scientific reports

OPEN

Deep learning as Ricci flow

Anthony Baptista[1,2,7]✉, Alessandro Barp[1,8], Tapabrata Chakraborti[1], Chris Harbron[3], Ben D. MacArthur[1,4,5] & Christopher R. S. Banerji[1,6,7]✉

Deep neural networks (DNNs) are powerful tools for approximating the distribution of complex data. It is known that data passing through a trained DNN classifier undergoes a series of geometric and topological simplifications. While some progress has been made toward understanding these transformations in neural networks with smooth activation functions, an understanding in the more general setting of non-smooth activation functions, such as the rectified linear unit (ReLU), which tend to perform better, is required. Here we propose that the geometric transformations performed by DNNs during classification tasks have parallels to those expected under Hamilton's Ricci flow—a tool from differential geometry that evolves a manifold by smoothing its curvature, in order to identify its topology. To illustrate this idea, we present a computational framework to quantify the geometric changes that occur as data passes through successive layers of a DNN, and use this framework to motivate a notion of 'global Ricci network flow' that can be used to assess a DNN's ability to disentangle complex data geometries to solve classification problems. By training more than 1500 DNN classifiers of different widths and depths on synthetic and real-world data, we show that the strength of global Ricci network flow-like behaviour correlates with accuracy for well-trained DNNs, independently of depth, width and data set. Our findings motivate the use of tools from differential and discrete geometry to the problem of explainability in deep learning.

Keywords Deep learning, Complex network, Differential geometry, Ricci flow

Artificial neural networks are ubiquitous tools in applied science, with remarkable success in understanding the distribution of complex data, and relationships between features[1]. This power is largely due to their capacity to closely approximate unknown, non-linear, multi-dimensional functions[2]. Classical work on universal approximation demonstrated that given sufficient width, a neural network with a single hidden layer can approximate any Borel measurable function[3,4]. Subsequently, deep neural networks (DNNs) comprising series of linearly weighted connections between multiple hidden layers of non-linear activation functions[2], were demonstrated to be exponentially more efficient (in terms of the number of hidden nodes required) at approximating certain classes of function, compared to their single layer counterparts[5–7]. Informally, each layer of a DNN provides a compositional representation of the function being approximated, which the next layer can build upon in a hierarchical way[8,9]. This step-wise approach is intuitively more efficient than a single layer neural network and empirical results have accordingly demonstrated the superiority of these deeper architectures[10,11].

Despite this intuition, theoretical understanding of precisely how DNNs achieve such remarkable results lags behind their practical application[12], although this issue has been approached from numerous perspectives. For instance, theoretical results on hyperplane arrangement have provided upper bounds on the number of linear 'pieces' that a network's output function can describe, when the activation functions at hidden nodes are all piece-wise linear[5,8]. This reasoning revealed that deep networks have the capacity to readily construct more complex outputs (i.e., with more linear 'pieces') than shallow networks with the same number of nodes. These theoretical results notwithstanding, empirical studies have demonstrated that under the practical constraints of training by gradient descent, the number of linear regions constructed by deep and shallow network output functions is often comparable[13]. As another example, a method based on connections between convex geometry and optimal transport, has proven powerful in computationally simplifying generative adversarial neural networks[14]. Such approaches which combine theory with the practical constraints of DNN training are clearly needed to better understand their success.

[1]The Alan Turing Institute, The British Library, London NW1 2DB, UK. [2]School of Mathematical Sciences, Queen Mary University of London, London E1 4NS, UK. [3]Roche Pharmaceuticals, Welwyn Garden City AL7 1TW, UK. [4]School of Mathematical Sciences, University of Southampton, Southampton SO17 1BJ, UK. [5]Faculty of Medicine, University of Southampton, Southampton SO17 1BJ, UK. [6]University College London Hospitals, NHS Foundation Trust, London NW1 2BU, UK. [7]Cancer Bioinformatics, School of Cancer & Pharmaceutical Sciences, Faculty of Life Sciences and Medicine, King's College London, London, UK. [8]Faculty of Mathematical & Physical Sciences, University College London, London, UK. ✉email: anthbapt@gmail.com; cbanerji@turing.ac.uk

A complementary strand of theoretical reasoning considers how input data is sequentially deformed as it passes through successive layers of a DNN[15–19]. Employing techniques for studying the discrete geometry and topology of data clouds[16,20] it has been proposed that data drawn from complex, non-linearly separable classes, are sequentially deformed by trained DNN classifiers into linearly separable structures at the output layer[17]. In support of this hypothesis, studies using tools from computational topology have shown that data sampled from a pair of entangled manifolds undergoes topological and geometric simplifications as it passes through the layers of an optimised feed-forward DNN binary classifier—ultimately mapping in the final layer to two distinct hyper-spheres[16]. Subsequent studies have associated the ability of a DNN to perform such simplifications with higher classification accuracy[19,20]. Importantly, it was observed that these topological simplifications do not always occur monotonically as data passes through successive layers of a DNN[19], suggesting that this disentanglement is not a straightforward gradient process.

Investigations of intrinsic dimension (the minimum number of dimensions required to describe a data set), by mean field theory[17] and nearest neighbour approaches[18], have similarly found that data is geometrically 'simplified' (i.e., its intrinsic dimension is reduced) as it passes through the layers of a DNN. While data representations at successive DNN layers will typically retain a non-linear character[18], a popular heuristic is that the total curvature of the data representation reduces as data passes through successive layers, and though some empirical evidence supports such data 'flattening'[15], it is not ubiquitous[18].

The notion that DNNs orchestrate topological and geometric changes in data structures has motivated some to consider differential geometry approaches to the analysis of DNN efficacy[21]. For example, a Riemannian geometry framework has been proposed in which each activation layer of the DNN is considered as a coordinate transformation of the previous layer, finally resulting in a coordinate system in which input data classes are linearly separable[21]. While elegant, this analytical formulation relies on continuity assumptions and so does not directly apply to learning with non-smooth activation functions, such as the widely used rectified linear unit (ReLU)[22], and its variations. A similar line of research interprets the transformations performed by DNNs from a dynamical systems perspective [23–25]. While this framework can be used to design and improve the stability of DNN architectures, it also makes continuity assumptions that are not satisfied by widely used models that employ non-smooth activation functions. New geometric tools, that facilitate practical assessment of the topological simplifications of deep learning classifiers are thus needed.

Here, we outline a new approach that makes use of the fact that DNNs trained for classification tasks tend to transform a data cloud until each class is linearly separable. Underpinning this observation is a 'manifold hypothesis' in three parts: (1) given a set of data points sampled from a (possibly high dimensional) continuous feature space, we expect that associations between the features will force the sampled points to lie on a (possibly disconnected) subset, or manifold, of the feature space. (2) For a separable classification problem, the challenge facing the DNN is then to disentanlge important topological properties of the manifold from its, perhaps very complex, geometry. (3) The DNN solves this problem by evolving the manifold such that its geometry is unravelled, to reveal topological properties of the data relevant to the problem at hand. Viewed from this perspective, the dynamics associated with the DNN can therefore be seen as an example of a geometric 'flow'.

A well known tool in differential geometry for understanding evolving manifolds, with a rich theoretical and empirical literature, is Ricci flow—a partial differential equation that was introduced by Hamilton to evolve a metric on a compact manifold according to its Ricci curvature[26], and was famously central to the proof of the Poincaré conjecture by Perelman[27,28]. Intuitively, Ricci flow smooths irregularities in a complex manifold, and allows us to obtain topological information from its geometry, by evolving the initial complex manifold towards its connected sum decomposition into simpler (locally homogeneous) manifolds [29]. This has appealing parallels with the geometric view of deep learning, articulated above. However, while Ricci flow is a time continuous process, the geometric view of deep learning is discrete in both time (due to the fact that learning occurs via a sequence of discrete transformations, associated with the network layers) and space (due to the fact that the training data constitutes a point sample from the underlying manifold, not the manifold per se).

Importantly, discrete analogues of Ricci curvature have been developed from consideration of the structure of simplicial complexes[30] and optimal transport processes[31] which allow construction of discrete Ricci flows, with numerous applications in network theory[32–37].

In machine learning contexts, metric flows, of which Ricci flow is an example, have been used to interpret the training process of DNNs [38,39]. Discrete Ricci flow has been used for community detection in weighted networks—a problem with close parallels to classification[35]—and has proven useful for rewiring graph neural networks to avoid bottlenecks[36,37]. Crucially, these applications show that discrete Ricci flow shares with its continuous analogue the capacity to evolve the geometry of a network (here, a discrete analogue of a manifold) by stretching regions of negative curvature.

Here, we construct an empirical framework to quantify Ricci flow-like behaviour in a well-trained binary classifier DNN. By performing experiments on a range of synthetic and real data sets, using DNNs with a variety of architectures, we find that Ricci flow-like behaviour can generally be seen, so long as the data and its representations at DNN layers are observed at an appropriate scale. Moreover, we observe that stronger Ricci flow-like behaviour is positively associated with classification accuracy, independently of DNN width, depth and data set. Based on these results we propose that Ricci flow metrics can be used to optimise DNN architecture and possibly inform data point specific uncertainty estimates. Collectively, these findings motivate further study of how tools from differential geometry can be used to aid DNN design and improve our understanding of the theoretical basis of deep learning.

Methods

Binary classification with deep learning

Following the notation of[16] we consider a binary classification problem: given a set of randomly sampled labelled data points $X \times Y \subset M \times \{a, b\}$, where $M \subset \mathbb{R}^N$ is a compact manifold given by the union of submanifolds $M = M_a \cup M_b$, for each $x \in X$ we wish to develop a procedure to correctly determine whether it was drawn from M_a or M_b.

We propose to solve this problem with a feed-forward DNN containing L hidden layers. Following standard construction, at each layer $l = 1, \ldots, L$, comprising N_l nodes, we generate an output x_l via:

$$h_l = W_l x_{l-1} + b_l$$
$$x_l = \psi(h_l)$$

where $x_0 \in X$, $W_l \in \mathbb{R}^{N_{l-1} \times N_l}$ is a matrix describing weights on edges connecting nodes in layer $l - 1$ to layer l, $b_l \in \mathbb{R}^{N_l}$ is a vector of biases and $\psi : \mathbb{R}^{N_l} \to \mathbb{R}^{N_l}$ is a non-linear activation function implemented by each node. Here we choose the ReLu operator:

$$\psi(x) = \max(x, 0), \tag{1}$$

which was previously found to be the most efficient function for topological simplification of data through the layers of a DNN[16]. The activation function for the output layer is chosen as the sigmoid function.

The weights W_l and biases b_l are optimised using a standard RMSprop optimiser, with learning rate 0.001 and a binary cross-entropy loss function, using a labelled training data set $X_{\text{train}} \subset X$, employing *keras* (v. 2.9)[40].

The trained DNN can then be applied to an unseen test data set $X_{\text{test}} \subset X/X_{\text{train}}$, for bench-marking and further analysis. Here we will focus on how X_{test} is deformed by each layer using geometric tools associated with discrete Ricci flow.

Discrete Ricci flow

For a compact manifold M a Ricci flow evolves a Riemannian metric g_t (which describes infinitesimal distances) such that:

$$\frac{\partial g_t}{\partial t} = -\alpha \text{Ric}(g_t) \tag{2}$$

where $\text{Ric}(g_t)$ is the Ricci curvature of M at time $t \in (a, b)$ and $\alpha \in \mathbb{R}^+$. The choice $\alpha = 2$ is conventional but any positive real-valued scalar will guarantee existence of a solution in finite time. The negative sign ensures the equation is parabolic and thus the dynamics tend to 'average out' the curvature, just like heat flow averages out the distribution of heat.

To investigate whether Ricci flow-like behaviour is present in deep learning, we first need to adapt the above continuous formulation to a (spatially) discrete setting. To do so, we denote the outputs of each hidden layer $l = 1, \ldots, L$, of the trained DNN applied to the test data by X_{test}^l. For each layer $l = 1, \ldots, L$ of our trained DNN X_{test}^l provides a sample of how representative test points are deformed by the DNN up to and including the l-th layer. We can obtain a coarse approximation of the geometry of this set by constructing a k-nearest neighbour graph $\mathcal{G}_k(X_{\text{test}}^l) := (V, E_k^l)$, where V is a set of vertices corresponding to the test data points (and thus remains fixed for each k, l) and E_k^l is a set of edges in the k nearest neighbour graph constructed on X_{test}^l. We approximate the distance between two points $i, j \in V$ by the shortest path distance between i and j in $\mathcal{G}_k(X_{\text{test}}^l)$, which we denote $\gamma_k^l(i, j)$, and which forms the discrete analogue of Riemannian distance.

A number of approaches have been developed for calculating discrete Ricci curvature on graphs, including Ollivier–Ricci Curvature[31] and Forman–Ricci Curvature[30]. Ollivier–Ricci curvature is derived from optimal transport considerations and has a rich geometric interpretation[14]. Computation of Ollivier–Ricci curvature on a single weighted edge requires the solution of a linear program, making it a potentially computationally expensive measure to compute on large networks[31]. Forman–Ricci curvature is derived from consideration of simplicial complexes and has a combinatorial construction, which is computationally efficient[30]. Direct comparisons of these two discrete curvatures have yielded comparable results[41]. Here we will compute $O(10^{10})$ edgewise curvatures and thus consider the Forman–Ricci curvature, which has previously been employed for discrete Ricci flow[33,42] and which is efficient to compute.

For a graph $\mathcal{G} := (V, E)$, with V a set of vertices and E a set of edges and associated vertex weights $(W_i)_{i \in V}$ and edge weights $(\omega_{ij})_{(i,j) \in E}$, the Forman–Ricci curvature $R_{\mathcal{G}}(i, j)$ on edge $(i, j) \in E$ is given by:

$$R_{\mathcal{G}}(i, j) = W_i + W_j - (\omega_{ij})^{1/2} \left[W_i \sum_{k \neq j} (\omega_{ik})^{-1/2} + W_j \sum_{k \neq i} (\omega_{kj})^{-1/2} \right]. \tag{3}$$

Whenever $(i, j) \notin E$, $R_{\mathcal{G}}(i, j)$ is not defined.

For $\mathcal{G}_k(X_{\text{test}}^l)$, we set edge and vertex weights to 1 and the Forman–Ricci curvature on edge $(i, j) \in E_k^l$ simplifies to:

3

$$R_k^l(i,j) = 4 - \deg_k^l(i) - \deg_k^l(j), \tag{4}$$

where $\deg_k^l(i)$ is the degree of vertex i in $\mathcal{G}_k(X_{\text{test}}^l)$.

Defining global Ricci network flow-like behaviour in deep learning

Several studies have shown that as data passes through the layers of a DNN trained to solve a classification problem, there is a geometric simplification and smoothing, which leads to a gradual separation of data points from distinct classes[15,16,18]. While it is clear that geometric and/or topological simplifications through the layers of a DNN do not necessarily proceed in a monotone manner[15,16,18,19], this behaviour does put one in mind of a Ricci flow (Fig. 1). Intuitively, under a Ricci flow, regions of negative curvature in the manifold M expand, while regions of positive curvature contract[35]. Similarly, as data passes through a DNN we intuitively expect that, on average, data points belonging to the same class will cluster more closely together, while disparate points will pull apart. In this sense, well-trained DNNs may use geometric information to evolve the discrete network representation of X_{test}^l given by $\mathcal{G}_k(X_{\text{test}}^l)$ towards a simpler architecture that is easier to cut (i.e., consisting of two dense subnetworks, sparsely connected to each other). With this in mind, it is notable that the Forman–Ricci
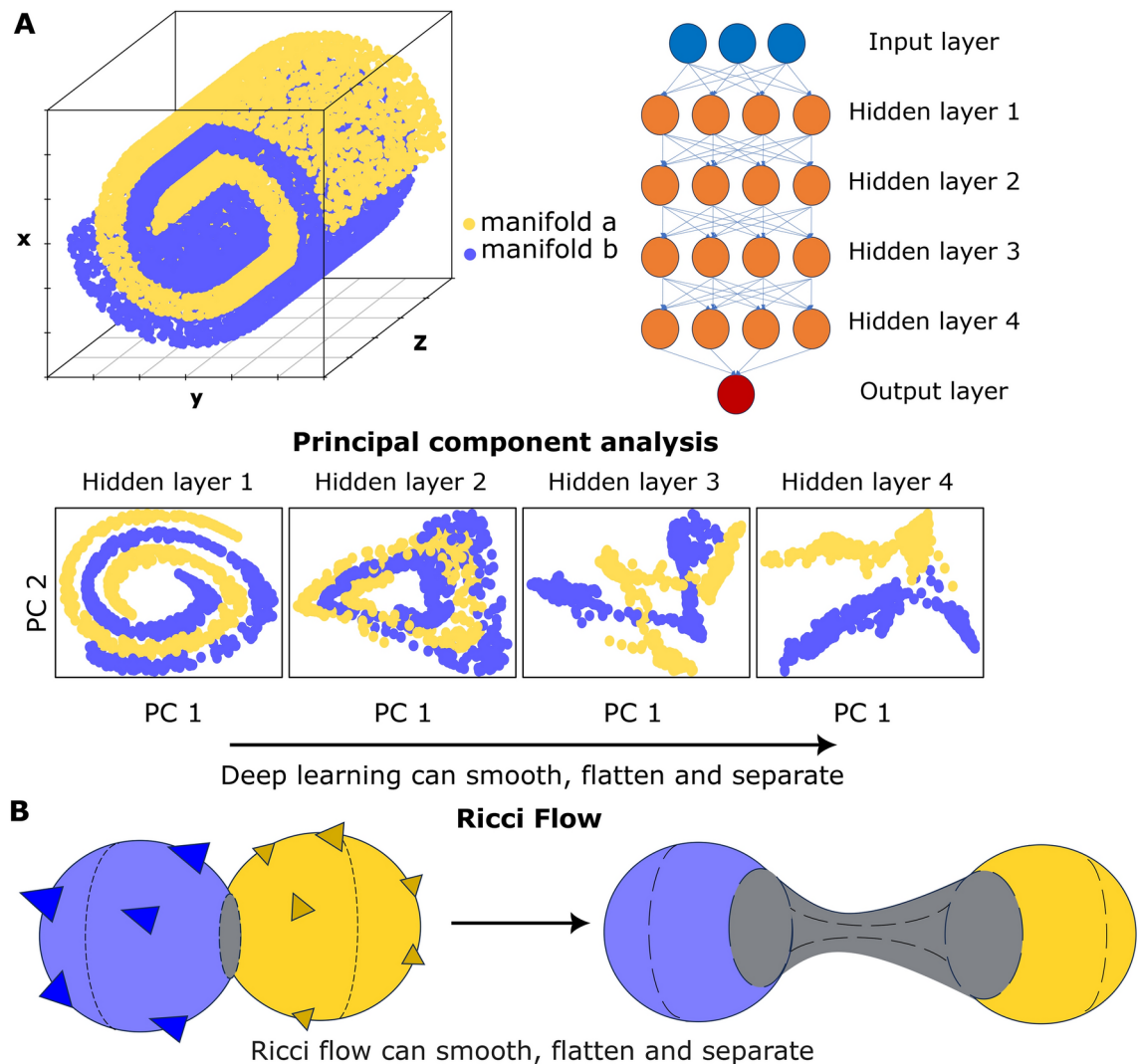

Fig. 1. Deep learning and Ricci flow. (**A**) An example of deep learning. The structure of two non-linearly separable, entwined, manifolds is learned by a deep neural network (DNN). A test set, consisting of random samples drawn from the two manifolds, is passed through the trained DNN and the output of each layer is visualised via its first two principal components. As the test set passes through the layers of the trained DNN, irregularities in the geometry of the data are smoothed, and the two manifolds are separated. (**B**) An example of Ricci flow. An irregular manifold, consisting of two generally positively curved regions joined by a region of negative curvature, evolves according to a Ricci flow. The irregularities on the positively curved regions are smoothed and the negatively curved region expands, separating them.

curvature is closely related to graph Laplacian, whose second eigenpair describes the algebraic connectivity of the network and therefore the extent to which it may be easily partitioned.

To formalise this intuition, by analogy with Eq. (2) we therefore propose that

$$g_{l+1} - g_l \approx -\alpha \mathrm{Ric}(g_l), \tag{5}$$

for each l, where

$$\mathrm{Ric}(g_l) = \mathrm{Ric}_l = \sum_{(i,j) \in E_k^l} R_k^l(i,j), \tag{6}$$

and

$$g_l = \sum_{i,j \in V} \gamma_k^l(i,j), \tag{7}$$

whenever the data manifold passing through the layers of a well trained DNN evolves analogously to a global Ricci flow. We will refer to such dynamics as *global Ricci network flow*.

The left hand side of Eq. (5) represents the total change in distance between pairs of test points at the l-th and $l+1$-th layers of the DNN (i.e., an estimate of the total expansion/contraction of the underlying manifold between layers l and $l+1$), and the right hand side is an estimate of the total curvature of the underlying manifold at layer l.

Denoting $\eta_l = g_{l+1} - g_l$ the sample Pearson correlation coefficient,

$$\rho_k = \frac{\sum_{l=1}^{L-1}(\eta_l - \bar{\eta})(\mathrm{Ric}_l - \bar{\mathrm{Ric}})}{\sqrt{\sum_{l=1}^{L-1}(\eta_l - \bar{\eta})^2}\sqrt{\sum_{l=1}^{L-1}(\mathrm{Ric}_l - \bar{\mathrm{Ric}})^2}} \tag{8}$$

is then a measure of the extent to which the DNN exhibits global Ricci network flow-like behaviour, where

$$\bar{\eta} = \frac{1}{L-1}\sum_{l=1}^{L-1}\eta_l, \tag{9}$$

and

$$\bar{\mathrm{Ric}} = \frac{1}{L-1}\sum_{l=1}^{L-1}\mathrm{Ric}_l, \tag{10}$$

represent the average expansion/contraction and curvature of the manifold as it passes through successive layers of the DNN, respectively. We will refer to ρ_k as the *Ricci coefficient* of the DNN. Negative values of the Ricci coefficient indicate the presence of global Ricci network flow-like dynamics in the DNN.

The Ricci coefficient for a DNN of depth L is computed as a Pearson correlation over $L-1$ data points. To compare DNNs of various depths we therefore employed Fisher's z transformation of Pearson's r, to calculate *adjusted Ricci coefficients*:

$$z_k = \frac{\mathrm{arctanh}(\rho_k)}{\sqrt{L-4}}. \tag{11}$$

Results
Numerical experiments
To investigate the extent to which trained DNNs exhibit Ricci flow-like dynamics, we considered 3 synthetic data sets (Fig. 2A). All describe the union of two manifolds, with varying degrees of entanglement. Data set A has been previously investigated[15] and the two manifolds, while geometrically entangled, are not topologically entangled. Data set B is similar to another previously investigated set[16] and comprises manifolds which cannot be made linearly separable without altering their topology. Data set C comprises intersecting manifolds—namely a sine wave and a horizontal plane through the origin—which are thus not easily separable either by geometric or topological transformations. We also considered the well-studied MNIST data set, containing handwritten digits $0-9$ as 28×28 pixel black and white images. As simple binary classification exemplars, we considered 2 problems: first distinguishing '1' from '7' and second distinguishing '6' from '8'. Lastly we considered the fashion MNIST (fMNIST) data set, containing pictures of different items of clothing as 28×28 pixel black and white images. As simple binary classification exemplars we again considered 2 problems: first distinguishing 'sandals' from 'ankle boots' and second distinguishing 'shirts' from 'coats'.

For each synthetic data set we trained multiple feed-forward DNNs on a training set of 1000 data points, uniformly sampled from each data set A, B and C. For the MNIST problems, we used the full training data set
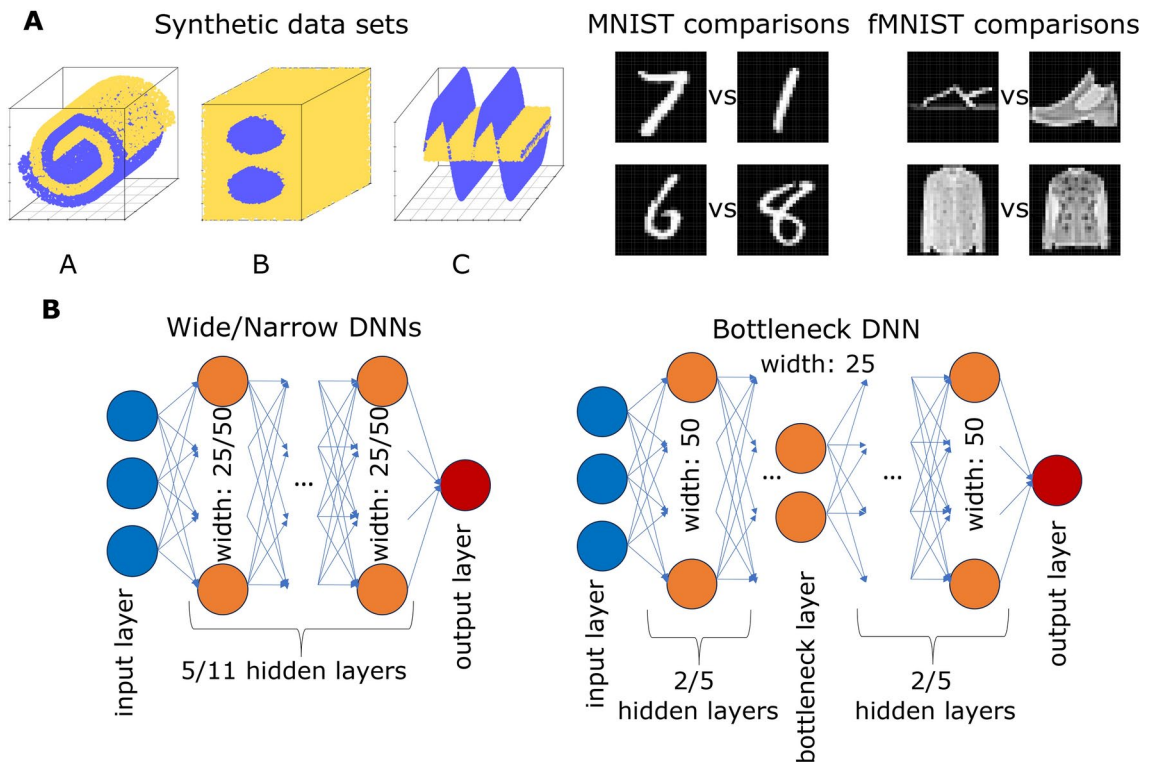
Fig. 2. Data sets for binary classification and DNN architectures trained. (**A**) Three synthetic data sets A, B and C describe binary classification problems with different degrees of geometric and topological entanglement. We also considered two binary classification problems from the MNIST data set: distinguishing similar looking numbers ('1' vs '7' and '6' vs '8'). Finally, we considered two binary classification problems from the fashion MNIST data set: distinguishing similar looking items of clothing ('sandals' vs 'ankle boots' and 'shirts' vs 'coats'). (**B**) For each problem, three different DNN widths were considered: narrow (25 nodes wide); wide (50 nodes wide); and bottleneck, as shown. For each choice of width, two depths were trained: shallow (5 hidden layers) and deep (11 hidden layers).

containing 6742 '1' digits, 6275 '7' digits, 5918 '6' digits and 5851 '8' digits. For the fMNIST problems we used the full training set containing 6000 images each of 'sandals', 'ankle boots', 'shirts' and 'coats'. For each problem we trained ensembles of DNNs with three widths (Fig. 2B): narrow (in which each layer has width 25), wide (in which each layer has width 50) and bottleneck (in which the first layer has width 50, compressing to a middle layer bottleneck of width 25 nodes and expanding back to 50 nodes on output). For each width we trained a shallow version (with 5 layers) and a deep version (with 11 layers). To account for stochastic effects, and provide a sufficient number of trained classifiers for subsequent statistical analysis, we trained 36 DNNs for each combination of width and depth.

For each synthetic data set A, B and C we also extracted a test data set X_{test} consisting of 1000 data points disjoint from the respective training sets. For the MNIST problems, the test data contains 1135 '1' digits, 1028 '7' digits, 958 '6' digits and 974 '8' digits, disjoint from the training data. For the fMNIST problems, the test data contains 1000 images of each item of clothing. Following previous studies[15,16] we investigated DNNs that are near-perfect classifiers: that is, $> 99\%$ test accuracy for data sets A and B and MNIST and $> 91\%$ test accuracy for data set C (where perfect accuracy is, by construction, impossible). For fMNIST no DNN trained reached $> 99\%$ accuracy, so near-perfect classification was considered at $> 98\%$ accuracy for the 'sandals' versus 'ankle boots' comparison and at $> 90\%$ for 'shirts' versus 'coats'. In total, 151 of 1512 DNNs trained (9.9%) failed to reach this level of test set accuracy, resulting in 1361 near perfect classifiers (≥ 25 of each DNN architecture, for each data set).

Discretising the manifold

The Ricci coefficient defined above depends upon the parameter *k*, the number of nearest neighbours in the graph $\mathcal{G}_k(X_{\text{test}}^l)$, which captures the appropriate scale of the manifold and its representations by the DNN. It is not *a prori* clear which choice of *k* will be optimal to observe global Ricci network flow-like behaviour for a given data set and DNN, nor whether there always exists a *k* that yields a negative Ricci coefficient.

We assume that the optimal *k*, if it exists, depends on input data, DNN width and depth. In order for the Ricci coefficient to be defined, *k* must be sufficiently large to ensure $\mathcal{G}_k(X_{\text{test}}^l)$ consists of a single connected component for all *l*, otherwise there will exist $i, j \in V$ such that $\gamma_k^l(i, j)$ is not-defined. For each DNN and data set, we therefore considered a range of values for *k* from the smallest *k* that ensures that $\mathcal{G}_k(X_{test})$ is a single connected component, up to at least $\sim 20\%$ of the test data set size. For the synthetic data sets where the test data set comprised 1000 points,

we considered $k \in \{6, 7, 9, 10, 15, 18, 20, 30, 50, 90, 100, 120, 140, 150, 160, 180, 200\}$. For MNIST and fMNIST where the test data set comprised ~ 2000 points we considered $k \in \{10, 20, 30, 50, 90, 100, 120, 150, 250, 350, 500\}$.

For each DNN of a given width and depth trained on a given data set, we calculated η_l and Ric_l and aggregated values across the ≥ 25 DNNs trained to near-perfect accuracy to compute a single aggregated Ricci coefficient for each considered value of k. This uncovered a complex relationship between the aggregated Ricci coefficient and k, which is influenced by data set and DNN width and depth (Fig. 3). For the synthetic data sets, hierarchical clustering revealed the relationship between aggregated Ricci coefficient and k to be strongly influenced by data set, while for the MNIST and fMNIST data sets, DNN depth was a more dominant factor.

A value of k admitting a negative aggregated Ricci coefficient was found for every binary comparison, however the range of k values varied significantly. For synthetic data sets the value of k yielding the most negative aggregated Ricci coefficient was typically low (data set A: $k = 3\% - 10\%$ of $|X_{\text{test}}|$, data set B: $k = 0.6\% - 1\%$ of $|X_{\text{test}}|$, data set C: $k = 0.6\% - 5\%$ of $|X_{\text{test}}|$). Conversely for the MNIST and fMNIST data sets the k yielding the most negative aggregated Ricci coefficient was much higher (MNIST '1' versus '7' $k = 16\% - 23\%$ of $|X_{\text{test}}|$, MNIST '6' versus '8' $k = 18\% - 25\%$ of $|X_{\text{test}}|$, fMNIST 'sandals' versus 'ankle boots' $k = 12.5\% - 25\%$ of $|X_{\text{test}}|$, fMNIST 'coats' versus 'shirts' $k = 25\%$ of $|X_{\text{test}}|$).

Increasing k tends to increase the average curvature of the network since a low value of k results in a sparser structure, while a higher value will increase the network density. The finding that our synthetic data sets show stronger Ricci flow-like behaviour at low values of k, suggests that progressing through the layers of a well trained DNN may favour increasing distances between points from different classes. Conversely, for the MNIST and fMNIST data sets stronger Ricci flow-like behaviour occurs at higher values of k, suggesting that a well trained DNN may instead favour decreasing distances between points of the same class.
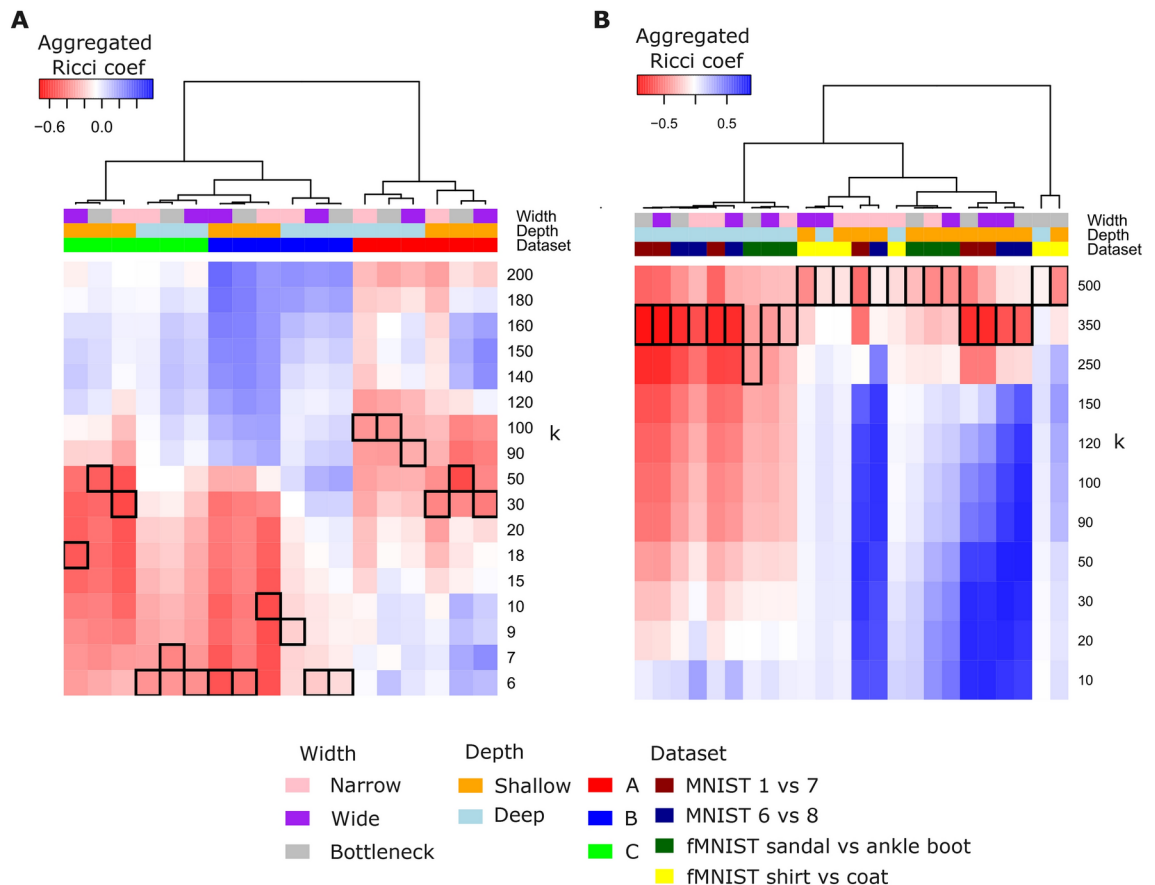

Fig. 3. Ricci flow-like behaviour and the number of nearest neighbours k. (**A**) Heatmap of aggregated Ricci coefficients, computed across ≥ 25 DNNs of a given width and depth trained on a given data set, for various values of k evaluated for synthetic test data sets A, B and C of 1000 points. (**B**) Heatmap of aggregated Ricci coefficients, computed across ≥ 25 DNNs of a given width and depth trained on a given data set, for various values of k evaluated for binary comparisons in the MNIST and fMNIST data sets with test data sets of ~ 2000 points. Black boxes outline the value of k yielding the most negative aggregated Ricci coef. For both heatmaps the dendrogram shows the results of hierarchical clustering using the aggregated Ricci coefficients as a feature. We see that higher values of k are required to observe Ricci flow-like behaviour in the MNIST data sets compared to the synthetic sets.

Ricci flow-like behaviour indicates DNNs perform dataset-specific manifold deformations

To measure global Ricci network flow-like behaviour in individual trained networks, we calculated the Ricci coefficients, given in Eq. (8), using the value of k which yielded the most negative aggregated Ricci coefficient for the associated data set, DNN width and depth, established above. $\mathcal{G}_k(X_{\text{test}}^l)$ did not consist of a single connected component, for at least one value of l, for this k in 134/1361 (9.8%) DNNs and these were excluded from further analysis. As expected by construction the majority of analysed DNNs (1144/1227, 93.2%) displayed negative Ricci coefficients. Representative plots of total curvature at layer l, against total change in distance between point pairs at layers $l + 1$ and l, for each data set and various DNN architectures are shown in (Fig. 4). Ricci coefficients for each DNN alongside test set accuracy are provided in (Table S1).

As a low optimal value of k results in low curvature representations of the data, we posited that Ricci flow across layers of a well trained DNN under such a k will prefer to separate points from distinct classes, rather than converge points from the same class. In line with this hypothesis, we found that for DNNs trained on synthetic data (where a lower k is optimal), total curvature decreases as we progress through the layers. Conversely, for
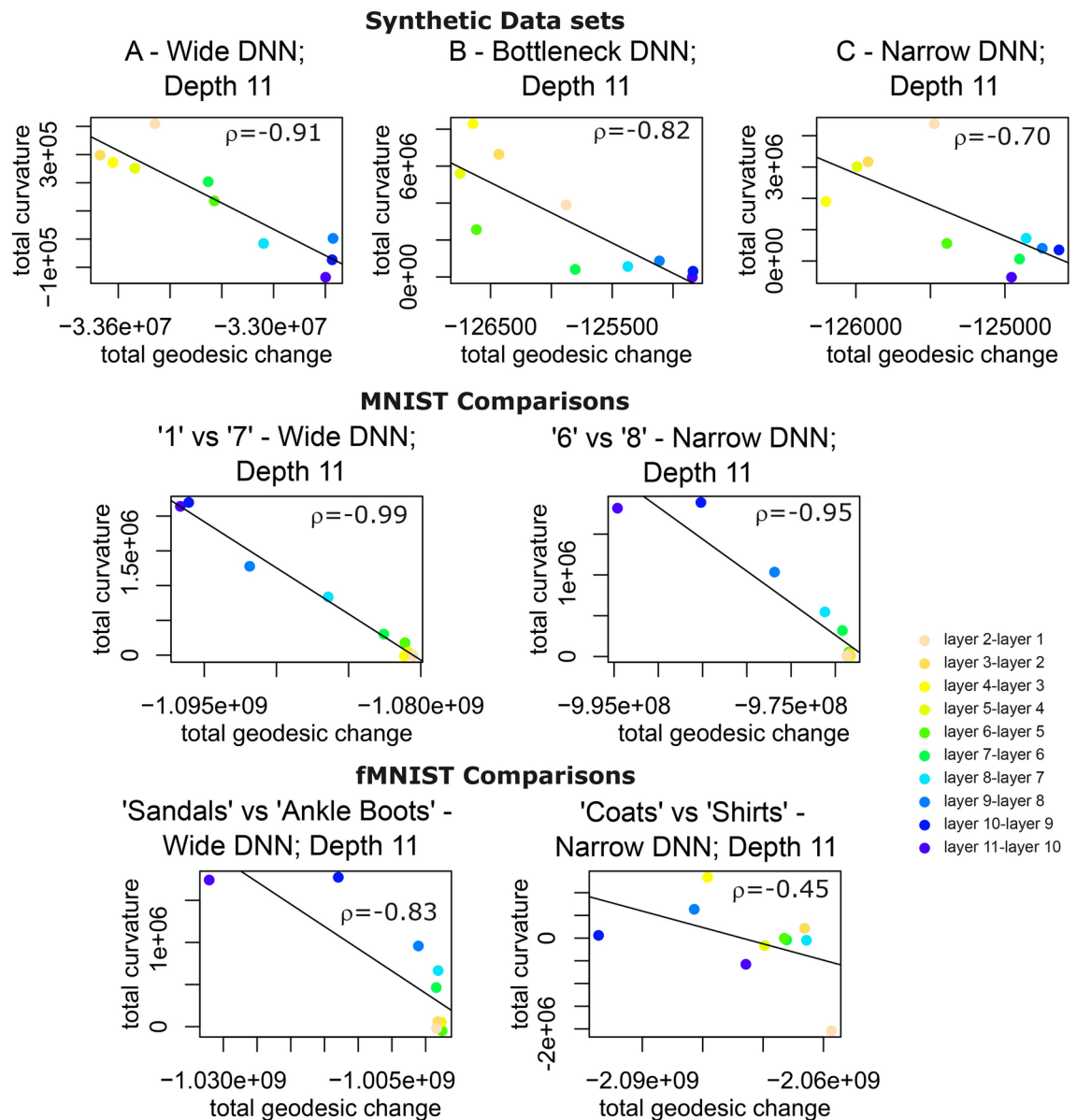

Fig. 4. Ricci flow-like behaviour has different implications for different data sets Scatter plots display total curvature at layer l against total change in distance between point pairs at layers $l + 1$ and l, for each data set and various DNN architectures. The Ricci coefficient (ρ) for each DNN is presented on each plot. For the synthetic data sets, total distance change between points increases through the layers and curvature drops, implying a separation of points from different classes. Conversely for MNIST and fMNIST total distance change decreases through the layers and curvature increases, implying an aggregation of points from the same class.

DNNs trained on MNIST and fMNIST (where a higher k is optimal), curvature increases through the layers of DNNs, suggesting that DNNs trained on these data sets prefer to push similar points together (Fig. 4).

Global Ricci network flow-like behaviour and test set accuracy

Although we selected well trained DNNs with an accuracy $> 90\%$, there was variation in the accuracy achieved above this threshold. To determine whether the Ricci coefficient associated with this variation in accuracy, we fit a multivariate linear model of test accuracy as a function of Ricci coefficient, data set, DNN width and DNN depth across the 1227 DNNs trained to sufficient accuracy, on which a Ricci coefficient was calculable. We found that the Ricci coefficient negatively associated with test accuracy independently of data set or DNN (regression coefficient t-value$= -2.27$, $p = 0.023$, Table S2), indicating that stronger global Ricci network flow-like behaviour associates with higher test accuracy.

Discussion

Central to the power of DNNs is their ability to generalise, which has been related to their capacity to identify broad geometric principles[43]. Accordingly, a number of studies have shown that consistent changes occur in the discrete geometry and topology of data clouds as they pass through the layers of a well-trained DNN classifier[15–17]. Here, we have highlighted a conceptual similarity between such behaviour and that expected by Ricci flow. By adapting recently developed tools to study discrete Ricci flow[32,35,42], we constructed a framework to assess the significance and strength of global Ricci network flow-like behaviour in data clouds passing through DNN layers. In particular, we have introduced an easily computable metric that quantifies Ricci flow-like behaviour, that we refer to as the Ricci coefficient. Using this metric, we provided a framework to identify a scale at which Ricci flow-like behaviour can be observed in a set of DNNs with a given architecture trained on a given data set. We found that this scale can provide insight into how binary classification is performed for a given data set by DNNs, in particular revealing the trade off between separation of disparate points and convergence of similar points. Our framework is computationally and conceptually simple, yet yields meaningful results relating DNN accuracy to Ricci flow-like behaviour. This motivates further investigation and deeper theoretical characterisation of how we can use Ricci flow-like behaviour to facilitate transparency in deep learning.

There is currently no consensus on how to best optimise neural network architectures. Common general approaches such as cross validation are typically computationally expensive, precluding their use in some applications[44]. There is, therefore, a need for design processes that leverage theoretical understanding of deep learning[45]. Here, we have found that the Ricci coefficient of a trained DNN computed at an appropriate scale is positively associated with classifier test accuracy, independently of DNN architecture or data set, suggesting that it may be a useful model selection tool. In particular, our numerical experiments suggest that a more negative Ricci coefficient is indicative of a DNN architecture that is appropriately matched to the problem at hand, and able to generalise well; while a less negative Ricci coefficient is indicative of an architecture that is over-fit to the problem at hand an unable to generalise well. The intuition behind why a good binary classifier DNN may act like a Ricci flow derives in part from its application to solve the Poincaré conjecture. In this setting a metric on a manifold is evolved according to the negative of its curvature, to smooth complex geometry and better unveil information about topology. Our findings suggest that a well trained binary classifier DNN may perform a similar task, smoothing out the complex and entangled input data manifold to reveal binary classes as linearly separable. While preliminary, this observation suggests that further consideration of metrics based on Ricci flow and associated geometric tools may aid in the design, development and optimisation of neural network architectures.

Normalisation of Ricci flow, by adding a term to the left hand side of equation (2), is a common approach to preserve properties of the manifold, such as volume[33]. In the discrete setting normalised Forman–Ricci flows have been applied to drive convergence of one network state to another and have proven informative in biological settings to predict realistic network rewiring trajectories[42]. Our Ricci coefficient assesses the extent to which data passing through a well trained DNN deforms according to an un-normalised Ricci flow. It is likely that adding a normaliser, which describes a network state reflecting perfect separation of ground truth classes, may better capture data deformations, and have more predictable long-time behaviour. Here we have chosen not to include such a normaliser informed by the ground truth of the test data set, as it would confound our assessment of the association between global Ricci network flow-like behaviour and test set accuracy. However, further characterisation of global Ricci network flow behaviour in DNNs may be achieved in future studies through the inclusion of an appropriate flow normaliser.

To conclude, we note that there is a distinction between global and local Ricci flow in our framework. Here, we have only considered the global geometric changes in data as it passes through a neural network, as summarised in a single constant. We find that for a suitable choice of scale, changes in total distance between data point pairs are typically negatively correlated with an assessment of the total curvature between pairs of connected data points. This finding suggests that when the total curvature of a data set representation at a given layer is negative, the total distance between data point pairs at the next layer will increase, which is the behaviour expected from a global Ricci network flow. Under a local Ricci flow, a stronger condition pertains: the geodesic distance between *any* pair of data points will be inversely correlated to the curvature along the shortest path connecting those specific points. While a local Ricci flow implies a global flow, the converse is not true. Indeed, intuitively, we do not expect that deep learning is associated with local Ricci flow for two reasons. First, this would imply that trained DNNs could be specified by a single parameter (the strength of the local flow), a highly unlikely outcome, except for perfect classifiers. Second, for data sets describing entangled classes an element of 'unraveling' will likely be required before distinct data class manifolds are separable, as has been observed in previous studies of topological simplifications in deep learning[18,20]. During this unraveling phase, strongly connected data points may move away from one another—the opposite behaviour to that expected under a

local Ricci flow. Our finding that global Ricci network flow-like behaviour associates with greater classification accuracy does, however, suggest that distances between many (but not all) data point pairs may indeed change according to a local Ricci flow, and, moreover, the proportion of data point pairs for which local Ricci flow pertains is positively associated with classifier accuracy. We hypothesise that data point pairs that exhibit strong local Ricci flow-like behaviour under the action of a DNN are precisely those points that are well characterised by the DNN, while data point pairs that do not exhibit local Ricci flow-like behaviour are those that the DNN struggles to distinguish. Further investigation of local and global Ricci network flow dynamics in DNNs could thus provide a novel approach to point specific uncertainty quantification, fairness assessment, and detection of anomalies and out of distribution data.

Data availability

The datasets used are available through the datasets class of the Python library Keras. The datasets can be loaded within a Python environment with the following functions: keras.datasets.mnist.load_data() and keras.datasets.fashion_mnist.load_data().

Code availability

The codes to run the Ricci flow-like analysis are available on GitHub at the following link: https://github.com/anthbapt/Ricci-NN. Both Python and R versions of the code are provided, along with associated data from the Fashion-MNIST dataset.

References

1. Kotsiantis, S. B. *et al.* Supervised machine learning: A review of classification techniques. *Emerg. Artif. Intell. Appl. Comput. Eng.* **160** (1), 3–24 (2007).
2. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521** (7553), 436–444 (2015).
3. Cybenko, G. Approximation by superpositions of a sigmoidal function. *Math. Control Signals Syst.* **2** (4), 303–314 (1989).
4. Hornik, K., Stinchcombe, M. & White, H. Multilayer feedforward networks are universal approximators. *Neural Netw.* **2** (5), 359–366 (1989).
5. Montufar, G. F., Pascanu, R., Cho, K. & Bengio, Y. On the number of linear regions of deep neural networks. *Adv. Neural Inf. Process. Syst.* **27** (2014).
6. Delalleau, O. & Bengio, Y. Shallow vs deep sum-product networks. In *Advances in Neural Information Processing Systems* Vol. 24 (eds Shawe-Taylor, J. et al.) (Curran Associates Inc, 2011).
7. Mhaskar, H., Liao, Q. & Poggio, T. Learning functions: when is deep better than shallow. arXiv preprint arXiv:1603.00988 (2016).
8. Pascanu, R., Montufar, G. & Bengio, Y. On the number of response regions of deep feed forward networks with piece-wise linear activations. arXiv preprint arXiv:1312.6098 (2013).
9. Mhaskar, H. N. & Poggio, T. Deep vs. shallow networks: An approximation theory perspective. *Anal. Appl.* **14** (06), 829–848 (2016).
10. Krizhevsky, A., Sutskever, I. & Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **25** (2012).
11. Goodfellow, I. J., Bulatov, Y., Ibarz, J., Arnoud, S. & Shet, V. Multi-digit number recognition from street view imagery using deep convolutional neural networks. arXiv preprint arXiv:1312.6082 (2013).
12. Serra, T., Tjandraatmadja, C. & Ramalingam, S. Bounding and counting linear regions of deep neural networks. In *International Conference on Machine Learning*, 4558–4566 (PMLR, 2018).
13. Hanin, B. & Rolnick, D. Deep relu networks have surprisingly few activation patterns. *Adv. Neural Inf. Process. Syst.* **32** (2019).
14. Lei, N., Su, K., Cui, L., Yau, S. & Gu, X. A geometric view of optimal transportation and generative model. *Comput. Aid. Geom. Des.* **68**, 1–21 (2019).
15. Brahma, P. P., Dapeng, W. & She, Y. Why deep learning works: A manifold disentanglement perspective. *IEEE Trans. Neural Netw. Learn. Syst.* **27** (10), 1997–2008 (2016).
16. Naitzat, G., Zhitnikov, A. & Lim, L.-H. Topology of deep neural networks. *J. Mach. Learn. Res.* **21** (1) (2020).
17. Cohen, U., Chung, S. Y., Lee, D. D. & Sompolinsky, H. Separability and geometry of object manifolds in deep neural networks. *Nat. Commun.* **11** (1), 746 (2020).
18. Ansuini, A., Laio, A., Macke, J. H. & Zoccolan, D. Intrinsic dimension of data representations in deep neural networks. *Adv. Neural Inf. Process. Syst.* **32** (2019).
19. Wheeler, M., Bouza, J. & Bubenik, P. Activation landscapes as a topological summary of neural network performance. In *2021 IEEE International Conference on Big Data (Big Data)*, 3865–3870 (IEEE, 2021).
20. Magai, G. & Ayzenberg, A. Topology and geometry of data manifold in deep learning. arXiv preprint arXiv:2204.08624 (2022).
21. Hauser, M. & Ray, A. Principles of Riemannian geometry in neural networks. *Adv. Neural Inf. Process. Syst.* **30** (2017).
22. Benfenati, A. & Marta, A. A singular Riemannian geometry approach to deep neural networks I. Theoretical foundations. *Neural Netw.* **158**, 331–343 (2023).
23. Haber, E. & Ruthotto, L. Stable architectures for deep neural networks. *Inverse Probl.* **34**(1), 014004 (2017).
24. Lu, Y., Zhong, A., Li, Q. & Dong, B. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *International Conference on Machine Learning*, 3276–3285 (PMLR, 2018).
25. Sander, M. E., Ablin, P., Blondel, M. & Peyré, G. Sinkformers: Transformers with doubly stochastic attention. In *International Conference on Artificial Intelligence and Statistics*, 3515–3530 (PMLR, 2022).
26. Hamilton, R. S. The Ricci flow on surfaces. *Contemp. Math.* **71**, 237–262 (1988).
27. Perelman, G. Ricci flow with surgery on three-manifolds. arXiv preprint arXiv:math/0303109 (2003).
28. Perelman, G. The entropy formula for the Ricci flow and its geometric applications. arXiv preprint arXiv:math/0211159 (2002).
29. Anderson, M. T. Geometrization of 3-manifolds via the Ricci flow. *Not. AMS* **51**, 184–193 (2004).
30. Robin Forman, R. Bochner's method for cell complexes and combinatorial Ricci curvature. *Discrete Comput. Geom.* **29** (3), 323–374 (2003).
31. Ollivier, Y. Ricci curvature of metric spaces. *Comptes Rendus Math.* **345** (11), 643–646 (2007).
32. Weber, M., Jost, J. & Saucan, E. Forman-Ricci flow for change detection in large dynamic data sets. *Axioms* **5** (4), 26 (2016).
33. Weber, M., Saucan, E. & Jost, J. Characterizing complex networks with Forman–Ricci curvature and associated geometric flows. *J. Complex Netw.* **5** (4), 527–550 (2017).

34. Cohen, H. et al. Object-based dynamics: Applying Forman–Ricci flow on a multigraph to assess the impact of an object on the network structure. *Axioms* **11** (9), 486 (2022).
35. Ni, C.-C., Lin, Y.-Y., Luo, F. & Gao, J. Community detection on networks with Ricci flow. *Sci. Rep.* **9** (1), 9984 (2019).
36. Topping, J., Di Giovanni, F., Chamberlain, B. P., Dong, X. & Bronstein, M. M. Understanding over-squashing and bottlenecks on graphs via curvature. arXiv preprint arXiv:2111.14522 (2021).
37. Bober, J., Monod, A., Saucan, E. & Webster, K. Rewiring networks for graph neural network training using discrete geometry. In *International Conference On Complex Networks And Their Applications*, 225–236 (2023).
38. Halverson, J. & Ruehle, F. Metric flows with neural networks. arXiv preprint arXiv:2310.19870 (2023).
39. Gukov, S., Halverson, J. & Ruehle, F. Rigor with machine learning from field theory to the Poincaré conjecture. *Nat. Rev. Phys.*, 1–10 (2024).
40. Chollet, F. *et al.* Keras. https://keras.io (2015).
41. Samal, A. et al. Comparative analysis of two discretizations of Ricci curvature for complex networks. *Sci. Rep.* **8**, 8650 (2018).
42. Baptista, A., MacArthur, B. D. & Banerji, C. R. S. Charting cellular differentiation trajectories with Ricci flow. *Nat. Commun.* **15** (1), 2258 (2024).
43. Stephenson, C., Padhy, S., Ganesh, A., Hui, Y. Tang. H. & Chung, S. On the geometry of generalization and memorization in deep neural networks. arXiv preprint arXiv:2105.14602 (2021).
44. Rao, R. B., Fung, G. & Rosales, R. On the dangers of cross-validation. An experimental evaluation. In *Proceedings of the 2008 SIAM International Conference on Data Mining*, 588–596 (SIAM, 2008).
45. Calvo-Pardo, H., Mancini, T. & Olmo, J. Optimal deep neural networks by maximization of the approximation power. *Comput. Oper. Res.* **156**, 106264 (2023).

Acknowledgements

Author contributions

C.R.S.B. designed research; A.Bap., A.Barp, T.C., and C.R.S.B. performed research; A.Bap., A.Barp, T.C., and C.R.S.B. analysed data; C.R.S.B. and A.Bap. created numerical code; A.Bap., A.Barp, T.C., C. H., B.D.M., and C.R.S.B. wrote the paper.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-024-74045-9.

Correspondence and requests for materials should be addressed to A.B. or C.R.S.B.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.