < 0.9999$. Here, we have to emphasize that for approximation of A_∞ we cannot use this kind of definition. We just set p_c as the $\frac{1}{2}$ threshold of probability $\mathbb{P}(B_n = \emptyset)$.

3.1 Estimation of a single parameter *p*

Here, we consider $[p\ldots p]$ model. We have used the flow through the generated lattice (represented by the matrix) and use a recursive depth first search (checking whether or not the flow makes it to the bottom of the grid). The graphs of a simulated data possess a sigmoidal shape which signals the presence of a threshold. Moreover, we have categorical dependent variable represented by the outcomes pass/fail. These facts indicate that in order to determine the threshold, we fit a logistic model (log-odds)

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x,$$

to simulated data and the threshold value for p (or $1 - p$) is estimated such that $p = \frac{1}{2}$ implies $0 = \beta_0 + \beta_1 x$, and therefore, estimation of p_c equals $-\frac{\beta_0}{\beta_1}$, i.e., minus quotient of the intercept and the regression coefficient. Notice that similarly the authors in [1] used logistic function to find threshold in order to examine the effects of mixed dispersal strategies on the spatial structure of a population considering a spatially explicit birth–death model. A discussion on how different sigmoidal models can be applied to predict the percolation threshold of electrical conductivity for ethylene vinyl acetate (EVA) copolymer and acrylonitrile butadiene (NBR) copolymer conducting composite systems filled with different carbon fillers is given in [17]. On the other hand, an experiment using the phenomenon of percolation has been conducted to demonstrate the implementation of neural functionality in [16], where the curve was found to be almost exactly described by the sigmoid form.

In Table 1, we plot the results for specific r, see also Figs. 3 and 4, where the simulations of percolation probability for given p are shown. The logistic model seems to fit the curve very well. However, for $r = 1$ (important generator case) we can explicitly find the analytic form of the model function. By going through all the possibilities, we can directly find the polynomial

$$(1-p)^8+8\,(1-p)^7p+20\,(1-p)^6p^2+20\,p^3(1-p)^5 \\ +10\,p^4(1-p)^4+2\,p^5(1-p)^3,$$

which can be simplified to

$$\mathcal{P}(p) = (1-p)^3\left[(1-p)^5 - 2(1-p)^4 + 2\right]. \tag{5}$$

The threshold value $\mathcal{P}(p) = \frac{1}{2}$ is given approximately as 0.341. See Fig. 3 for excellent fitting and notice that logit model fits very closely with the value 0.3498.

3.2 Estimation of two parameters p and q

Here, we assume two parameters p and q, e.g., we consider $[pq]$ or $[pppq]$ model. A binomial logistic model

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x + \beta_2 y \tag{6}$$

is again fitted to simulated data and the threshold values for p (or $1-p$) and q are estimated such that $p = \frac{1}{2}$ implies relationship $0 = \beta_0 + \beta_1 p_c + \beta_2 q_c$, and therefore, estimation of p_c and q_c is given by this (part of) line. This is obviously a different result, since we have infinitely many solutions unless some suitable constraint $p = f(q)$ (such that the intersection of the constraint curve and this line is nonempty) is not given. Notice that even for a given constraint it can happen that more than one solution is achieved. If we, for example, assume that $f = \text{id}$, i.e., $p = q$, we obtain one-parameter estimation model from the previous section. Table 2 plots the results for constraint $p = 2q$; see also Figs. 5 and 6. For example, for $[pq]$ model we obtained that $b_0 = 6.621, b_1 = -5.725, b_2 = -5.455$, for $[ppq]$ model we obtained that $b_0 = 9.053, b_1 = -8.188, b_2 = -5.492$ and for $[pqq]$ model we obtained $b_0 = 9.159, b_1 = -5.798, b_2 = -7.970$. Here, we have to emphasize the difference between $[ppq]$ and $[pqq]$ where for the latter constraint $p = 2q$ is not suitable, since no solution exists.

Notice, however, that if we add a mixed term into regression

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x + \beta_2 y + \beta_3 xy, \tag{7}$$

then, e.g., for $[pqq]$ model we obtained that $b_0 = 4.827, b_1 = 1.241, b_2 = -1.734$ and $b_3 = -10.745$ which yields hyperbolic curve. For better graphical illustration of the difference (this difference is not obvious from standard point of view), see Fig. 7. From a qualitative point of view, there exists the difference, e.g., nonexistence of solution for the mixed case even if such a solution exists for the linear case.

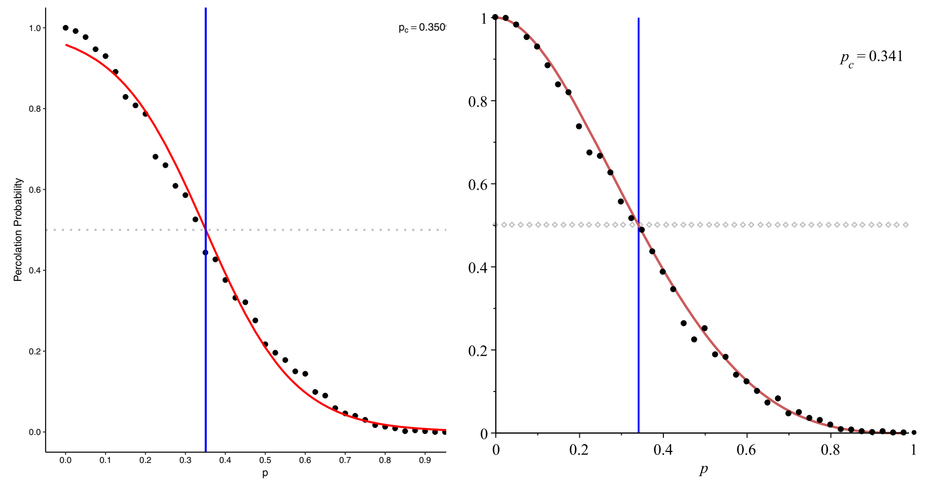
Table 1 Estimation of percolation threshold p_c for given r

r	1	2	3
est. p_c	0.3498	0.5246	0.6195

4 Self-normalizing neural network

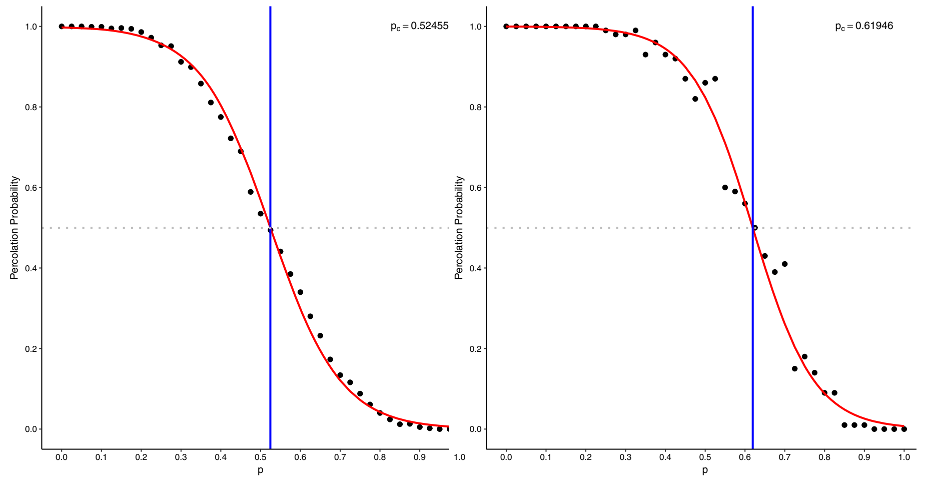
Self-normalizing neural networks (SNNs) are expected to be robust to perturbations and not to have high variances in their training errors. SNNs push neuron activations to zero mean and unit variance, thereby leading to the same effect as batch normalization, which enables to learn many layers in a robust way. Here, we introduce our SNNs based on percolation function (5).

For a neural network with activation function Ac, we consider two consecutive layers connected by a weight matrix $\mathbf{W}$. We assume that all activations x_i of the lower layer have the same mean $\mathbb{E}[x_i] = \mu$ and variance $\mathbb{V}[x_i] = \sigma^2$ and are mutually independent. A single activation $y = f(z)$ in the higher layer has network input $z = \mathbf{w}^T\mathbf{x}$, mean $\mathbb{E}[y] = \tilde{\mu}$ and variance $\mathbb{V}[y] = \tilde{\sigma}^2$. From this, we can obtain $\mathbb{E}[z] = \sum_{i=1}^n w_i \mathbb{E}[x_i] = \mu \sum_{i=1}^n w_i := \mu\,\omega$ and $\mathbb{V}[z] = \sum_{i=1}^n w_i^2 \mathbb{V}[x_i] = \sigma^2 \sum_{i=1}^n w_i := \sigma^2\,\tau^2$. Central limit theorem implies under the regularity conditions that $z \sim \mathcal{N}(\mu\,\omega, \sigma^2\tau^2)$, i.e., the pdf of z has the form $f(z) = \frac{1}{\sqrt{2\pi\sigma^2\tau^2}} e^{-\frac{(z-\mu\omega)^2}{2\sigma^2\tau^2}}$. Consider now vector mapping $\mathbf{g}$ that maps mean and variance of the activations from one layer to mean and variance of the activations in the next



(a) Estimated logit model. **(b)** Exact polynomial function.

Fig. 3 Percolation probability fitted to simulated data, $r = 1$



(a) $r = 2$ **(b)** $r = 3$

Fig. 4 Percolation probability as a function of parameter p

Table 2 Estimation of percolation threshold p_c and q_c for a given model with constraint $p = 2q$

Model	$[pq]$	$[ppq]$	$[pqq]$	$[pqq]mix$
est. p_c	0.7833	0.828	0.9362	0.9833
est. q_c	0.3917	0.4140	0.4681	0.4917

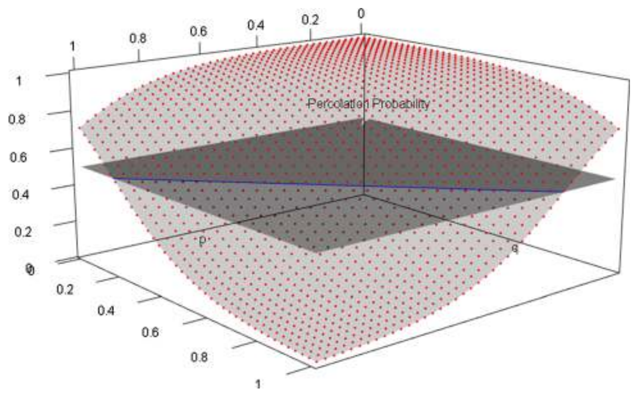


Fig. 5 Percolation probability for $[pq]$ model

layer, i.e., $g_1(\mu, \sigma^2) = \tilde{\mu}$ and $g_2(\mu, \sigma^2) = \tilde{\sigma}^2$. The following definition recalls a self-normalizing neural network.

Definition 4.1 (*Self-normalizing neural net*) ([11]) We say that a neural network is self-normalizing if it possesses a mapping $\mathbf{g} : \Omega \rightarrow \Omega$ for each activation y that maps mean and variance from one layer to the next and has a stable and attracting fixed point depending on ω and τ^2 in $\Omega := [\mu_{min}, \mu_{max}] \times [\sigma^2_{min}, \sigma^2_{max}]$. Furthermore, $\mathbf{g}(\Omega) \subseteq \Omega$. When iteratively applying the mapping $\mathbf{g}$, each point within converges to this fixed point.

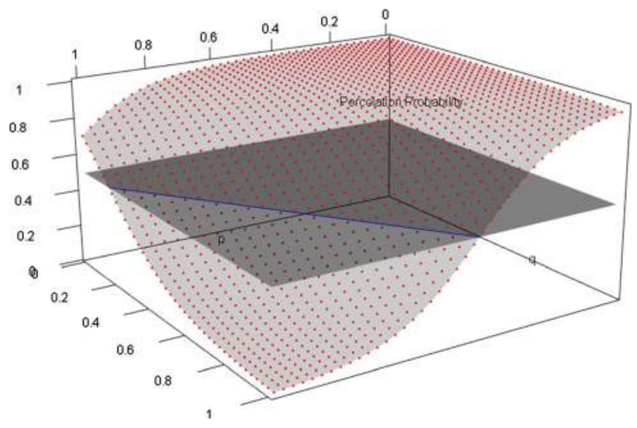


Fig. 6 Percolation probability for $[ppq]$ model

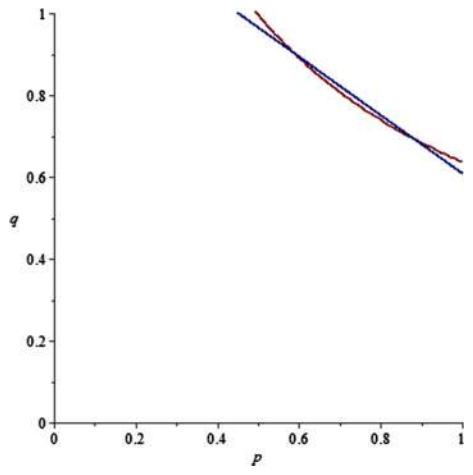


Fig. 7 Relationship between p and q for logit (6) and (7), respectively

For arbitrary activation function $Ac(z)$, the mapping $\mathbf{g}$ is given by the relations

$$\tilde{\mu}(\mu, \omega, \sigma^2, \tau^2, \mathbf{a}) = \mathbb{E}_f[Ac(z; \mathbf{a})] \tag{8}$$

and

$$\tilde{\sigma}^2(\mu, \omega, \sigma^2, \tau^2, \mathbf{a}) = \mathbb{E}_f[Ac^2(z; \mathbf{a})] - \tilde{\mu}^{\cdot} \tag{9}$$

Obviously moments are given by integrals $\mathbb{E}_f[Ac(z; \mathbf{a})] = \int_{\mathbb{R}} f(z)\, Ac(z; \mathbf{a})\, dz$ and $\mathbb{E}_f[Ac^2(z; \mathbf{a})] = \int_{\mathbb{R}} f(z)\, Ac^2(z; \mathbf{a})\, dz$. [11] proposed $\omega = 0$ and $\tau^2 = 1$ for all units in the higher layer for the weight initialization. If the Jacobian of $\mathbf{g}$ has a norm smaller than 1 at the fixed point, then it is a contraction mapping and the fixed point is stable. The goal is to find parameters $\mathbf{a} \in A$ such that fixed point [0, 1] does exist and that $||\mathbf{J_f}[0, 1]|| < 1$. This problem is formulated as to find $\mathbf{a} \in A$ such that

$$\begin{aligned} 0 &= \mathbb{E}_{f(z;0,1)}[Ac(z; \mathbf{a})], \\ 1 &= \mathbb{E}_{f(z;0,1)}[Ac^2(z; \mathbf{a})], \end{aligned} \tag{10}$$

$||\mathbf{J}|| < 1$, where (here we omit parameters for the sake of short notation)

$$\begin{aligned} J_{11} &= \mathbb{E}_f[zAc(z)], \\ J_{12} &= \mathbb{E}_f\left[\frac{z^2 - 1}{2} Ac(z)\right], \\ J_{21} &= \mathbb{E}_f[zAc^2(z)] - 2\mathbb{E}_f[Ac(z)]\mathbb{E}_f[zAc(z)], \\ J_{22} &= \mathbb{E}_f\left[\frac{z^2 - 1}{2} Ac^2(z)\right] - 2\mathbb{E}_f[Ac(z)]\mathbb{E}_f\left[\frac{z^2 - 1}{2} Ac(z)\right], \end{aligned}$$

which can be simplified to

$$\mathcal{J} = \begin{bmatrix} \mathbb{E}_f[zAc(z)] & \mathbb{E}_f\left[\frac{z^2 - 1}{2} Ac(z)\right] \\ \mathbb{E}_f[zAc^2(z)] & \mathbb{E}_f\left[\frac{z^2 - 1}{2} Ac^2(z)\right] \end{bmatrix}.$$

Thus, we can formulate the following problem. For a given activation function Ac, find $\mathbf{a} \in A$ such that Eq. (10) and $||\mathcal{J}|| < 1$ hold. The next theorem follows directly from the Hölder inequality for $p = 1$ and $q = \infty$. It gives sufficient condition for a stable and attracting fixed point [0, 1].

Theorem 4.2 *If parameters* $\mathbf{a}$ *for activation function* $Ac(x)$ *are such that Eq.* (10) *are satisfied and*

$$\sup_{x \in \mathbb{R}}\{|Ac(x)|, Ac^2(x)\} < \sqrt{\frac{\pi e}{2}} \frac{1}{1 + \sqrt{e}} \approx 0.7801,$$

or

$$\sup_{x \in \mathbb{R}}\{|Ac(x)|\} + \sup_{x \in \mathbb{R}}\{Ac^2(x)\} < \sqrt{\frac{\pi}{2}} \approx 1.2533,$$

then the mapping $\mathbf{g}$ *has a stable and attracting fixed point [0, 1].*

5 Scaled polynomial constant unit (SPOCU) activation function

Here, we define "scaled polynomial constant unit" (SPOCU) as the novel, well-motivated activation function. We also study its properties. The SPOCU activation function is given by

$$s(x) = \alpha h\left(\frac{x}{\gamma} + \beta\right) - \alpha h(\beta) \tag{11}$$

where $\beta \in (0, 1), \alpha, \gamma > 0$ and

$$h(x) = \begin{cases} r(c), & x \geq c, \\ r(x), & x \in [0, c), \\ 0, & x < 0, \end{cases} \tag{12}$$

with $r(x) = x^3(x^5 - 2x^4 + 2)$ and $1 \leq c < \infty$. (We admit c goes to infinity with $r(c) \to \infty$.) Clearly s is continuous $s(0) = 0$ and $s'(x) = \frac{\alpha}{\gamma} h'\left(\frac{x}{\gamma} + \beta\right)$. Notice that for $c = 1$ one has $h'(1^+) = h'(1^-) = 0$ and $h'(0^+) = h'(0^-) = 0$ which implies that s' is continuous too. (This is not true for the second derivative.) For $c = 1$, the range of function s is $H_s = [s(-\beta\gamma), s((1-\beta)\gamma)] = [-\alpha r(\beta), \alpha(1 - r(\beta))], r(\beta) \in [0, 1]$.

5.1 Theoretical comparison

Let us first have a look at SNN of SPOCU. We selected $c = 1$ and $\gamma = 1$, and we computed numerically $\alpha = 2.1959, \beta = 0.6641$ from Eq. (10), whereas Jacobi matrix is

$$\mathbf{J} = \begin{bmatrix} 0.8603 & -0.0098 \\ -0.0269 & 0.1001 \end{bmatrix}$$

with $||\mathbf{J}|| < 1$. See Fig. 8 for the sigmoidal function s with this choice of parameters. Notice that for these parameters the conditions in Theorem 4.2 are not satisfied, which confirms that theorem does not present a necessary condition. Nevertheless, we have been able to find one triple of parameters which yields SNN derived from SPOCU.

Notice that SNNs cannot be derived, e.g., with ReLU, sigmoid units, tanh units and leaky ReLU. This gives an advantage of SPOCU over ReLU, but not necessarily over SELU. The same is true for another desired property. If activation function is nonlinear, then a two-layer neural network can be proved to be a universal function approximator; see [4]. Gradient-based training methods tend to be more stable if activation function has finite range, which is true only for SPOCU. Further properties that give SPOCU an advantage over ReLU and SELU are continuous differentiability, which enables gradient-based optimization methods, and the fact that it approximates identity near the origin, i.e., $s'(0) = \frac{\alpha}{\gamma} r'(\beta) = 1$. Then the neural network will learn efficiently when its weights are initialized with small random values; otherwise, a special care must be used when initializing the weight; see [23]. In SPOCU case, this is thanks to additional free parameter. Monotonicity is the only common property for all three activation function. Thus, the error surface associated with a single-layer model is guaranteed to be convex; see [26].

Table 3 summarizes the comparison. As we can see, SPOCU has six out of seven desirable properties. In contrast, ReLU and SELU share only two out of seven and three out of seven good properties, respectively.

5.2 Experimental comparison

Here, we show that SPOCU significantly outperforms other two activation functions on simple two-layer DNN model. We have used source code from iris_dnn.R, [28]. For illustration, we use a small dataset, Edgar Anderson's Iris Data (iris), well-known built-in dataset in stock R for machine learning. We have built two-layer DNN model, and subsequently, it was tested. First we transformed the data (into interval $[-\beta\gamma, (1-\beta)\gamma]$) by mapping $\gamma \frac{\mathbf{x} - \min \mathbf{x}}{\max \mathbf{x} - \min \mathbf{x}} - \beta\gamma$ with parameters given above, in order to capture polynomial (sigmoidal) influence. Then the dataset is split into two parts for training and testing. Then the training set was used to train the model. See Fig. 9 for the results about the data loss in train set and the accuracy in test compared to SELU and ReLU. For both criteria, SPOCU outperformed both SELU and ReLU.

5.3 SPOCU with $c = \infty$

Here, we illustrate SPOCU with $c = \infty$. Thus we consider activation function (11) for $c = \infty$. For such SPOCU, the range is infinite; thus, the training is generally more efficient because pattern presentations significantly affect most of the weights. The only property we lose here is monotonicity. We computed numerically $\alpha = 3.0937, \beta = 0.6653, \gamma = 4.437$ from equations (10); moreover, Jacobi matrix is

$$\begin{bmatrix} 0.8331 & -0.1169 \\ 0.0874 & 0.5334 \end{bmatrix}.$$

See Fig. 8 for the graph of function S with these parameters. See Fig. 10 for the results, the data loss in train set and the accuracy in test compared to SELU and ReLU. Here, the loss for SPOCU is uniformly better (moreover it falls much faster) with respect to losses of both SELU and ReLU. Moreover, also SPOCU accuracy is better uniformly until 950 steps; then, it may be a bit worse.

We also validated SPOCU at the MNIST database (Modified National Institute of Standards and Technology database, [12]) that is a large database of handwritten digits that is commonly used for training various image processing systems. It is also widely used for training and testing in the field of machine learning. Each image in the dataset has dimensions of 28x28 pixels and contains a centered, grayscale digit. The model will take the image as input, and it will output one of the ten possible digits (0 through 9). There are 70000 images in the data. It contains 60000 training images and 10000 testing images. We normalized inputs in order to better facilitate training of the network. We have worked within `keras` and `tensorflow` libraries (free and open-source software libraries for

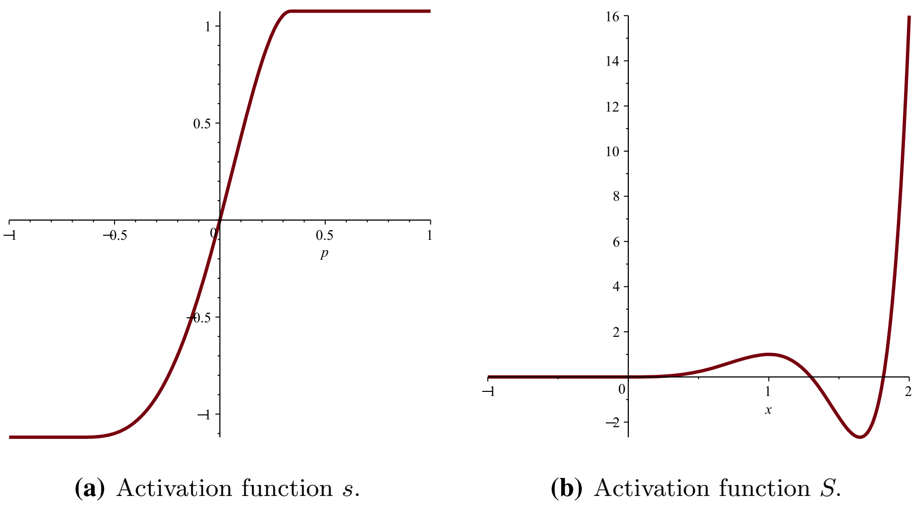


Fig. 8 SPOCU activation functions

Table 3 Comparison of the properties for three activation functions

Act. func.	Range	Smooth.	Nonlin.	Monot.	Monot. deriv.	Id. near 0	SNN
SPOCU	$[-\alpha r(\beta), \alpha(1 - r(\beta))]$	C^1	Yes	Yes	No	Yes	Yes
ReLU	$[0, \infty)$	C^0	Piecewise lin.	Yes	Yes	No	No
SELU	$(-\lambda\alpha, \infty)$	C^0	Yes	Yes	No	No	Yes

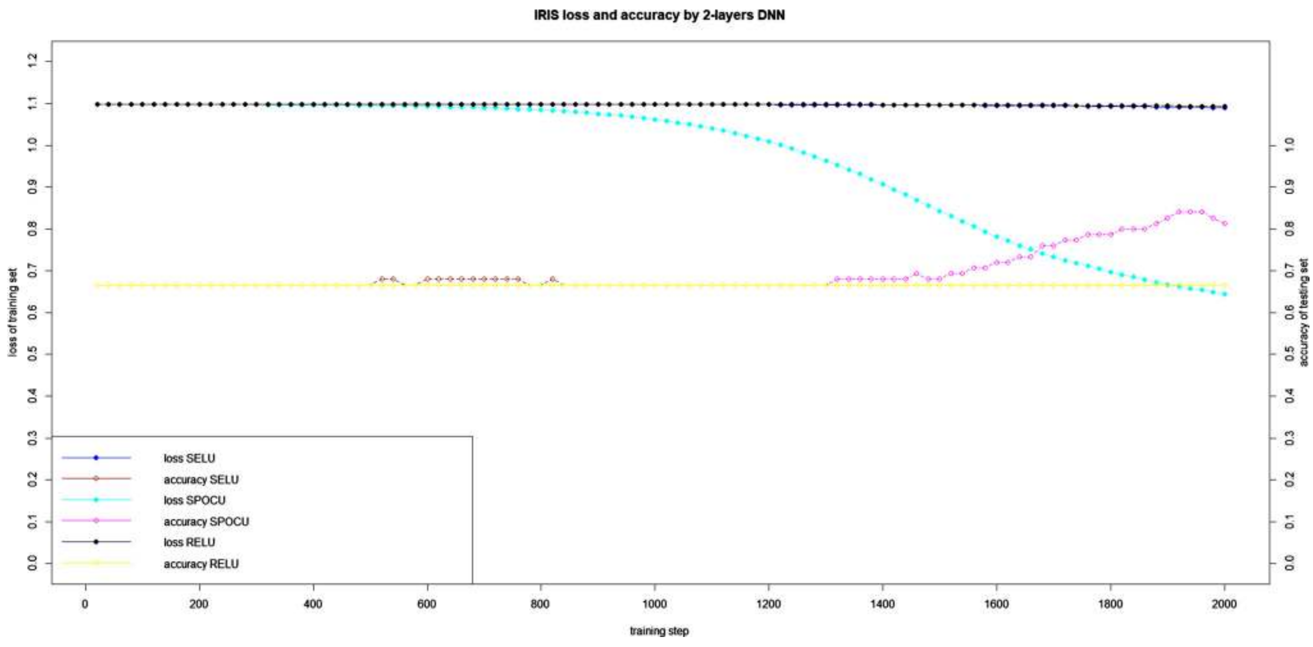


Fig. 9 Loss and accuracy for transformed dataset iris and activation function s

dataflow and differentiable programming). A sequential model is used, where each layer has exactly one input tensor and one output tensor. A 2D convolution layer (e.g., spatial convolution over images) was used. This layer creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs. A pooling layer `MaxPooling2D` followed by regularization layer `Dropout` was also used. Between the dropout and the

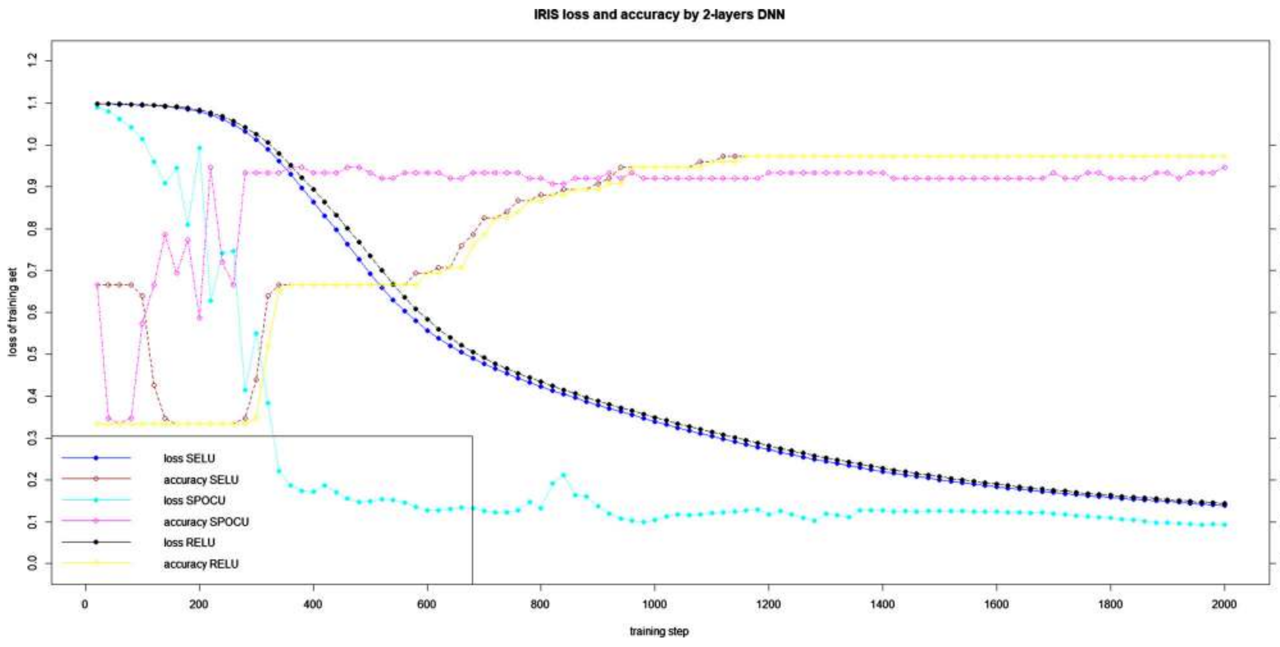


Fig. 10 Loss and accuracy for dataset iris and activation function S

dense layers, there is the `Flatten` layer, which converts the 2D matrix data to a vector. For results, see Figs. 11 and 12. Clearly SPOCU overcomes SELU in loss and ReLU in both criteria. Moreover, SPOCU and SELU reached comparable accuracy.

6 Application: cancer tissue discrimination

Developing of the algorithmic methods which can assist in cancer risk assessment is an important topic nowadays. Discrimination between mammary cancer and mastopathy tissues plays a crucial role in a clinical practice; see [9]. Noninvasive techniques may produce generally an inverse problems, e.g., estimating a Hausdorff fractal dimension from boundary of examined tissue; see [10]. Main problem here can be formulated as follows: “How can be cancer tissue discriminated from healthy tissue?”

Here, we study benign prostatic hyperplasia (BPH) and normal prostate vs. prostate cancer (PC). We consider the standard coloring of images, by hematoxylin and eosin, and we consider two magnifications, namely 100× and 200× of images; see Figs. 13. Moreover, carcinoma of the breast and mastopathy are given in Fig. 14.

Here, we have used simple estimators, i.e., reduced estimators $\mathcal{N}(k) = (k^3p)^n$, expressing retained elements in average, of $k^{3n}\tilde{p}_n$ with $k = 2$ for standard SC and $k = 3$ for modified (9 elements instead of 8) from Sect. 2.2. This implies $\hat{p} = N^{\frac{1}{n}}/3^k$, where N is the number of 1’s in measured matrix of data. Since we set the resolution of the figures to 729×729 and $729 = 3^6$, we have $n = 6$. In our case, modified SC is suitable, since otherwise data implies estimation of p over 1. We had to use only two values—binarization, i.e., each dark pixel was converted to 1, and if it is clear, it was converted to 0. If the picture is not in black and white, it was converted according to the formula $(R + G + B)/3 > 0.5$. We see the results in Table 4. One can see statistically significant difference between cancer and noncancer images, whether we take probability parameter p or fractal dimension $\dim_H$. Moreover, we can also see that the package `fractaldim` cannot catch these differences. This makes this result very valuable.

Notice that we cannot use directly more parameters without obtain more information than resulting matrix. This is quite interesting since it seems to happen that one parameter is not enough, i.e., dimension of the problem is more than one. We have also estimated values for $[p \cdot q]$ model based on theory of [9] (notice that 0 and 1 has opposite meaning in their work). The obtained results confirmed the same. We illustrate this for both CA and MA; we obtained estimations $\hat{p} = 0.24447$ and $\hat{q} = 0.11515$ for CA and $\hat{p} = 0.1934$ and $\hat{q} = 0.09675$ for MA. An alternative from the perspective of inter-patient variability can be a multifractality (see [15]). Development of such kind of techniques for analysis of several slices from 3D tissue body will be of interest for complicated cases of the National Institutes of Health (NIH) databases, like [18]. This will be a valuable future research direction.

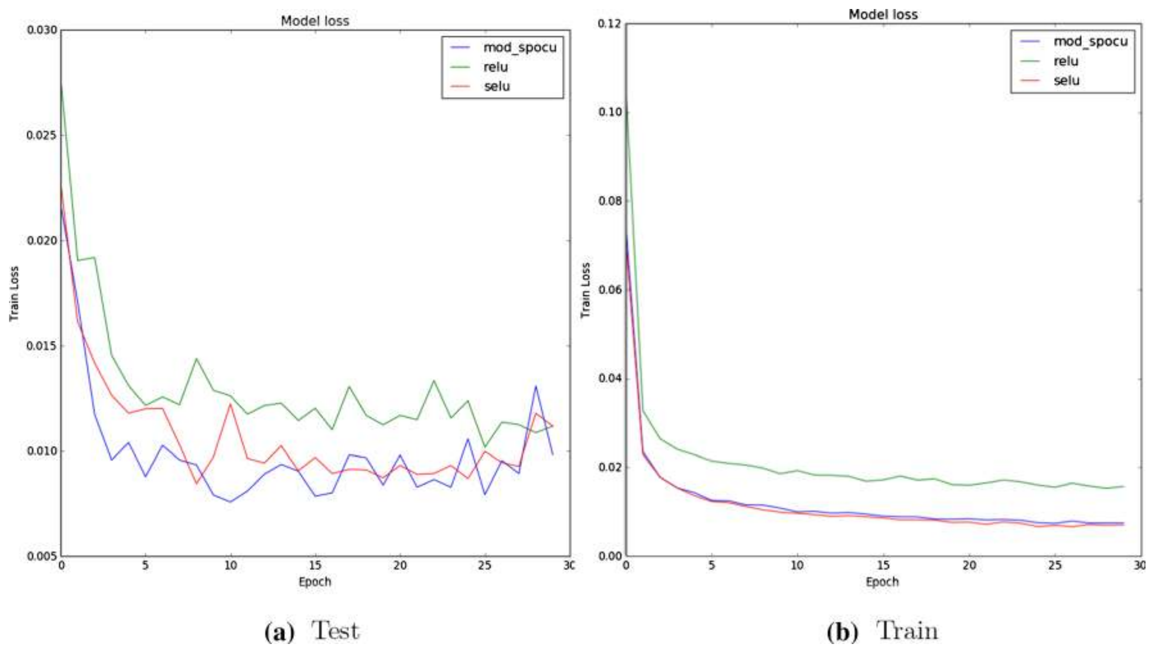


Fig. 11 Loss

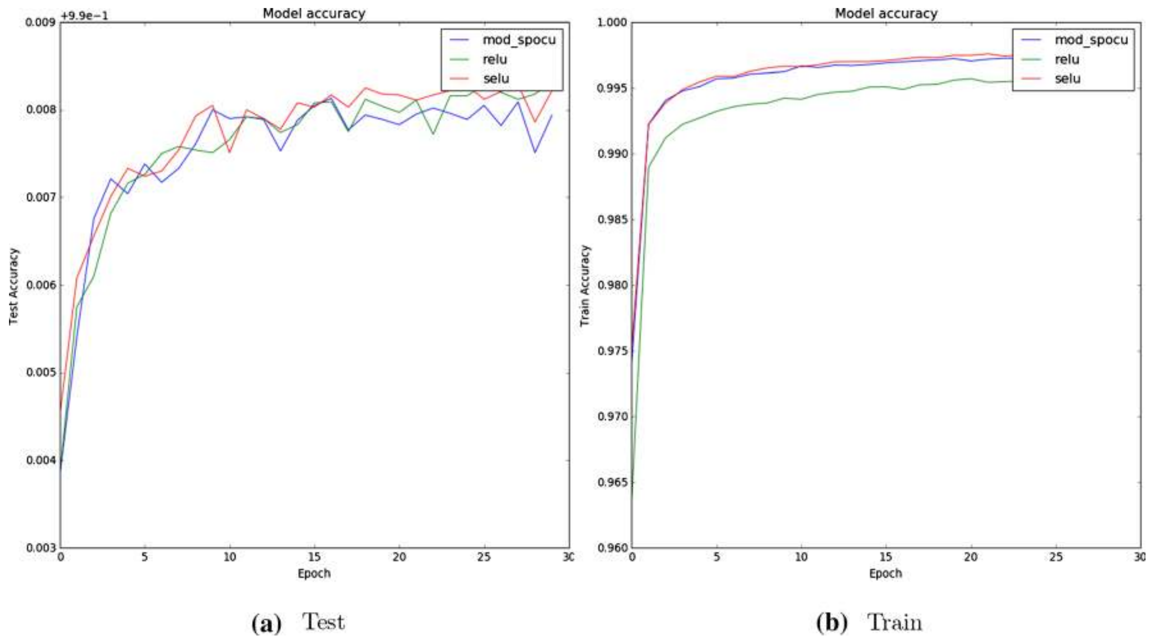


Fig. 12 Accuracy

6.1 The diagnosis of breast tissues (M = malignant, B = benign).

Here, we compare modified SPOCU, ReLU and SELU on the Wisconsin Diagnostic Breast Cancer (WDBC) dataset [25]. We measured their classification test accuracy and loss. The data, obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg, contain measurements on cells in suspicious lumps in a women's breast. In total, 357 observations (62.7%) indicate the absence of cancer cells, and 212 (37.3%) show the presence of cancerous cell. The percent is unusually large; it does not represents a typical medical analysis distribution. Typically, we will have a considerable large number of cases that represents negative vs. a small number of cases that represents positives (malignant tumors). Data include

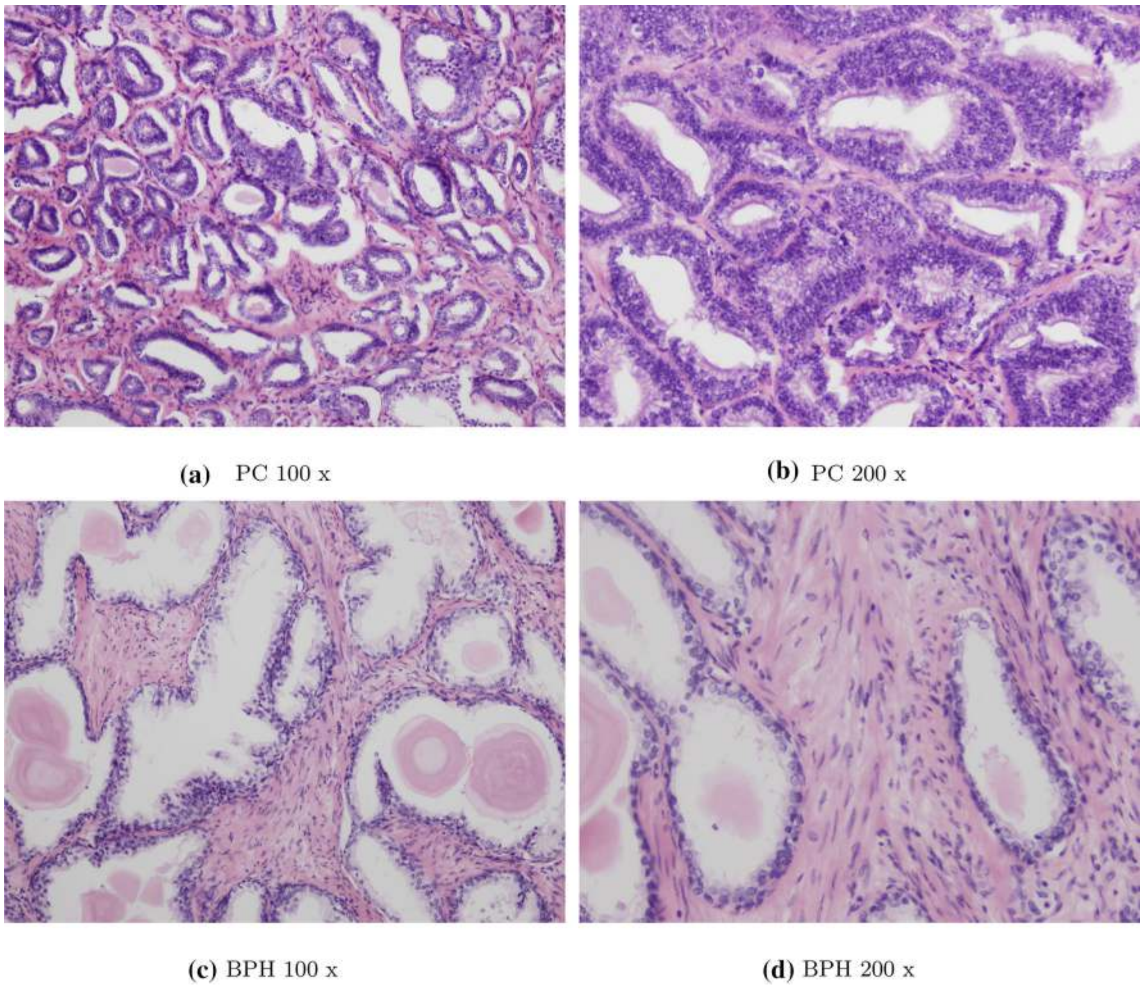


Fig. 13 PC versus BPH

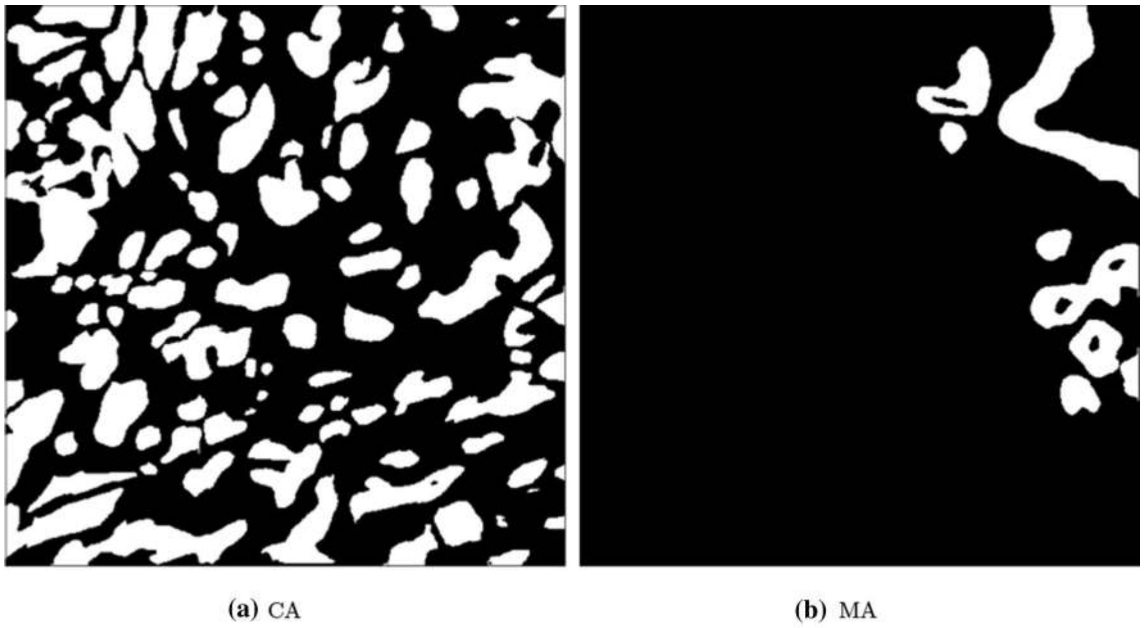


Fig. 14 CA versus MA

Table 4 Estimation of p and $\dim_H$

Estimate	PC 100	PC 200	BPH 100	BPH 200	CA	MA
$\hat{p}$	0.2702146	0.2711992	0.2319127	0.222787	0.3097428	0.3296161
$\hat{\dim}_H$	1.808917	1.812227	1.669782	1.63324	1.933188	1.989792
Fractaldim	1.57355	1.570568	1.592969	1.553224	1.495512	1.502383

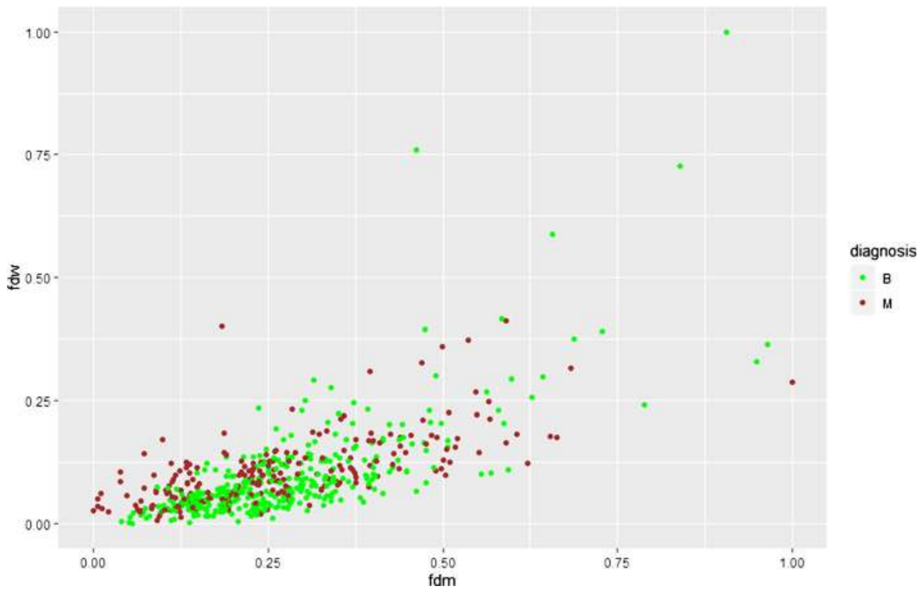


Fig. 15 fdm versus fdw for

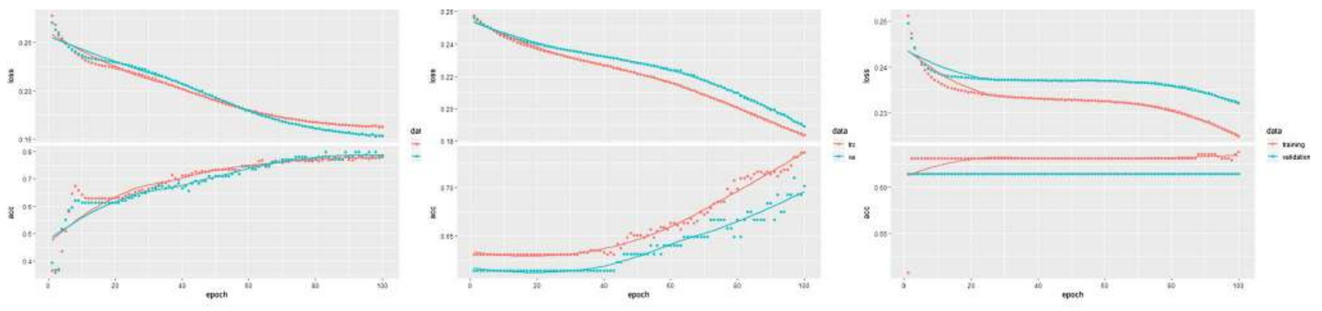
ten real-valued features computed for each cell nucleus. We deliberately focus only on variables of measured data related to fractal properties (the mean and "worst" or largest (mean of the three largest values) of fractal dimension we computed for each image—fdm and fdw), since cancer tissue discrimination is our ultimate goal. In Fig. 15, one can see relation between fdm and fdw; this confirms that the clustering is by no means unambiguous. We built DNNs with `keras` with 3 hidden layers. The number of instances is 569, and we use 80% training samples. In Fig. 16 and Table 5, we can see the results. SPOCU achieved the best results, almost 80% of absolute accuracy which means $96-99\%$ performance with respect to benchmark developed in [9].

Table 5 Comparison of three activation function at the final epoch

	Loss	Val_loss	Acc	Val_acc
SPOCU	0.1629	0.1537	0.7868	0.7807
SELU	0.1836	0.1891	0.7363	0.7018
RELU	0.2249	0.2321	0.6374	0.6140

7 Conclusion

We introduced novel percolation-based activation function SPOCU which is flexible, and in several important setups, it overcame classical SELU or ReLU approaches. We successfully validated SPOCU on both large and small



(a) Spocu **(b)** Selu **(c)** Relu

Fig. 16 Loss and accuracy

datasets, including Wisconsin Diagnostic Breast Cancer (WDBC) dataset and large dataset MNIST. We also provided careful theoretical comparisons of SPOCU to SELU or ReLU competitors.

Acknowledgements Open access funding provided by Johannes Kepler University Linz. We acknowledge LIT-2016-1-SEE-023 project "Modeling complex dependencies: how to make strategic multicriterial decisions?/mODEC" and FONDECYT Regular N1151441. Jozef Kiseľák was partially supported by the Slovak Research and Development Agency under the contract No. APVV-16-0337 and No. APVV-17-0568. We acknowledge the professional support of Editor-in-Chief Professor John MacIntyre, the unknown Associate Editor and Referees for their constructive comments.

Compliance with ethical standards

Conflict of interest On behalf of all authors, I hereby attest that there are no conflicts of interest regarding financial relationships, intellectual property or any point mentioned under the publishing ethics.

Appendix A: Auxiliaries and proofs

Remark "Appendix A.1." Naturally model (1) can be generalized also in the following sense:

$$1 \to \begin{bmatrix} \mathcal{B}_{p_{13,n}} & \mathcal{B}_{p_{23,n}} & \mathcal{B}_{p_{33,n}} \\ \mathcal{B}_{p_{12,n}} & \mathcal{B}_{p_{22,n}} & \mathcal{B}_{p_{32,n}} \\ \mathcal{B}_{p_{11,n}} & \mathcal{B}_{p_{21,n}} & \mathcal{B}_{p_{31,n}} \end{bmatrix},$$

i.e., probabilities vary not only in time n, but also in the "space" position for fixed time. Notice that $p_{9,n} = 1$ implies deterministic value 0 in the middle, which covers our model.

Example "Appendix A.2." The fractal given by $\bigotimes_{i=1}^{\infty} \mathbf{X}$ can be thus represented by the "Matrix" $\mathbf{X}$. We list several known fractals.

a) Cantor set—$\mathbf{X} = (1, 0, 1)^T$
b) Sierpiński triangle (Pascal's triangle mod 2)—$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$
c) Moiré-like pattern—$\mathbf{X} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Let $\mathbf{A}$ be an $m \times n$ matrix and $\mathbf{B}$ be an $r \times s$ matrix. The Kronecker product of them is defined as the $(m\,r) \times (n\,s)$ matrix

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{bmatrix}.$$

It is associative $(\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} = \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C})$, distributive $(\mathbf{A} + \mathbf{B}) \otimes \mathbf{C} = \mathbf{A} \otimes \mathbf{C} + \mathbf{B} \otimes \mathbf{C}$, but noncommutative.

Example "Appendix A.3." One can show that

1. $\dim_H(A_\infty) = \frac{\ln 8}{\ln 3} \approx 1.8928$, if $p_j = \mathrm{e}^{-\frac{1}{j^2}}$, which equals case $p_j = 1$,
2. $\dim_H(A_\infty) = \frac{8\ln 8 - 3}{8\ln 3} \approx 1.5515$, if $p_j = \mathrm{e}^{-\cos^4 j}$, which equals case $p_j = \mathrm{e}^{-\frac{3}{8}}$,
3. $\dim_H(A_\infty) = \frac{2\ln 8 - 1}{2\ln 3} \approx 1.4377$, if $p_j = \mathrm{e}^{-\sin^2 j}$, which equals case $p_j = \mathrm{e}^{-\frac{1}{2}}$,
4. $\dim_H(A_\infty) = \frac{\ln 8 - 1}{\ln 3} \approx 0.9826$, if $p_j = \mathrm{e}^{-\frac{j}{j+1}}$, which equals case $p_j = \mathrm{e}^{-1}$,
5. $\dim_H(A_\infty) = \frac{\ln 8 - 2}{\ln 3} \approx 0.0723$, if $p_j = \mathrm{e}^{-2}$,
6. $\dim_H(A_\infty) = 0$, if $p_j = \mathrm{e}^{-j}$.
7. $\dim_H(A_\infty) = 0$, if we have negative binomial distribution, i.e., for $j \geq 1$

$$p_j = \binom{j + r - 2}{j - 1} q^{j-1}(1 - q)^r \quad \text{for } r \in \mathbb{N}, q \in (0, 1).$$

Example "Appendix A.4."

$$p_j = \begin{cases} P, & j \text{is even.} \\ Q, & j \text{is odd,} \end{cases}$$

where $0 < P, Q < 1$. Since

$$6 \sum_{i=1}^{n} \left((2i - n - 1) \sum_{j=1}^{i} \ln p_j \right) = c_n^P \ln P + c_n^Q \ln Q,$$

where $c_n^P = c_n^Q = \frac{n}{2}(n^2 - 1)$ if n is odd and $c_n^P = \frac{n}{2}(n^2 + 2), c_n^Q = \frac{n}{2}(n^2 - 4)$ if n is even

Lemma "Appendix A.5." For $X_{p_i}^j = \mathcal{B}_{j,p_i}$, $\tilde{p}_n = \prod_{j=1}^{n} p_j$ and $n \geq 1$, we have

(I)

$$\mathbf{Y}_n = \begin{cases} 0, & \text{if in resulted deterministic matrix is value 0} \\ \mathcal{B}_{\cdot,\tilde{p}_n} & \text{otherwise} \end{cases}$$

(II) for $k \in \mathbb{N}$, the $k-$th moment is

$$\mathbb{E}\left[(\mathbf{Y})_n^k\right] = \begin{cases} 0, & \text{if in resulted deterministic matrix is value 0} \\ \tilde{p}_n & \text{otherwise} \end{cases}$$

Proof of Theorem 2.3 From (4), we have for $P(i) := \ln \tilde{p}_i < 0$ and $s_k(n) := \sum_{i=1}^n i^k$

$$\begin{aligned} \mathrm{sl}_n &= \frac{3\ln 2(n\, s_2(n) - s_1^2(n)) + n\sum_{i=1}^n i\, P(i) - s_1(n)\sum_{i=1}^n P(i)}{\ln 3(n\, s_2(n) - s_1^2(n))} \\ &= \frac{\ln 8}{\ln 3} + \frac{n\sum_{i=1}^n P(i)(i - \frac{(n+1)}{2})}{\frac{n^2(n-1)(n+1)}{12}\ln 3}, \end{aligned}$$

which yields the result. □

Proof of Thoerem 2.4 Case when at least $p_j = 0$ is trivial. Otherwise, it is sufficient to show that there exist $n \in \mathbb{N}$ such that $\sum_{i=1}^n (2i - n - 1)P(i) \le 0$. Indeed, clearly $P(i) < 0$ (for some i) and $2i - n - 1 \ge 0$ for every $i \ge \lceil \frac{n+1}{2} \rceil$. The second part follows directly from the relationship $p \to \frac{\ln(8p)}{\ln 3}$, which maps bijectively [1/8, 1] to $[0, \ln 8/\ln 3]$. □

The next lemma is a generalization of the following mixed-product property $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD})$, a proof of which is obvious.

Lemma "Appendix A.6." If $\mathbf{A}_i, \mathbf{B}_i, i = 1\ldots r$, are matrices of such size that one can form the matrix products $\mathbf{A}_i\mathbf{B}_i$, then

$$\left(\otimes_{i=1}^r \mathbf{A}_i\right)\left(\otimes_{i=1}^r \mathbf{B}_i\right) = \left(\otimes_{i=1}^r \mathbf{A}_i\mathbf{B}_i\right) \qquad (A.1)$$

holds.

Proof of Theorem 2.5 Frobenius norm of the $m \times n$ matrix can be defined in various ways:

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2} = \sqrt{\mathrm{tr}(\mathbf{A}^*\mathbf{A})} = \sqrt{\sum_{i=1}^{\min\{m,n\}} \sigma_i^2},$$

where σ_i are the singular values of $\mathbf{A}$ and where $\mathbf{A}^*$ denotes the conjugate transpose of $\mathbf{A}$. Directly from the distributivity over Kronecker product $(\mathbf{A} \otimes \mathbf{B})^* = \mathbf{A}^* \otimes \mathbf{B}^*$, we have

$$\begin{aligned} \left\|\mathbf{M}_r^{p_1,\ldots,p_r}\right\|_F &= \sqrt{\mathrm{tr}\left((\mathbf{M}_r^{p_1,\ldots,p_r})^*\mathbf{M}_r^{p_1,\ldots,p_r}\right)} \\ &= \sqrt{\mathrm{tr}\left((\bigotimes_{i=1}^r \mathbf{X}_{p_i})^* \bigotimes_{i=1}^r \mathbf{X}_{p_i}\right)} = \sqrt{\mathrm{tr}\left(\bigotimes_{i=1}^r \mathbf{X}_{p_i}^* \bigotimes_{i=1}^r \mathbf{X}_{p_i}\right)}. \end{aligned}$$

Now from the generalization of the mixed-product property (A. 1) and the fact that the trace of a Kronecker product of square matrices is given by $\mathrm{tr}(\mathbf{A} \otimes \mathbf{B}) = \mathrm{tr}\mathbf{A}\,\mathrm{tr}\mathbf{B}$, we have

$$\left\|\mathbf{M}_r^{p_1,\ldots,p_r}\right\|_F = \sqrt{\mathrm{tr}\left(\bigotimes_{i=1}^r \mathbf{X}_{p_i}^*\mathbf{X}_{p_i}\right)} = \sqrt{\prod_{i=1}^r \mathrm{tr}\left(\mathbf{X}_{p_i}^*\mathbf{X}_{p_i}\right)}.$$

□

See Table 6.

Table 6 Estimation of probabilities $\mathcal{P}_{all}, \mathcal{P}_{col}$ for given $1-p$ and r with 10^5 (10^4 for $r > 5$) repetitions

$1-p\|r$	1	2	3	4	5	6	7
0.9	0.4327	0.6764	0.8129	0.8971	0.9396	0.9666	0.9840
	0.9854	0.9996	1	1	1	1	1
0.8	0.1651	0.3080	0.4217	0.5205	0.6011	0.6647	0.7300
	0.9136	0.9932	0.9994	1	1	1	1
0.7	0.0584	0.1117	0.1631	0.2113	0.2563	0.3022	0.3480
	0.7799	0.9549	0.9905	0.9973	0.9995	1	1
0.6	0.0163	0.0334	0.0508	0.0653	0.0820	0.0981	0.1160
	0.6042	0.8484	0.9404	0.9776	0.9903	0.9956	0.9984
0.5	0.0035	0.0073	0.0119	0.0156	0.0194	0.0221	0.0340
	0.4134	0.6617	0.8074	0.8910	0.9398	0.9622	0.9800
0.4	0.0007	0.0012	0.0018	0.0028	0.0033	0.0041	0.0060
	0.2620	0.4624	0.6055	0.6994	0.7784	0.8455	0.8851
0.3	0.00001	0.0001	0.0002	0.0003	0.0003	0.0006	0.0010
	0.1353	0.2560	0.3592	0.4497	0.5255	0.5995	0.6542

References

1. Achter JD, Webb CT (2006) Pair statistics clarify percolation properties of spatially explicit simulations. Theor Popul Biol, 69 (2): 155 – 164, ISSN 0040-5809. https://doi.org/10.1016/j.tpb.2005.07.003. URL http://www.sciencedirect.com/science/article/pii/S0040580905000997
2. Bucolo M, Buscarino A, Corradino C, Fortuna L, Frasca M (2019) Turing patterns in the simplest mcnn. Nonlinear Theory Appl IEICE 10(4):390–398. https://doi.org/10.1587/nolta.10.390
3. Chayes JT, Chayes L, Durrett R (1988) Connectivity properties of mandelbrot's percolation process. Probab Theory Related Fields., pp 307–324. https://doi.org/10.1007/BF00319291 ISSN 1432-2064
4. Cybenko G (1989) Approximation by superpositions of a sigmoidal function. Math Control Signals Syst 2(4):303–314. https://doi.org/10.1007/BF02551274 ISSN 0932-4194; 1435-568X/e
5. Dekking FM, Meester RWJ (1990) On the structure of mandelbrot's percolation process and other random cantor sets. J Stat Phys 58(5):1109–1126. https://doi.org/10.1007/BF01026566 ISSN 1572-9613
6. Falconer K (2013) Fractal geometry: mathematical foundations and applications. Wiley. ISBN 9781118762868. URL https://books.google.at/books?id=XJN7AgAAQBAJ
7. Ghazal GA, Neudecker H (2000) On second-order and fourth-order moments of jointly distributed random matrices: a survey. Linear Algebra Appl, 321 (1): 61 – 93. Eighth special issue on linear algebra and statistics. ISSN 0024-3795. https://doi.org/10.1016/S0024-3795(00)00181-6. URL http://www.sciencedirect.com/science/article/pii/S0024379500001816
8. Goras L, Chua LO (1995) Turing patterns in CNNS. II. Equations and behaviors. IEEE Trans Circuits Syst I Fund Theory Appl 42(10):612–626
9. Hermann P, Mrkvička T, Mattfeldt T, Minárová M, Helisová K, Nicolis O, Wartner F, Stehlík M (2015) Fractal and stochastic geometry inference for breast cancer: a case study with random fractal models and quermass-interaction process. Stat Med 34 (18): 2636–2661, ISSN 1097-0258. https://doi.org/10.1002/sim.6497. URL http://dx.doi.org/10.1002/sim.6497. sim.6497
10. Kisel'ák J, Pardasani KR, Adlakha N, Stehlík M, Agrawal M (2013) On some probabilistic aspects of diffusion models for tissue growth. Open Stat Probab J 5: 14–21. ISSN 1876-5270/e
11. Klambauer G, Unterthiner T, Mayr A, Hochreiter S (2017) Self-normalizing neural networks. CoRR. arxiv:1706.02515
12. LeCun Y, Cortes C (2010) MNIST handwritten digit database. URL http://yann.lecun.com/exdb/mnist/
13. Liu X, Zhou J, Qian H (2019) Comparison and evaluation of activation functions in term of gradient instability in deep neural networks. In: 2019 Chinese control and decision conference (CCDC), pp 3966–3971
14. Mandelbrot BB (1974) Intermittent turbulence in self-similar cascades: divergence of high moments and dimension of the carrier. J Fluid Mech 62(2):331–358. https://doi.org/10.1017/S0022112074000711
15. Nicolis O, Kisel'ák J, Porro F, Stehlík M (2017) Multi-fractal cancer risk assessment. Stoch Anal Appl 35(2):237–256
16. Pignon D, Parmiter PJM, Slack JK, Hands MA, Hall TJ, van Daalen M, Shawe-Taylor J (Feb 1996) Sigmoid neural transfer function realized by percolation. Opt Lett 21(3):222–224. 10.1364/OL.21.000222. http://ol.osa.org/abstract.cfm?URI=ol-21-3-222
17. Rahaman M, Aldalbahi A, Govindasami P, Khanam NP, Bhandari S, Feng P, Altalhi T (2017) A new insight in determining the percolation threshold of electrical conductivity for extrinsically conducting polymer composites through different sigmoidal models. Polymers, 9 (10), ISSN 2073-4360. https://doi.org/10.3390/polym9100527. URL http://www.mdpi.com/2073-4360/9/10/527
18. Roth HR, Farag A, Turkbey EB, Lu L, Liu J, Summers RM. Nih pancreas-ct dataset. https://doi.org/10.7937/K9/TCIA.2016.tNB1kqBU
19. Shallit J, Stolfi J (1989) Two methods for generating fractals. Comput Gr 13 (2): 185–191. ISSN 0097-8493. https://doi.org/10.1016/0097-8493(89)90060-5. URL http://www.sciencedirect.com/science/article/pii/0097849389900605
20. Steeb W-H (2011) The nonlinear workbook. Chaos, fractals, cellular automata, genetic algorithms, gene expression programming, support vector machine, wavelets, hidden Markov models, fuzzy logic with C++, Java and SymbolicC++ programs. 5th ed. World Scientific, Hackensack, NJ. ISBN 978-981-4335-77-5/hbk; 978-981-4335-78-2/pbk; 978-981-4335-79-9/ebook
21. Strelniker YM, Havlin S, Bunde A (2009) Fractals and Percolation. Springer, New York, pp 3847–3858. ISBN 978-0-387-30440-3. https://doi.org/10.1007/978-0-387-30440-3_227
22. Sun W, Gao B, Chi M et al (2019) Understanding memristive switching via in situ characterization and device modeling. Nat Commun 10(2):3453
23. Sussillo D, Abbott LF (2014) Random walk initialization for training very deep feedforward networks. Neural Evolutionary Computing. arXiv:1412.6558v3
24. Wang Y, Li Y, Song Y, Rong X (2020) The influence of the activation function in a convolution neural network model of facial expression recognition. Appl Sci 10 (5). URL https://www.mdpi.com/2076-3417/10/5/1897
25. Wolberg WH, Street WN, Mangasarian OL (1992) Breast cancer wisconsin (diagnostic) data set. UCI Mach Learn Repos.http://archive.ics.uci.edu/ml/
26. Wu H (2009) Global stability analysis of a general class of discontinuous neural networks with linear growth activation functions. Inf Sci 179 (19): 3432 – 3441, ISSN 0020-0255. https://doi.org/10.1016/j.ins.2009.06.006. URL http://www.sciencedirect.com/science/article/pii/S0020025509002539
27. Xue D, Zhu Y, Zhu G-X, Yan X (1996) Generalized kronecker product and fractals. https://doi.org/10.1117/12.235499
28. Zhao P (2016) R for deep learning (i). URL https://github.com/PatricZhao/ParallelR/blob/master/ParDNN/iris_dnn.R