How deep is deep enough? - Optimizing deep neural network architecture

Achim Schilling[1], Jonas Rietsch[1], Richard Gerum[2], Holger Schulze[1], Claus Metzner[2], and Patrick Krauss[1]

[1]Experimental Otolaryngology, Neuroscience Group, University Hospital Erlangen, Germany
[2]Biophysics Group, Department of Physics, Friedrich-Alexander University Erlangen-Nürnberg (FAU), Germany

November 6, 2018

Corresponding author:
Dr. Patrick Krauss
Neuroscience Group
Experimental Otolaryngology
Friedrich-Alexander University of Erlangen-Nürnberg (FAU)
Waldstrasse 1
91054 Erlangen, Germany
Phone: +49 9131 85 43853
E-Mail: patrick.krauss@uk-erlangen.de

Keywords:
Deep Learning, Unsupervised Feature Extraction, Hyperparameter Optimization, Cluster Analysis, Representation Learning

Abstract

Deep neural networks use stacked layers of feature detectors to repeatedly transform the input data, so that structurally different classes of input become well separated in the final layer. While the method has turned out extremely powerful in many applications, its success depends critically on the correct choice of hyperparameters, in particular the number of network layers. Here, we introduce a new measure, called the generalized discrimination value (GDV), which quantifies how well different object classes separate in each layer. Due to its definition, the GDV is invariant to translation and scaling of the input data, independent of the number of features, as well as independent of the number and permutation of the neurons within a layer. We compute the GDV in each layer of a Deep Belief Network that was trained unsupervised on the MNIST data set. Strikingly, we find that the GDV first improves with each successive network layer, but then gets worse again beyond layer 30, thus indicating the optimal network depth for this data classification task. Our further investigations suggest that the GDV can serve as a universal tool to determine the optimal number of layers in deep neural networks for any type of input data.

Introduction

Recently, Artificial Intelligence (AI) has moved into more and more domains such as medicine, science, finance, engineering, or even entertainment, and is about to become ubiquitous in 21st century life. Especially in the field of Deep Learning, which again is a subfield of machine learning, the pace of progress has been extraordinary by any measure. Deep Neural Networks have proven to do well in a vast and still even increasing number of different applications like e.g. image classification, speech processing, or translation [1]. In combination with reinforcement learning the networks are even capable of learning to play video games [2]. Recently, a deep neural network from Google DeepMind, AlphaGo zero, learned to play the ancient game Go, which is thought to be the most complex board game, from scratch and even outperformed the world's best human Go player [3, 4].

However, the exact function of deep neural networks and the internal representations that emerge during learning are still not really understood [5,6], and are viewed instead as a kind of *black box*. This is known as the interpretability problem [7]. In addition, AI faces a reproducibility crisis, i.e. unpublished code and sensitivity to training conditions make many claims hard to verify [8]. Finally, the 'alchemy' problem refers to the lack of empirical rigor across the field [9]. For a given task it is not clear which set of hyper-parameters (e.g. number of layers, number of neurons per layer, etc.) is optimal. Thus, choosing a set of hyperparameters still relies on trial and error without understanding why one is better than others [10].

The hallmark of unsupervised deep learning is to find useful distributed representations of the input at multiple levels of abstraction. Complex higher-level features are defined in terms of lower-level features, thereby discovering unknown structure in the input [1]. It is assumed that the input distribution is structurally related to the task of interest, for example classification [11]. The distributed representations of the input in a certain layer of the neural network can be interpreted as a transformation of the input space into an abstract feature space. The dimensionality of this feature space corresponds to the number of neurons in the considered layer. Therefore, if the input training data is transformed into a distributed representation where it naturally divides into distinct, non-overlapping multidimensional clusters, the classification task (i.e. the assignment of labels to unknown test data) becomes trivial.

However, the network's ability of finding a good representation for a given task and input data set crucially depends on the choice of hyperparameters of which the number of layers, i.e. the depth of the neural network, is undoubtedly one of the most important. While too shallow network architectures may not generalize well, too deep architectures unnecessarily increase computational costs and may even lead to a worsening of representations.

As stated above, finding a good distributed representation is equivalent with finding a transformation of the input data into a feature space, where data naturally divide into disjoint clusters. In a previous paper we introduced the concept of the *discrimination value* which quantifies both, compactness within and distinction between given clusters of data [12]. The more dense and the more disjoint the considered clusters are, the smaller the discrimination value becomes. Our concept already proved to do well when applied to multichannel neuronal recordings obtained from different neuroimaging techniques [12, 13].

Here we generalize this concept in terms of invariance with respect to scale, dimensionality, and number of different input classes and derive a *generalized discrimination value* (GDV) in order to analyze representations of the MNIST data set [14] obtained from unsupervised training of Deep Belief Networks (DBN).

Remarkably, it turns out that in DBN the GDV as a function of depth (i.e. number of layers) first improves systematically and then reaches a minimum indicating the optimal depth and hence the best representation of a given input. We argue that our GDV may serve as a universal tool for objective determination of the optimum number of layers for a given input and thereby moving AI 'alchemy' one step towards AI 'chemistry'.

Results

GDV is invariant against dimensionality

First, we demonstrate the GDV's features by using artificial data (figure 1. Two clusters generated from two distinct 2-dimensional Gaussian distributions with $\boldsymbol{\mu_1} = (0,0)^T$, $\boldsymbol{\mu_2} = (1,1)^T$ and $\boldsymbol{\Sigma_1} = \boldsymbol{\Sigma_2} = \left(\begin{smallmatrix} 0.04 & 0 \\ 0 & 0.04 \end{smallmatrix}\right)$ lead to a discrimination value of -0.72 (figure 1a). In contrast, two clusters generated from two overlapping 2-dimensional Gaussian with $\boldsymbol{\mu_3} = (0,0)^T$, $\boldsymbol{\mu_4} = (1,1)^T$ and $\boldsymbol{\Sigma_3} = \boldsymbol{\Sigma_4} = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$ distributions lead to a discrimination value of -0.14 (figure 1c). Increasing the dimensionality to 3, i.e. adding an 'empty' dimension carrying no additional information, in both cases has no effect on the GDV (figure 1b, d, e). Even if the dimensionality is successively further increased up to 1000 dimensions (i.e. adding 998 'empty' dimensions), the discrimination values for both cases remain constant (figure 1f).

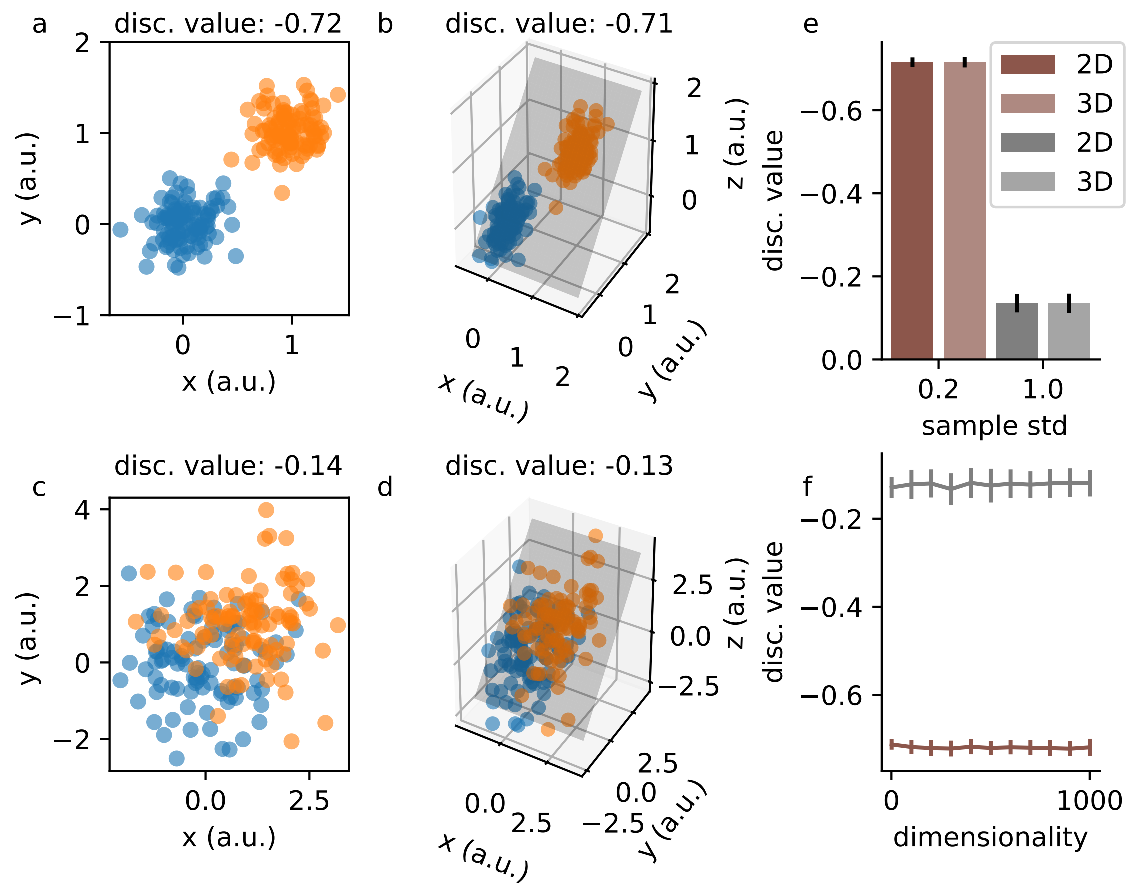

Figure 1: **Dependency of the GDV on the dimensionality.** a: two easy separable clusters drawn from two 2D Gaussian distributions ($\boldsymbol{\mu_1} = (0,0)^T$, $\boldsymbol{\mu_2} = (1,1)^T$, $\boldsymbol{\Sigma_1} = \boldsymbol{\Sigma_2} = (\begin{smallmatrix} 0.04 & 0 \\ 0 & 0.04 \end{smallmatrix})$). b: Increase of the dimensionality by copying the values of the y-coordinates also to the z-coordinate. That means each point has the location $r = (x, y, y)$. Although the dimensionality is increased the discrimination value stays constant. c, d: Analog analysis as shown in a, b for two Gaussian distributions with larger standard deviations ($\boldsymbol{\mu_3} = (0,0)^T$, $\boldsymbol{\mu_4} = (1,1)^T$, $\boldsymbol{\Sigma_3} = \boldsymbol{\Sigma_4} = (\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix})$). For both cases the GDV for 2D and 3D representation of the data is of the same value. This analysis can be done for continuously increasing dimensionality as shown in f. The plot shows that the value does not depend on the dimensionality of the data and thus the GDV cannot be artificially increased or decreased by just increasing the number of dimensions.

GDV as a function of network depth reaches a minimum

We set up two different DBN architectures, both consisting of 40 layers. The first one with a constant width of 500 neurons per layer, whereas the second one's width linearly decreases from 500 neurons in layer 1 to 110 neurons in layer 40. All together, we trained 20 different DBNs — 10 instances per architecture with different random weight initializations — on the 784-dimensional MNIST database which consists of 60,000 labeled training and 10,000 labeled test images of handwritten digits with a resolution of 28×28 pixels [14]. We emphasize that the images' labels were not used for unsupervised network training.

After learning the training images we analyzed the representations of the MNIST data that were found by the DBNs in each layer, using the test images. As the Boltzmann neurons' activations are probabilistic, each test image was propagated 1000 times through the layers to estimate the mean activation for a given input image in the considered layer. These mean activations correspond to the representation of a given input based on the learned features. The different representations of the data set in consecutive layers can be viewed as a re-coding, or a transformation, from one (simpler) feature space to another (more abstract) feature space. In this view, the number of neurons per layer correspond to the number of dimensions of that abstract feature space, and the mean activation of a certain neuron for a given input image corresponds to the image's coordinate in this dimension. Thus, each image corresponds to a certain point in the considered feature space. Analogues, each image of the non-processed, raw MNIST database can be viewed as a point in an abstract 784-dimensional input space.

Using the labels we calculated the discrimination value of the training images in each layer of all networks and, in addition, the discrimination value of the non-processed input space. Furthermore, using t-SNE to project from high-dimensional space to a 2-dimensional plane we visualized exemplary representations.

The results are summarized in figure 2 for the 10 DBNs with constant width architecture, and in figure 3 for the 10 DBNs with decreasing width architecture, respectively. The 20 individual networks' GDVs for a certain layer show only a small variance, especially in the first 25 layers (figure 2a, 3a). Since each of the 20 networks have been initialized with random weights, this demonstrates that the GDV is also invariant against permutations of the dimensions. Furthermore, we found in all networks that the GDV first decreases with increasing depth, i.e. number of layers, and then reaches a minimum at layer 30. This means that with each successive layer the unsupervised learned features, i.e. the representation of the input, becomes more and more efficient in terms of naturally dividing the data into more and more disjoint and dense clusters. Beyond layer 30 the GDV stays constant or even slightly increases again (figure 2a, 3a).

Using t-SNE we visualized the raw input data (figure 2b, 3b), the representations learned by layer 1 (figure 2c, 3c), and layer 30 (figure 2d, 3d), respectively on a 2-dimensional plane, whereby all points were colored according to the corresponding labels. Remarkably, it turns out that the points not only cluster according to the digits they represent, but also that the clusters become both more distinct and dense with increasing network depth.

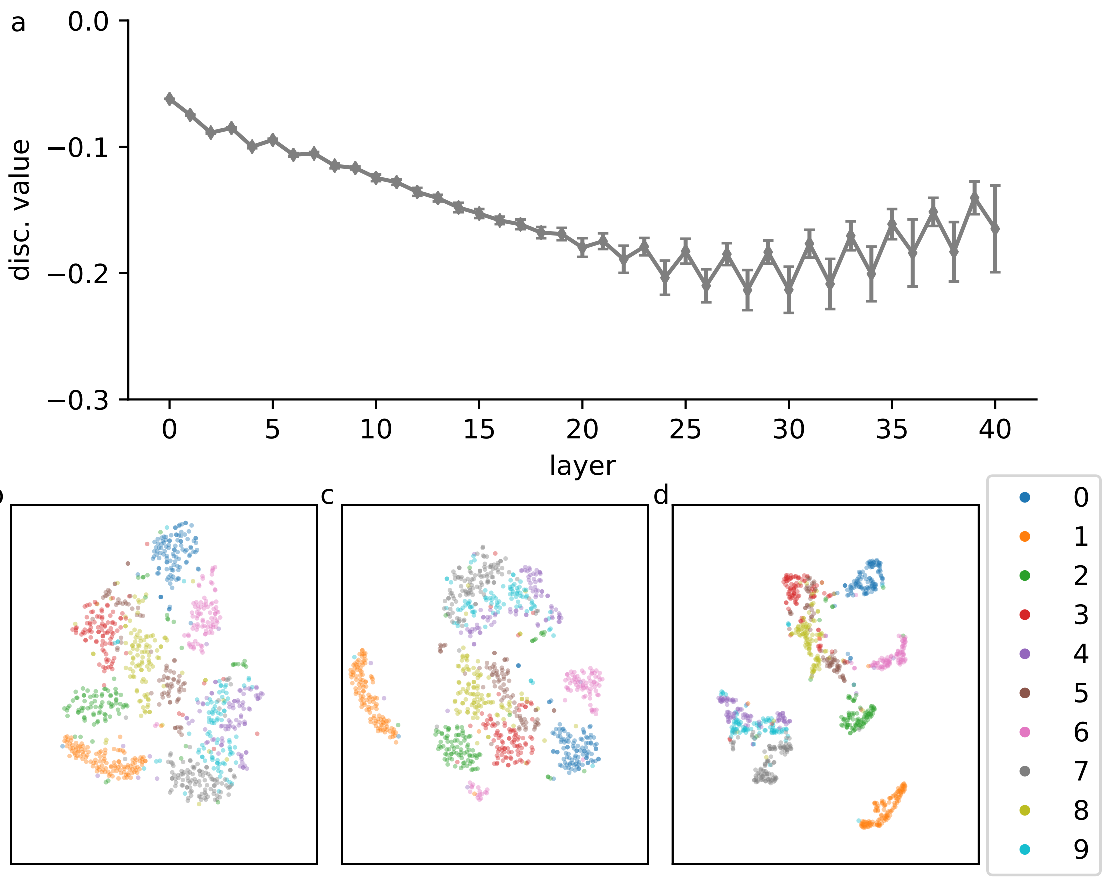

Figure 2: **GDV as a function of the number of layers with constant number of neurons per layer.** a: the separability of the data increases with successive network depth up to layer 30 where the GDV reaches a minimum. Beyond layer 30 the separability of the different clusters becomes worse again. Shown are the mean GDVs of 10 different DBNs. Error bars indicate the standard error. b-d: exemplary t-SNE projections of the MNIST data representations. b: the non-processed network input. c: representation in layer 1. d: representation in layer 30.

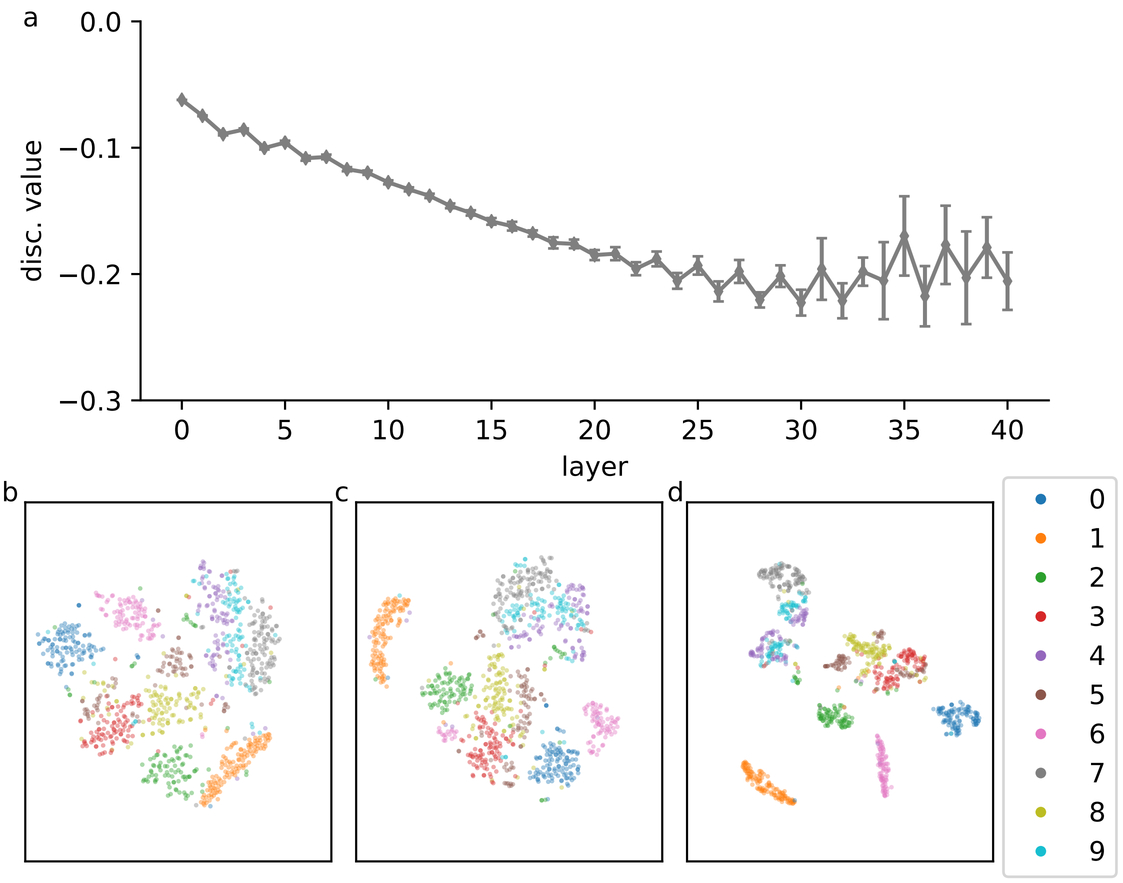

Figure 3: **GDV as a function of the number of layers with linearly decreasing number of neurons per layer.** a: as in the constant width DBNs, the separability of the data increases with successive network depth up to layer 30 where the GDV reaches a minimum. Beyond layer 30 the separability of the different clusters becomes worse again. Shown are the mean GDVs of 10 different DBNs. Error bars indicate the standard error. b-d: exemplary t-SNE projections of the MNIST data representations. b: the non-processed network input. c: representation in layer 1. d: representation in layer 30.

GDV is better in networks with decreasing width

A direct comparison of both network architectures is shown in figure 4a. From layer 1 to layer 30 the GDVs are almost identical for both architectures. Since the difference of dimensionality in both architectures increases with increasing depth (from 0 in layer 1 to 290 in layer 300) this also clearly demonstrates that our GDV is indeed invariant against dimensionality. However, beyond layer 30 the GDV of the decreasing width networks stays constant, whereas the discrimination value of the constant width networks even slightly increases, i.e. becomes worse, again.

GDV correlates with complexity of data

Using the constant width DBNs we analyzed two subsets of the MNIST data with different complexity. The less complex subset consisted of the relatively different shaped, i.e. well discriminable, digits 0, 1, and 6. The more complex subset comprised the more similar shaped, i.e. worse discriminable, digits 4, 7, and 9. As expected, the less complex data subset yielded smaller, i.e. better, GDV than the more complex subset (figure 4b).

Learned prototype digits in different layers correspond to GDV

We analyzed and visualized the representations of the dataset that evolved in different layers of a DBN with constant width. For this purpose we applied a technique derived from deep dreaming [15]. In order to visualize the input pattern that corresponds to a certain activity in a given layer, we start from the average representations of the test data samples and perform a winner-takes-all like sparsification. The 10 percent most active neurons of the considered layer are set to 1.0, whereas the remaining neurons are set to 0.0. This activity is then reversely propagated through all layers down to the input layer, where it results in an activity distribution of the 28×28 input matrix. This procedure is repeated for each test image. Subsequently we average over all such activations of the input matrix that correspond to the same label, i.e. digit. This results in a prototype input pattern for each digit represented in the considered layer. The results are summarized in figure 5. It turns out that the prototype patterns are first blurry (e.g. in layers 1 and 5) and then become increasingly clear. Remarkably, the digits appear to be

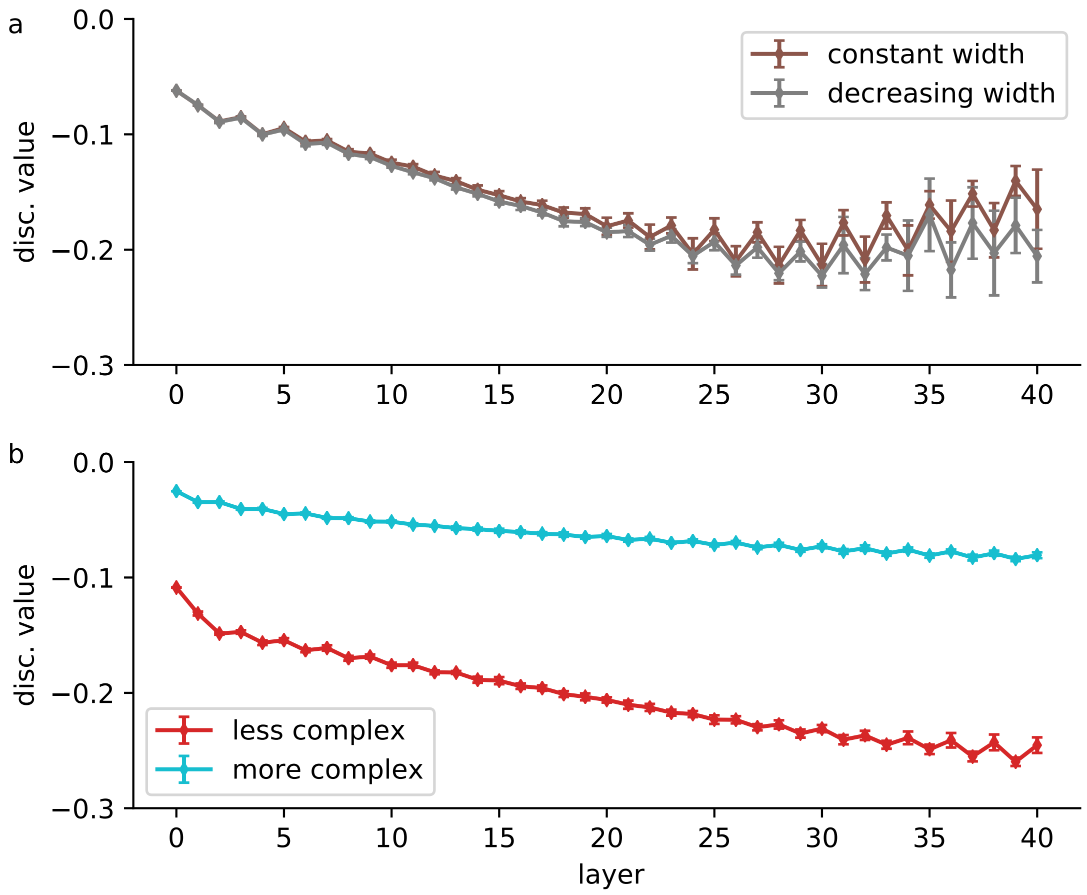

Figure 4: **Comparison of the GDV for different network architectures and different subsets of the MNIST data set.** a: comparison of the graphs shown in figures 2a and 3a. For both network architectures (constant and linearly decreasing neuron number per layer) at a network depth of 30 layers the GDV becomes minimal and thus the separability of the clusters maximal. b: analysis of two different subsets of the MNIST data. More distinct digits (0, 1, 6) yield smaller, i.e. better, GDVs than more similar digits (4, 7, 9) using the constant width DBNs.

most clear and most similar to what a human would call a prototype digit in layer 20 where the GDV has not reached its minimum. At the GDV's minimum at layer 30 the prototypes seem to be more blurry again. However, if one takes into account that the MNIST data set does not only contain clearly written digits, but also very messy written digits that should not be misclassified but instead have to be covered by the correct corresponding prototypes as well in order to further improve the GDV, this result is explainable and makes sense. Beyond layer 30 the different digit prototypes become increasingly similar and less distinct from each other which is perfectly in line with the GDV that also becomes worse again beyond layer 30.

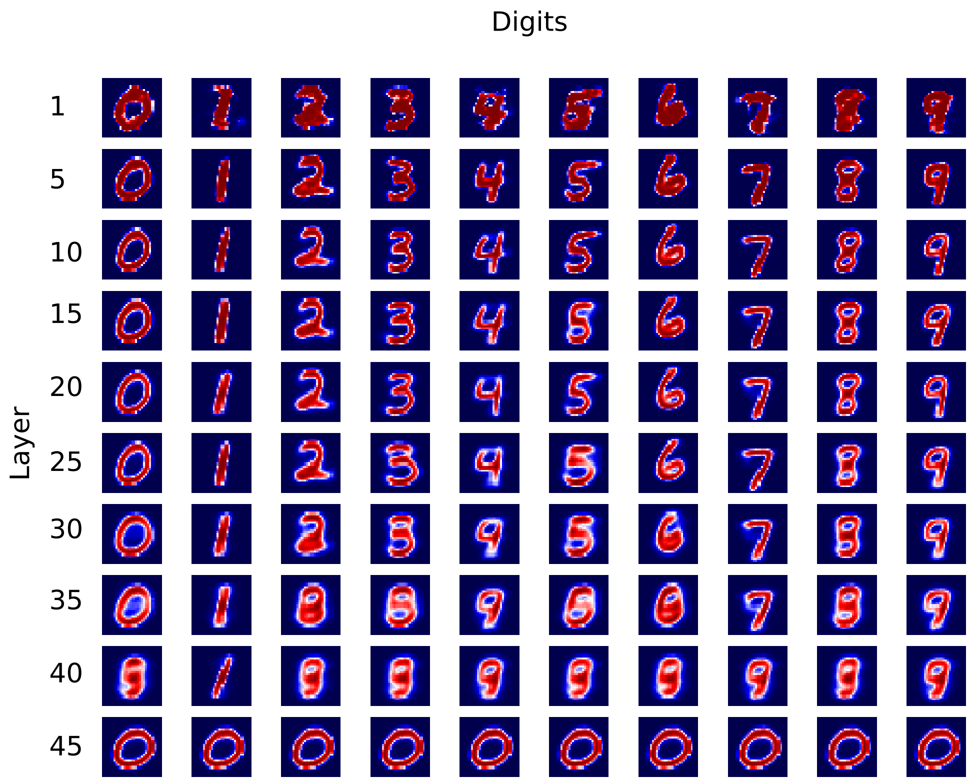

Figure 5: **Dreamed prototype templates of digit classes.** Prototype patterns are first blurry (layers 1 and 5) and then become increasingly clear (layers 10 and 20). At the GDV's minimum at layer 30 the prototypes appear to be more blurry again. Beyond layer 30 the different digit prototypes become increasingly similar and less distinct from each other.

Discussion

We generalized the previously introduced concept of the discrimination value by deriving the generalized discrimination value (GDV) to quantify both, separability and density of clusters of points in abstract feature spaces of arbitrary dimensionality. Using z-scoring in each dimension the GDV becomes invariant against translation and isotropic or even anisotropic scaling. Furthermore, the GDV is invariant against permutation of dimensions, which is trivial. Note that this permutation invariance together with scale invariance also makes the GDV invariant against mirroring, because scaling invariance also implies invariance against multiplication with a factor of -1.0.

In addition, using artificially generated data and unsupervised learned representations of the MNIST database, we demonstrated that the GDV is highly invariant against dimensionality.

As stated above, a direct comparison of both network architectures reveals that, although from layer 1 to layer 30 the discrimination values are almost identical for both architectures, beyond layer 30 the GDV of the decreasing width networks stays constant, whereas the GDV of the constant width networks becomes worse again. We speculate that compared to the small number of neurons per layer in the decreasing width architecture the relatively large number of neurons beyond layer 30 in the constant width architecture introduces some kind of noise or redundancy into the representation which in turn could explain the worse discrimination values.

However, in all analyzed networks we find that the GDV decreases with increasing depth, reaches a minimum at layer 30 and stays constant or even slightly increases again beyond layer 30. We argue that this GDV minimum indicates the optimal depth and hence the best representation of a given labeled input. However, regarding the digits' prototype templates beyond layer 30 (cf. figure 5), we hypothesize that the same input database, yet with another labeling, may lead to altered GDVs and thus different values for the GDV minimum, i.e. different optimal depths.

Furthermore, so far our proposed method of determining the optimal network depth using the GDV has been demonstrated only to do well with the MNIST database. Thus in future work the proposed method might be scrutinized using other standard databases, e.g. the CIFAR-10 dataset [16]. Nevertheless, we believe that our generalized discrimination value may serve as a universal tool to determine the optimal depth for a given input and thereby may help to push the field of deep learning towards more scientific rigor.

Methods

Generalized discrimination value (GDV)

In a previous paper we introduced the concept of the discrimination value which quantifies both, compactness within and distinction between given clusters of data [12]. The more dense and the more disjoint the considered clusters are, the smaller the discrimination value becomes. Here we generalize this concept in terms of invariance with respect to scale, dimensionality and number of different labels.

We consider a set of N points in D dimensions $\mathbf{x_{n=1..N}} = (x_{n,1}, \cdots, x_{n,D})$ and assume that all points are distributed over a finite number L of different classes $C_{l=1..L}$, i.e. to each point a label $l \in \{1, \cdots, L\}$ is assigned indicating the class the point belongs to.

First, in order to become invariant against scaling and translation, each dimension is separately z-scored (i.e. subtraction of mean and division by standard deviation) and for later convenience multiplied with $\frac{1}{2}$.

$$s_{n,d} = \frac{1}{2} \cdot \frac{x_{n,d} - \mu_d}{\sigma_d} \tag{1}$$

with $\mu_d = \frac{1}{N}\sum_{n=1}^{N} x_{n,d}$ the mean, and $\sigma_d = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(x_{n,d} - \mu_d)^2}$ the standard deviation of dimension d.

Subsequently, the rescaled points $\mathbf{s_n} = (s_{n,1}, \cdots, s_{n,D})$ are used to calculate the mean intra class distances $\bar{d}(C_l)$ and inter class distances $\bar{d}(C_l, C_m)$, respectively, which are defined as

$$\bar{d}(C_l) = \frac{2}{N_l \cdot (N_l - 1)} \sum_{i=1}^{N_l - 1} \sum_{j=i+1}^{N_l} d(\mathbf{s}_i^{(l)}, \mathbf{s}_j^{(l)}) \tag{2}$$

and

$$\bar{d}(C_l, C_m) = \frac{1}{N_l \cdot N_m} \sum_{i=1}^{N_l} \sum_{j=1}^{N_m} d(\mathbf{s}_i^{(l)}, \mathbf{s}_j^{(m)}) \tag{3}$$

with N_l, N_m the number of points in class C_l, or C_m respectively, $\mathbf{s}_i^{(l)}$ and $\mathbf{s}_j^{(m)}$ the i^{th} point of class C_l, and the j^{th} point of class C_m, respectively and

$d(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{d=1}^{D}(a_d - b_d)^2}$ the Euclidean distance between two points $\mathbf{a}$, $\mathbf{b}$ in D dimensions.

Finally, the discrimination value Δ is calculated from the mean intra and inter class distances $\bar{d}(C_l)$, and $\bar{d}(C_l, C_m)$ respectively.

$$\Delta = \frac{1}{\sqrt{D}} \cdot \frac{1}{L} \left[\sum_{l=1}^{L} \bar{d}(C_l) \; - \; \frac{2}{L-1} \sum_{l=1}^{L-1} \sum_{m=l+1}^{L} \bar{d}(C_l, C_m) \right] \tag{4}$$

where D is the number of dimensions, and L the number of different classes.

The pre-factor $\frac{2}{L-1}$ of the mean inter class distances $\bar{d}(C_l, C_m)$ serves to keep the weight of the mean intra and inter class distances on the discrimination value balanced. The factor $\frac{1}{L}$ compensates for the number of different classes and $\frac{1}{\sqrt{D}}$ keeps the discrimination value invariant against dimensionality. Together with the afore mentioned additional factor of $\frac{1}{2}$ during z-scoring the resulting discrimination value becomes -1.0 if two clusters of Gaussian distributed points are located such that the mean inter cluster distance is two times the standard deviation of the clusters.

Computational resources, software and data

All simulations were performed on a Desktop PC equipped with an i7 octa-core CPU and two Nvidia TitanX GPUs and have been implemented using Python 3.6. For efficient mathematical operations the following Python libraries were used: NumPy [17] and SciPy for mathematical operations [17], SciKitLearn [18] for the TSNE [19], Matplotlib [18] and Pylustrator [20] for the visualization of the data. Furthermore, the DBN networks were written in TensorFlow [21] including the GPU interface for efficient training of huge data sets. For the development and reliability testing of the generalized discrimination value we used the MNIST data set which is a standard data set for machine learning approaches [14, 22–24].

Boltzmann neurons

Our study is based on Boltzmann neurons [25]. The total input $z_i(t)$ of neuron i at time t is calculated as:

$$\mathrm{z_i}(t) = b_i + \sum_{j=1}^{N} w_{ij} \, y_j(t-1) \tag{5}$$

where $y_j(t-1)$ is the binary state of neuron j at time $t-1$ and w_{ij} is the corresponding weight from neuron j to neuron i and b_i is the bias of neuron i. The probability $p_i(t)$ of neuron i to be in state $y_i(t) = 1$ is given by:

$$p_i(t) = \sigma(z_i(t)), \tag{6}$$

where $\sigma(x)$ is the logistic function

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{7}$$

Deep belief networks

Deep Belief Networks (DBNs) are a class of autoencoders that have been introduced by Geoffrey Hinton [26]. They consist of a number of sub-networks, i.e. Restricted Boltzmann Machines (RBMs) [25, 27, 28], that are stacked in a way that each RBM's hidden layer serves as the visible layer for the next RBM. Thus, DBNs can be trained greedily and without supervision, one layer at a time, by applying contrastive divergence learning [29, 30] to each sub-network, starting from the lowest pair of layers (the lowest visible layer is a training set) [31]. Thereby, DBNs can learn without supervision to probabilistically reconstruct the input. The different layers then act as feature detectors.

t-distributed stochastic neighbor embedding (t-SNE)

For the purpose of visualization of the input data or the different learned representations, respectively, t-distributed stochastic neighbor embedding (t-SNE) [32] was used. It is a machine learning algorithm for visualization based on embedding high-dimensional data in low-dimensional space, e.g. two dimensions by nonlinear dimensionality reduction. Each high-dimensional object is modeled by a two-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability.

Data availability statement

All data will be made available upon request.

References

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[3] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[4] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[5] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[6] Henry W Lin, Max Tegmark, and David Rolnick. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168(6):1223–1247, 2017.

[7] Paul Voosen. The ai detectives, 2017.

[8] Matthew Hutson. Artificial intelligence faces reproducibility crisis, 2018.

[9] D Sculley, Jasper Snoek, Alex Wiltschko, and Ali Rahimi. Winner's curse? on pace, progress, and empirical rigor. 2018.

[10] Matthew Hutson. Ai researchers allege that machine learning is alchemy, 2018.

[11] Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pages 17–36, 2012.

[12] Patrick Krauss, Claus Metzner, Achim Schilling, Konstantin Tziridis, Maximilian Traxdorf, Andreas Wollbrink, Stefan Rampp, Christo Pantev, and Holger Schulze. A statistical method for analyzing and comparing spatiotemporal cortical activation patterns. *Scientific reports*, 8(1):5433, 2018.

[13] Patrick Krauss, Achim Schilling, Judith Bauer, Konstantin Tziridis, Claus Metzner, Holger Schulze, and Maximilian Traxdorf. Analysis of multichannel eeg patterns during human sleep: A novel approach. *Frontiers in human neuroscience*, 12:121, 2018.

[14] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010.

[15] Nicolas Le Roux and Yoshua Bengio. Representational power of restricted boltzmann machines and deep belief networks. *Neural computation*, 20(6):1631–1649, 2008.

[16] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[17] Stéfan Van Der Walt, S. Chris Colbert, and Gaël Varoquaux. The NumPy array: A structure for efficient numerical computation. *Computing in Science and Engineering*, 13(2):22–30, 2011.

[18] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2012.

[19] L J P Van Der Maaten and G E Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[20] R. Gerum. Pylustrator: An interactive interface to style matplotlib plots., 2018.

[21] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, Xiaoqiang Zheng, and Google Brain. TensorFlow: A System for Large-Scale Machine Learning TensorFlow: A system for large-scale machine learning. *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)*, 16:265–284, 2016.

[22] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2323, 1998.

[23] G. Hinton and R. Salakhutdinov. Reducing the Dimensionality of. *Science*, 313:504–508, 2006.

[24] Tijmen Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. *Proceedings of the 25th international conference on Machine learning - ICML '08*, pages 1064–1071, 2008.

[25] Geoffrey E Hinton and Terrence J Sejnowski. Optimal perceptual inference. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 448–453. Citeseer, 1983.

[26] Geoffrey E Hinton. Deep belief networks. *Scholarpedia*, 4(5):5947, 2009.

[27] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[28] Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted boltzmann machines. In *Proceedings of the 25th international conference on Machine learning*, pages 536–543. ACM, 2008.

[29] Miguel A Carreira-Perpinan and Geoffrey E Hinton. On contrastive divergence learning. *Artificial Intelligence and Statistics*, 10:33–40, 2005.

[30] Geoffrey E Hinton. A practical guide to training restricted boltzmann machines. In *Neural networks: Tricks of the trade*, pages 599–619. Springer, 2012.

[31] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[32] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Acknowledgments

This work was supported by the Deutsche Forschungsgemeinschaft (DFG, grant SCHU1272/12-1), and the Interdisciplinary Center for Clinical Research Erlangen (IZKF, ELAN-17-12-27-1-Schilling) at the University Hospital of the University of Erlangen-Nuremberg. The authors are grateful for the donation of two Titan Xp GPUs by the NVIDIA GPU Grant Program.

Author contributions

AS and PK designed the study. JR performed the computer simulations. AS, PK and CM developed the theoretical approach. AS and JR analyzed the neural networks. PK, AS and RG wrote the paper. HS, JR, CM and RG provided helpful discussion. All authors read and approved the final manuscript.

Additional Information

Competing interests

The authors declare no competing financial interests.