SIAM J. Imaging Sciences
Vol. 15, No. 3, pp. 1140–1156

Model-Centric Data Manifold: The Data Through the Eyes of the Model*

Luca Grementieri† and Rita Fioresi‡

Abstract. We show that deep ReLU neural network classifiers can see a low-dimensional Riemannian manifold structure on data. Such structure comes via the *local data matrix*, a variation of the Fisher information matrix, where the role of the model parameters is taken by the data variables. We obtain a foliation of the data domain, and we show that the dataset on which the model is trained lies on a leaf, the *data leaf*, whose dimension is bounded by the number of classification labels. We validate our results with some experiments with the MNIST dataset: paths on the data leaf connect valid images, while other leaves cover noisy images.

Key words. deep learning, information geometry, manifold learning, Fisher matrix

AMS subject classification. 68T07

DOI. 10.1137/21M1437056

1. Introduction.

In machine learning, models are categorized as discriminative models or generative models. From its inception, deep learning has focused on classification and discriminative models [22, 16, 7]. Another perspective came with the construction of generative models based on neural networks [19, 14, 33, 18]. Both kinds of models give us information about the data and the similarity between examples. In particular, generative models introduce a geometric structure on generated data. Such models transform a random low-dimensional vector to an example sampled from a probability distribution approximating the one of the training dataset. As proved by [5], generated data lie on a countable union of manifolds. This fact supports the human intuition that data have a low-dimensional manifold structure, but in generative models the dimension of such a manifold is usually a hyper-parameter fixed by the experimenter. A recent algorithm by [30] provides a way to find an approximation of the number of dimensions of the data manifold, deactivating irrelevant dimensions in a GAN (see [9] and [31] for another interesting perspective on data structure).

Similarly, here we try to understand if a discriminative model can be used to detect a manifold structure on the space containing data and to provide tools to navigate this manifold. The implicit definition of such a manifold and the possibility to trace paths between points on the manifold can open many possible applications. In particular, we could use paths to define a system of coordinates on the manifold (more specifically on a chart of the manifold). Such coordinates would immediately give us a low-dimensional parametrization of our data, allowing us to do dimensionality reduction. This question was also explored in the context of *manifold learning* (see [24, 11, 26, 1] and references therein). In particular in [34], where

*Received by the editors July 28, 2021; accepted for publication (in revised form) March 14, 2022; published electronically July 28, 2022.
https://doi.org/10.1137/21M1437056

†Zuru Tech, Modena, 41124, Italy (luca.g@zuru.tech).
‡University of Bologna, Bologna, 40126, Italy (rita.fioresi@unibo.it).

the authors take an infinitesimal approach, the approximation of the manifold of data points is obtained via its tangent space at suitable points in order to define a coordinate system. This suggests that our approach, more theoretical and global, can lead to a more intrinsic description of such manifold.

In supervised learning, a model is trained on a labeled dataset to identify the correct label on unseen data. A trained neural network classifier builds a hierarchy of representations that encodes increasingly complex features of the input data [28]. Through the representation function, a distance (e.g., Euclidean or cosine) on the representation space of a layer endows input data with a distance. The deeper the layer, the better the metric reflects the similarity of data according to the task at hand. This observation suggests that the model is implicitly organizing the data according to a suitable structure (another interesting point of view on the same matter appears also in [12]).

Unfortunately, these intermediate representations and metrics are insufficient to understand the geometric structure of data. First of all, representation functions are not invertible, so we cannot recover the original example from its intermediate representation or interpolate between data points. Moreover, the domain of representation functions is the entire data domain $\mathbf{R}^n$. This domain is mostly composed of meaningless noise, and data occupy only a thin region inside of it. So, even if representation functions provide us a distance, those metrics are incapable of distinguishing between meaningful data and noise.

We find out that a ReLU neural network implicitly identifies a low-dimensional submanifold of the data domain that contains real data. We prove that, if the activation function is piecewise linear (e.g., ReLU), the neural network decomposes the data domain $\mathbf{R}^n$ as the disjoint union of submanifolds (the leaves of a foliation, using the terminology of differential geometry). The dimension of every submanifold (every leaf of the foliation) is bounded by the number of classes of our classification model, so it is much smaller than n, the dimension of the data domain $\mathbf{R}^n$. Our main theoretical result, Theorem 3.1, stems from the study of the properties of a variant of the Fisher information matrix, the *local data matrix*. However, Theorem 3.1 cannot tell us which leaves of this foliation are meaningful or useful for practical applications: the interpretation of this geometric structure can only come from experiments. We report experiments performed on MNIST dataset. We choose to focus on MNIST because it is easily interpretable.

Our experiments suggest that all valid data points lie on only one leaf of the foliation, the *data leaf*. To observe this phenomenon we take an example from the dataset, and we try to connect it with another random example following a path along the leaf containing the starting point. If such a path exists, it means that the destination example belongs to the same leaf of the foliation; otherwise, the two data points belongs to different leaves. Figure 1 pictures a summary of our experiments.

Visualizing the intermediate points on these joining paths, we see that the low-dimensional data manifold defined by the model is not the anthropocentric data manifold composed of data meaningful for a human observer. The model-centric data manifold comprises images that do not belong to a precise class. The model needs those transition points to connect points with different labels. At the same time, it understands that such transition points represent an ambiguous digit: on such points, the model assigns a low probability to every class.

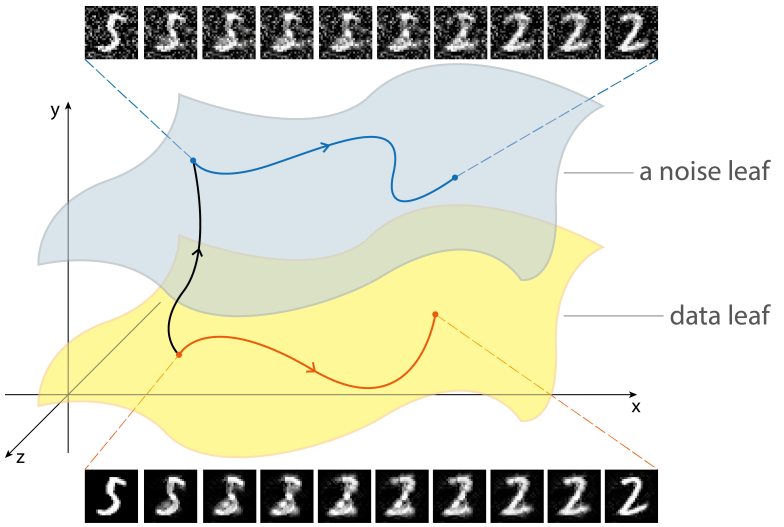

Figure 1. *We provide algorithms to move along a leaf and orthogonally to it. In our experiments, we observe that moving orthogonally to the leaves changes the amount of noise in images. On the other hand, a path on a leaf is able to transform an image into a different one preserving the amount of noise of the starting image.*

The experiments also show that moving orthogonally to the data leaf we find noisy images. This means that the other leaves of the foliation contain images with a level of noise that increases with the distance from the data leaf. These noisy images become soon meaningless to the human eye, while the model still classifies them with high confidence. This fact is a consequence of the property of the local data matrix: (3.2) prescribes that the model output does not change if we move in a direction orthogonal to the tangent space of the leaf on which our data are located.

This remark points us to other possible applications of the model-centric data manifold. We could project a noisy point on the data leaf to perform denoising, or we could use the distance from the data leaf to recognize out-of-distribution examples. Such applications require further research on the characterization of the model-centric data manifold.

The main contributions of the paper are

1. the definition of the local data matrix $G(x, w)$ at a point x of the data domain and for a given model w and the study of its properties;

2. the proof that the subspace spanned by the eigenvectors with a nonzero eigenvalue of the local data matrix $G(x, w)$ can be interpreted as the tangent space of a Riemannian manifold, whose dimension is bounded by the number of classes on which our model is trained;

3. the identification and visualization of the model-centric data manifold through paths, obtained via experiments on MNIST.

Organization of the paper. In section 2, we review the fundamentals of information geometry using a novel perspective that aims at facilitating the comprehension of the key concepts of the paper. We introduce the *local data matrix* $G(x, w)$, and we summarize its properties in Proposition 2.1. In section 3, we show that, through the local data matrix, under

some mild hypotheses, the data domain foliates as a disjoint union of leaves, which are all Riemannian submanifolds of $\mathbf{R}^n$, with a metric given via $G(x,w)$. In section 4, we provide evidence that all our dataset lies on one leaf of the foliation and that moving along directions orthogonal to the data leaf amounts to adding noise to data.

2. Information geometry. Here we collect some results pertaining to information geometry [3, 27, 17], using a novel perspective adapted to our question, namely, how to provide a manifold structure to the space containing data.

Let $p(y|x,w)$ be a discrete probability distribution on C classification labels, i.e., $p(y|x,w) = (p_i(y|x,w))_{i=1,\dots,C}$, $x \in \Sigma \subset \mathbf{R}^n$, $w \in \mathbf{R}^d$. In the applications, x represents input data belonging to a certain dataset Σ, while w is the learning parameters, i.e., the parameters of the empirical model. As we are going to see in our discussion later on, it is fruitful to treat the two sets of variables x and w on equal grounds. This will naturally lead to a geometric structure on a low-dimensional submanifold of $\mathbf{R}^n$ that we can navigate through paths joining points in the dataset Σ (see section 4).

In order to give some context to our treatment, we define, following [3, section 3], the *information loss* $I(x,w) = -\log(p(y|x,w))$ and the *loss function* $L(x,w) = \mathbf{E}_{y\sim q}[I(x,w)]$. Typically $L(x,w)$ is used for practical optimizations, where we need to compare the model output distribution $p(y|x,w)$ with a certain known true distribution $q(y|x)$. We may also view $L(x,w)$ as the Kullback–Leibler divergence between $p(y|x,w)$ and $q(y|x)$ up to the constant $-\sum_i q_i(y|x)\log q_i(y|x)$, irrelevant for any optimization problem:

$$\begin{aligned} L(x,w) &= \mathbf{E}_{y\sim q}[-\log(p(y|x,w))] = \\ &= \sum_{i=1}^{C} q_i(y|x)\log\frac{q_i(y|x)}{p_i(y|x,w)} - \sum_{i=1}^{C} q_i(y|x)\log q_i(y|x) \\ &= \mathrm{KL}(q(y|x)||p(y|x,w)) - \sum_{i=1}^{C} q_i(y|x)\log q_i(y|x). \end{aligned} \tag{2.1}$$

A popular choice for $p(y|x,w)$ in deep learning classification algorithms is

$$p_i(y|x,w) = \mathrm{softmax}(s(x,w))_i = \frac{e^{s_i(x,w)}}{\sum_{j=1}^{C} e^{s_j(x,w)}}, \tag{2.2}$$

where $s(x,w) \in \mathbf{R}^C$ is a score function determined by parameters w. From such $p(y|x,w)$ we derive the cross-entropy with softmax loss function:

$$\begin{aligned} L(x,w) &= \mathbf{E}_{y\sim q}[I(x,w)] = \mathbf{E}_{y\sim q}[-\log p(y|x,w)] \\ &= -s_{y_x}(x,w) + \log\sum_{j=1}^{C} e^{s_j(x,w)}, \end{aligned} \tag{2.3}$$

where $L(x,w)$ is computed with respect to the probability mass distribution $q(y|x)$ assigning 1 to the correct label y_x of our datum x and zero otherwise.

Let us now define the following two matrices:

$$F(x,w) = \mathbf{E}_{y\sim p}[\nabla_w \log p(y|x,w) \cdot (\nabla_w \log p(y|x,w))^T], \tag{2.4}$$

$$G(x,w) = \mathbf{E}_{y\sim p}[\nabla_x \log p(y|x,w) \cdot (\nabla_x \log p(y|x,w))^T]. \tag{2.5}$$

We call $F(x,w)$ the *local Fisher matrix* at the datum x and $G(x,w)$ the *local data matrix* given the model w. The Fisher matrix [3] is obtained as $F(w) = \mathbf{E}_{x\sim\Sigma}[F(x,w)]$, and it gives information on the metric structure of the space of parameters. Similarly, we can reverse our perspective and see how $G(x,w)$ allows us to recognize some structure in our dataset.

The following observations apply to both $F(x,w)$ and $G(x,w)$ and provide the theoretical cornerstone of section 3.

Proposition 2.1. *Let the notation be as above. Then,*

1. $F(x,w)$ *and* $G(x,w)$ *are positive semidefinite symmetric matrices;*
2. $\ker F(x,w) = (\text{span}_{i=1,\dots,C}\{\nabla_w \log p_i(y|x,w)\})^\perp$; $\ker G(x,w) = (\text{span}_{i=1,\dots,C}\{\nabla_x \log p_i(y|x,w)\})^\perp$;
3. $\text{rank } F(x,w) < C$, $\text{rank } G(x,w) < C$.

Proof. We prove the results for $F(x,w)$; the proofs for $G(x,w)$ are akin.

1. It is immediate to see that $F(x,w)$ is symmetric because it is a weighted sum of symmetric matrices. To see that it is positive semidefinite we check that the property $u^T F(x,w)u \geq 0$ is satisfied $\forall u \in \mathbf{R}^n$.

$$\begin{aligned} u^T F(x,w)u &= \mathbf{E}_{y\sim p}\left[u^T \nabla_w \log p(y|x,w)(\nabla_w \log p(y|x,w))^T u\right] \\ &= \mathbf{E}_{y\sim p}\left[\langle \nabla_w \log p(y|x,w), u\rangle^2\right] \geq 0. \end{aligned} \tag{2.6}$$

2. We show that $\ker F(x,w) \subseteq (\text{span}_{i=1,\dots,C}\{\nabla_w \log p_i(y|x,w)\})^\perp$:

$$\begin{aligned} u \in \ker F(x,w) &\Rightarrow u^T F(x,w)u = 0 \Rightarrow \mathbf{E}_{y\sim p}\left[\langle \nabla_w \log p(y|x,w), u\rangle^2\right] = 0 \\ &\Rightarrow \langle \nabla_w \log p_i(y|x,w), u\rangle = 0 \quad \forall\ i = 1,\dots,C. \end{aligned} \tag{2.7}$$

On the other hand, if $u \in (\text{span}_{i=1,\dots,C}\{\nabla_w \log p_i(y|x,w)\})^\perp$, then $u \in \ker F(x,w)$:

$$F(x,w)u = \mathbf{E}_{y\sim p}\left[\nabla_w \log p_i(y|x,w)\langle \nabla_w \log p_i(y|x,w), u\rangle\right] = 0. \tag{2.8}$$

3. From Proposition 2.1, we know that $\text{rank } F(x,w) \leq C$. The vectors $\nabla_w \log p_i(y|x,w)$ are linearly dependent since

$$\begin{aligned} \mathbf{E}_{y\sim p}[\nabla_w I(x,w)] &= \sum_{i=1}^{C} p_i(y|x,w)\nabla_w \log p_i(y|x,w) \\ &= \sum_{i=1}^{C} \nabla_w p_i(y|x,w) = \nabla_w\left(\sum_{i=1}^{C} p_i(y|x,w)\right) = \nabla_w 1 = 0. \end{aligned} \tag{2.9}$$

Therefore, we deduce $\text{rank } F(x,w) < C$. ∎

This result tells us that the rank of both $F(x,w)$ and $G(x,w)$ is bounded by C, the number of classes in our classification problem. The bound on rank $G(x,w)$ will allow us to define a submanifold of $\mathbf{R}^n$ of dimension rank $G(x,w)$ (section 3) that our experiments show contains our dataset (section 4) In practical situations, this dimension is much lower than the size of $G(x,w)$, i.e., the input size n, as shown in Table 1.

Table 1
Bound on the rank of $G(x,w)$ for popular image classification tasks.

Dataset	$G(x,w)$ size	rank $G(x,w)$ bound
MNIST	784	10
CIFAR-10	3072	10
CIFAR-100	3072	100
ImageNet	150528	1000

3. The model view on the data manifold. We now turn to examine some properties of the matrices $F(x,w)$ and $G(x,w)$ that will enable us to discover a submanifold structure on the portion of $\mathbf{R}^n$ occupied by our dataset (see section 4) and to prove the claims at the end of the above section.

We recall that, given a perturbation of the weights w, the Kullback–Leibler divergence, gives, to second order approximation, the following formula:

$$\begin{aligned} &\mathrm{KL}(p(y|x,w+\delta w)||p(y|x,w)) \\ &= (\delta w)^T F(x,w)(\delta w) + \mathcal{O}(||\delta w||^3). \end{aligned} \tag{3.1}$$

Equation (3.1), together with Proposition 2.1, effectively expresses the fact that, during stochastic gradient descent dynamics with a minibatch of size 1, we have only a very limited number of directions, namely, $C-1$, in which the change δw affects the loss.

Taking the expectation with respect to $x \sim \Sigma$ on both sides of the equation, we obtain the analogous property for the Fisher matrix $F(w) = \mathbf{E}_{x\sim\Sigma}[F(x,w)]$. While $F(x,w)$ has a low rank, the rank of $F(w)$ is bounded by $|\Sigma|(C-1)$, a number that is often higher than the size of $F(w)$. Thus $F(w)$, when nondegenerate, is an effective metric on the parameter space. It allows us to measure, according to a certain step δw, when we reach a stable predicted probability and thus the end of model training (see [25] and references within).

It must be noted, however, that $F(w)$ is meaningful *away* from the trained model, that is, well before the end of the training phase (see [2] for an empirical validation of such statements). We are going to see, with our experiments in section 4, that a similar phenomenon occurs for $G(x,w)$.

We now turn to the local data matrix $G(x,w)$, thus interpreting (3.1) in the data domain $\mathbf{R}^n$. For a perturbation δx of the data x, we have, up to second order approximation,

$$\begin{aligned} &\mathrm{KL}(p(y|x+\delta x,w)||p(y|x,w)) \\ &= (\delta x)^T G(x,w)(\delta x) + \mathcal{O}(||\delta x||^3). \end{aligned} \tag{3.2}$$

This equation tells us that, if we move along the directions of $\ker G(x,w)$, the probability distribution $p(y|x,w)$ is constant (up to a second order approximation), while our data are changing. Those are the vast majority of the directions since $\operatorname{rank} G(x,w) < C$ and typically $C \ll n$; hence, we interpret them as the *noise directions*. On the other hand, if we move from a data point x along the directions in $(\ker G(x,w))^{\perp}$, data will change along with the probability distribution $p(y|x,w)$ associated with it. These are the directions going toward data points that the model classifies with confidence.

Equation (3.2) suggests to view $G(x,w)$ as a metric on the data domain. However, because of its low rank (see Proposition 2.1), we need to restrict our attention to the subspace $(\ker G(x,w))^\perp$, where $G(x,w)$ is nondegenerate. $G(x,w)$ allows us to define a *distribution* $\mathcal{D}$ on $\mathbf{R}^n$. In general, in differential geometry, we call distribution on $\mathbf{R}^n$ an assignment:

$$x \mapsto \mathcal{D}_x \subset \mathbf{R}^n \qquad \forall x \in \mathbf{R}^n,$$

where $\mathcal{D}_x$ is a vector subspace of $\mathbf{R}^n$ of a fixed dimension k.

Assume now $G(x,w)$ has constant rank; as we shall see in our experiments, this is the case for a non–fully trained model. We thus obtain a *distribution* $\mathcal{D}$:

$$x \mapsto \mathcal{D}_x = (\ker G(x,w))^\perp \subset \mathbf{R}^n. \tag{3.3}$$

We now would like to see if our distribution (3.3) defines a *foliation structure*. This means that we can decompose $\mathbf{R}^n$ as the disjoint union of submanifolds, called *leaves* of the foliation, and there is a *unique* submanifold (leaf) going through each point x. The distribution comes into the play because it gives the tangent space to the leaf through x: $\mathcal{D}_x = (\ker G(x,w))^\perp$. In this way, moving along the directions in $\mathcal{D}_x$ at each point x will produce a path lying in one of the submanifolds (leaves) of the foliation.

The existence of a foliation, whose leaf through a point x has tangent space $\mathcal{D}_x$, comes through the Frobenius theorem, which we recall here in the version that we need. Notation is as follows: as usual $[X,Y]$ denotes the Lie bracket of vectors in $\mathbf{R}^n$ interpreted as vector fields, i.e., derivations; hence $[X,Y] := X \circ Y - Y \circ X$.

Frobenius theorem. *Let $x \in \mathbf{R}^n$, and let $\mathcal{D}$ be a distribution in $\mathbf{R}^n$. Assume that in a neighborhood U of x*

$$[X,Y] \in \mathcal{D} \qquad \forall\ X, Y \in \mathcal{D}. \tag{3.4}$$

Then, there exists a (local) submanifold $N \subset \mathbf{R}^n$, $x \in N$, such that $T_z N = \mathcal{D}_z\ \forall z \in N$.

It is not reasonable to expect that a general classifier satisfies the involutive property (3.4); however, it is remarkable that for a large class of classifiers, namely, deep ReLU neural networks, this is the case, with p given by softmax as in (2.2).

Theorem 3.1. *Let w be the weights of a deep ReLU neural network classifier, p be given by softmax, and $G(x,w)$ be the local data matrix. Assume $G(x,w)$ has constant rank. Then, there exists a local submanifold $N \subset \mathbf{R}^n$, $x \in N$, such that its tangent space at z, $T_z N = (\ker G(z,w))^\perp\ \forall z \in N$.*

Proof. We first notice that the points in which the local data matrix is nondifferentiable are a closed subset in $\mathbf{R}^n$; in fact they consist of a union of hyperplanes. Since the result we want to prove is local, we may safely assume that our point x has a neighborhood where we can compute $\nabla_x \log p_i(y|x,w)$ and in particular $\nabla_x s_k(x,w)$.

By the Frobenius theorem, we need to check the involutivity property (3.4) for the distribution $x \mapsto \mathcal{D}_x = (\ker G(x,w))^\perp$ on $\mathbf{R}^n$. Since, by Proposition 2.1,

$$\mathcal{D}_x = (\ker G(x,w))^\perp = \operatorname{span}_{i=1,\dots,C}\{\nabla_x \log p_i(y|x,w)\},$$

we only need to show that

$$[\nabla_x \log p_i(y|x,w), \nabla_x \log p_j(y|x,w)] \in \text{span}_{k=1,\dots,C}\{\nabla_x \log p_k(y|x,w)\}.$$

By standard computations, we see that

$$\begin{aligned}[\nabla_x \log p_i(y|x,w), \nabla_x \log p_j(y|x,w)] = \mathbf{H}(\log p_i(y|x,w))\nabla_x \log p_j(y|x,w) + \\ - \mathbf{H}(\log p_j(y|x,w))\nabla_x \log p_i(y|x,w),\end{aligned} \tag{3.5}$$

where $\mathbf{H}(f)$ denotes the Hessian of a function f. Here we are using the fact that

$$\left[\sum_i a\partial_i, \sum_j b_j\partial_j\right] = \sum_{i,j}(a_i\partial_i b_j - b_i\partial_i a_j)\partial_j.$$

With some calculations, we have

$$\mathbf{H}(\log p_i(y|x,w)) = \frac{\mathbf{H}(p_i(y|x,w))}{p_i(y|x,w)} - \nabla_x \log p_i(y|x,w) \cdot (\nabla_x \log p_i(y|x,w))^T \tag{3.6}$$

and

$$\nabla_x p_i(y|x,w) = \sum_{k=1}^{C} p_i(y|x,w)(\delta_{ik} - p_k(y|x,w))\nabla_x s_k(x,w), \tag{3.7}$$

where s is the score function and we make use of the fact $p_i(y|x,w)$ is given by softmax. Notice $\partial_{s_k} p_i(y|x,w) = p_i(y|x,w)(\delta_{ik} - p_k(y|x,w))$, where δ_{ik} is the Kronecker delta. By (3.6) and (3.7),

$$\begin{aligned}\mathbf{H}(p_i(y|x,w)) &= \text{Jac}(\nabla_x p_i(y|x,w)) \\ &= \sum_{k=1}^{C} \nabla_x \left[p_i(y|x,w)(\delta_{ik} - p_k(y|x,w))\right](\nabla_x s_k(x,w))^T,\end{aligned} \tag{3.8}$$

where we use the fact that the Hessian of s_k is zero, where it is defined. In fact, s_k is piecewise linear since the activation function ReLU is piecewise linear. Hence,

$$\begin{aligned}\mathbf{H}(p_i(y|x,w)) = \frac{\nabla_x p_i(y|x,w)(\nabla_x p_i(y|x,w))^T}{p_i(y|x,w)} + \\ - p_i(y|x,w)\sum_{k=1}^{C} \nabla_x p_k(y|x,w)(\nabla_x s_k(x,w))^T.\end{aligned} \tag{3.9}$$

Now, in view of (3.5), using (3.6) and (3.9) we compute the expression

$$\begin{aligned}\mathbf{H}(\log p_i(y|x,w)) = &\frac{\nabla_x p_i(y|x,w)}{p_i(y|x,w)}\left(\frac{\nabla_x p_i(y|x,w)}{p_i(y|x,w)}\right)^T + \\ &- \sum_{k=1}^{C} \nabla_x p_k(y|x,w)(\nabla_x s_k(x,w))^T + \\ &- \nabla_x \log p_i(y|x,w)\cdot(\nabla_x \log p_i(y|x,w))^T \\ &= -\sum_{k=1}^{C} \nabla_x p_k(y|x,w)(\nabla_x s_k(x,w))^T.\end{aligned} \tag{3.10}$$

Hence,

$$\begin{aligned}&\mathbf{H}(\log p_i(y|x,w))\nabla_x \log p_j(y|x,w) = \\ &\quad -\sum_{k=1}^{C} \nabla_x p_k(y|x,w)(\nabla_x s_k(x,w))^T \nabla_x \log p_j(y|x,w).\end{aligned} \tag{3.11}$$

Since $(\nabla_x s_k(x,w))^T \nabla_x \log p_j(y|x,w)$ is a scalar, we obtain that $\mathbf{H}(\log p_i(y|x,w))$ $\nabla_x \log p_j(y|x,w)$ is a linear combination of the vectors $\nabla_x p_k(y|x,w)$ and therefore a linear combination of the vectors $\nabla_x \log p_k(y|x,w)$.

The same holds for $\mathbf{H}(\log p_j(y|x,w))$ $\nabla_x \log p_i(y|x,w)$; then also $[\nabla_x \log p_i(y|x,w)$, $\nabla_x \log p_j(y|x,w)]$ is a linear combination of vectors in $\mathcal{D}_x$ as we wanted to show.

Through the application of Frobenius theorem, Theorem 3.1 gives us a foliation of the data domain $\mathbf{R}^n$. $\mathbf{R}^n$ decomposes into the disjoint union of $C-1$-dimensional submanifolds, whose tangent space at a fixed $x \in \mathbf{R}^n$ is $\mathcal{D}_x = (\ker G(x,w))^{\perp}$. Every point x determines a unique submanifold corresponding to the leaf of the foliation through x. We may extend this local submanifold structure to obtain a global structure of manifold on a leaf, still retaining the above property regarding the distribution.

As we shall see in section 4, we can move from a point x in our dataset Σ to another point x' also in Σ with a *horizontal* path, that is, a path tangent to $\mathcal{D}_x$, hence lying on the leaf of x and x'. Our experiments show that we can connect every pair of points $(x,x') \in \Sigma \times \Sigma$ with horizontal paths. It means that all the dataset belongs to a single leaf, which we call the *data leaf* $\mathcal{L}$. Our model, through the local data matrix $G(x,w)$, enables us to move on the low-dimensional submanifold $\mathcal{L}$, to which all of our dataset belongs. Of course not all the points of $\mathcal{L}$ correspond to elements of the dataset; however, as we show in the experiments, on most points of $\mathcal{L}$ the model gives prediction compatible with human observers.

We also notice that each leaf of our foliation comes naturally equipped with a metric given at each point by the matrix $G(x,w)$, restricted to the subspace $(\ker G(x,w))^{\perp}$, which coincides with the tangent space to the leaf, where $G(x,w)$ is nondegenerate. Hence $G(x,w)$ will provide each leaf with a natural Riemannian manifold structure. We end this section with an observation, comparing our approach to the geometry of the data domain, with the parameter space.

Remark. Equation (3.1) provides a metric to the parameter space $\mathbf{R}^d$, motivating our approach to the data domain. For each $w \in \mathbf{R}^d$, we can define, as we did for the data domain,

a distribution $w \mapsto \mathcal{D}'_w := (\ker F(w))^{\perp}$, using the Fisher matrix. However, it is easy to see empirically that this distribution is *not involutive*; i.e., there is no foliation and no submanifold corresponding to it.

4. Experiments. We performed experiments on the MNIST dataset [23] and on the CIFAR-10 dataset [21]. We report in this section the experiments on the MNIST dataset only because they are easier to interpret in the geometrical framework (foliation and leaves) introduced in our previous section. CIFAR-10 experiments are reported in the dedicated Appendix A.

All the following experiments use the same simple CNN classifier trained on MNIST. The neural network is similar to LeNet, with 32 and 64 channels in the two convolutional layers and 128 hidden units in the fully connected layer. The architecture is identical to the one proposed in the official MNIST example of the PyTorch framework [29]. The network has ReLU activation function, so it satisfies the hypothesis of Theorem 3.1. We train this network with SGD, a batch size of 60, and a fixed learning rate of 0.01.

4.1. Rank and trace of the local data matrix. The definition of distribution in section 3 and the consequent results require the rank of $G(x,w)$ to be constant on every point x for a certain parameter configuration w. We remark that the rank of $G(x,w)$ can be at most $C-1$ (Proposition 2.1), i.e., much lower than the size of $G(x,w)$ (Table 1).

To check if our assumption is satisfied by a model, we should calculate the rank of several local data matrices varying the data point x. Unfortunately, it is very difficult to establish the rank of a matrix in numerical experiments because the rank is not robust with respect to small perturbations. In fact, if we perform the calculation, the numerically estimated rank is always higher than the theoretical upper bound $C-1$. Knowing that the rank corresponds to the ℓ_0 norm of the eigenvalues of a matrix, we can consider the ℓ_1 norm of the eigenvalues as a soft version of the rank. For a positive semidefinite symmetric matrix like $G(x,w)$ (see Proposition 2.1), the ℓ_1 norm of the eigenvalues corresponds to the trace of the matrix.

Figure 2 shows the trend of the mean trace of $G(x,w)$ for a fixed set of 1000 images during the training. The plot is similar to the ones reported in [2] for the Fisher information matrix, and it tells us that the local data matrix $G(x,w)$ loses its informative content at the end of training as well. When the model reaches the convergence the trace of $G(x,w)$ is almost null for many data points x, and the rank can be different for different x, breaking the hypothesis of the theorem. We could expect the observed decreasing trend of the trace since

$$\begin{aligned} \operatorname{Tr} G(x,w) &= \operatorname{Tr} \mathbf{E}_{y\sim p}[\nabla_x \log p(y|x) \cdot (\nabla_x \log p(y|x))^T] \\ &= \mathbf{E}_{y\sim p}[\operatorname{Tr}(\nabla_x \log p(y|x,w) \cdot (\nabla_x \log p(y|x,w))^T)] \\ &= \mathbf{E}_{y\sim p}[||\nabla_x \log p(y|x,w)||^2]. \end{aligned} \tag{4.1}$$

In fact a small perturbation of the data point x should not influence the prediction of a fully trained model; thus $||\nabla_x \log p_i(y|x,w)||^2$ should be almost null for every point x at the end of training.

All these observations motivate us to perform subsequent experiments on a partially trained model. In particular, we use the checkpoint at step 10000, i.e., at the end of the 10th epoch.

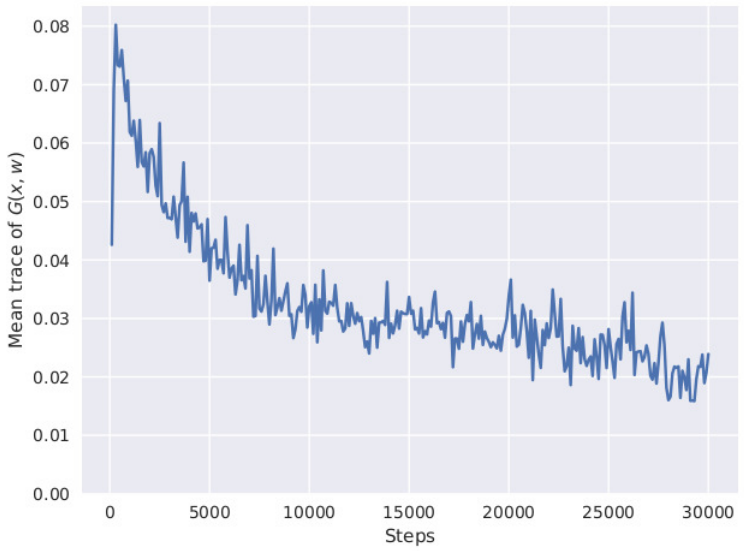

Figure 2. *Average trace of* $G(x, w)$ *computed on* 1000 *fixed data points* x *during training.*

4.2. Characterization of leaves. Since our neural network classifier satisfies all the hypothesis of Theorem 3.1, it views the data domain $\mathbf{R}^n$ as a *foliation*. Nevertheless, this result does not give us any clue about the characterization of leaves. We know, however, that, moving away from a data point x while remaining on the same leaf, we can obtain points with different labels, while moving in a direction orthogonal to the tangent space to the leaf of x, we obtain points with the same label as x, as long as the estimate (3.2) holds.

To understand the distinguishing factors of a leaf, we move from a data point x across leaves, and we inspect the crossed data points. More specifically, we start from an image in MNIST test set, and we let it evolve moving orthogonally to the tangent space $\mathcal{D}_{x_t} = (\ker\ G(x_t, w))^{\perp}$. Since the dimension of the orthogonal space is $n - C + 1$, there are many possible directions. Thus, to monotonically increase the distance from the starting point x, we use a fixed random direction, and at every step we project it on $\ker G(x_t, w)$, i.e., on the orthogonal space of the tangent space of current leaf.

We observe in Figure 3 that the noise in the images increases steadily. Moreover, we can see that, as predicted by (3.2), model predictions remain very certain even when the digit is indistinguishable for the human eye. This experiment makes us speculate that a leaf is characterized by a constant amount of noise. If that is the case, all valid data should reside on the same leaf characterized by the absence of noise, the *data leaf*.

4.3. Horizontal paths on leaves. We follow horizontal paths connecting two images from MNIST test set. If it is possible to join two inputs with a horizontal path, then we know that those points are on the same leaf. In our case a horizontal path is tangent to the distribution $\mathcal{D}$ described in the previous section, i.e., $\mathcal{D}_x = (\ker G(x, w))^{\perp} = \mathrm{span}_{i=1,\dots,C}\{\nabla_w \log p_i(y|x, w)\}$ from Proposition 2.1. We remark that $\mathcal{D}_x$ coincides with the row space of the Jacobian

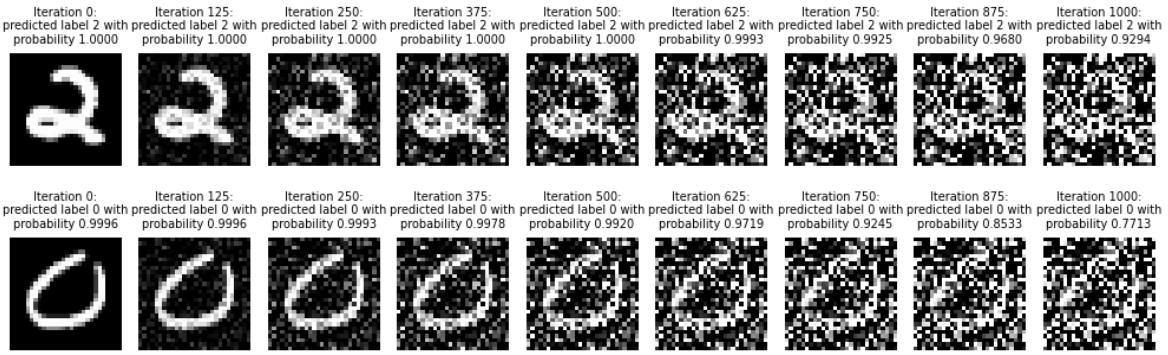

Figure 3. *Paths across leaves of the foliation. Notice how the images become increasingly indistinguishable from noise, while the classifier continues to assign a high probably to the class of the initial image. The further we move orthogonally to the data leaf, the more noise appears on the image.*

matrix $\mathrm{Jac}_x \log p(y|x,w)$. We use this equivalence to compute the projection of a vector on the distribution $\mathcal{D}_x$.

Algorithm 4.1 Find a horizontal path between points.

Input: source s, destination d, step size α, number of iterations T, model parameters w
$x_0 = s$
for $t = 1, \ldots, T$ **do**
  # Calculate Jacobian of $\log p(y|x_{t-1}, w)$ with respect to x_{t-1}
  $j = \mathrm{Jac}_x \log p(y|x_{t-1}, w)$
  # Project the gradient of $||d - x_{t-1}||^2$ on $\mathcal{D}_{x_{t-1}}$)
  $v = \mathrm{projection}(d - x_{t-1}, j)$
  # Update x_{t-1} to obtain the next point x_t on the path
  $x_t = x_{t-1} + \alpha \frac{v}{||v||}$
end for
Output: $\{x_t\}_{t=0,\ldots,T}$

Algorithm 4.1 finds an approximate horizontal path from a source point to a destination point. The algorithm is obtained as a simplification of Riemannian gradient descent [13] using the squared Euclidean distance from the destination point as a loss function. Since we do not have an explicit characterization of the leaves, we cannot perform the retraction step commonly used in optimization algorithms on manifolds. To circumvent this problem, we normalize the gradient vector v and use a small step size $\alpha = 0.1$ in our experiments. The normalization ensures that the norm of the displacement at every step is controlled by the step size.

Figure 4 shows the results of the application of Algorithm 4.1 on some pairs of images in MNIST test set. We observe that it is possible to link different images with a horizontal path, confirming our conjecture about the existence of the data leaf. This experiment shows that the model sees the data on the same manifold, namely, one leaf of the foliation determined by our distribution $\mathcal{D}$.

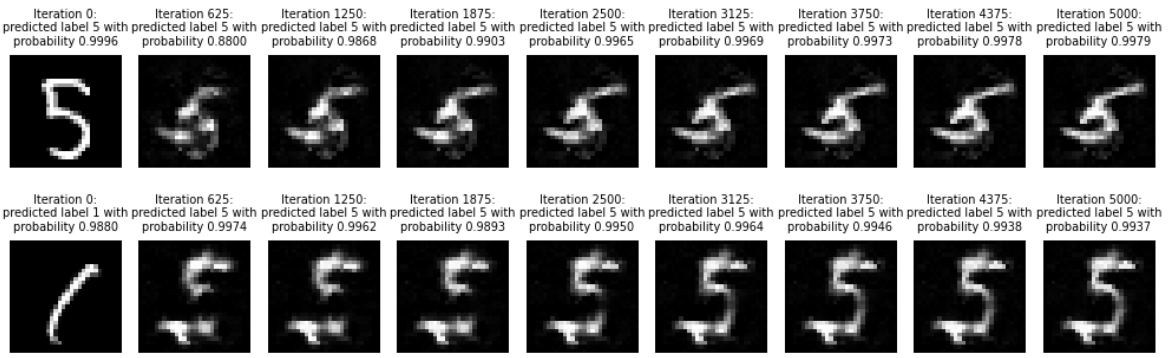

Figure 4. *Horizontal paths between two images in MNIST test set. Here, the paths require at most* 5000 *steps; different source-destination pairs can need more steps and a longer path. The sequences show that a horizontal path is very different from an interpolation. Any two images in MNIST appear to be connected by a horizontal path, i.e., by a path laying on one leaf.*

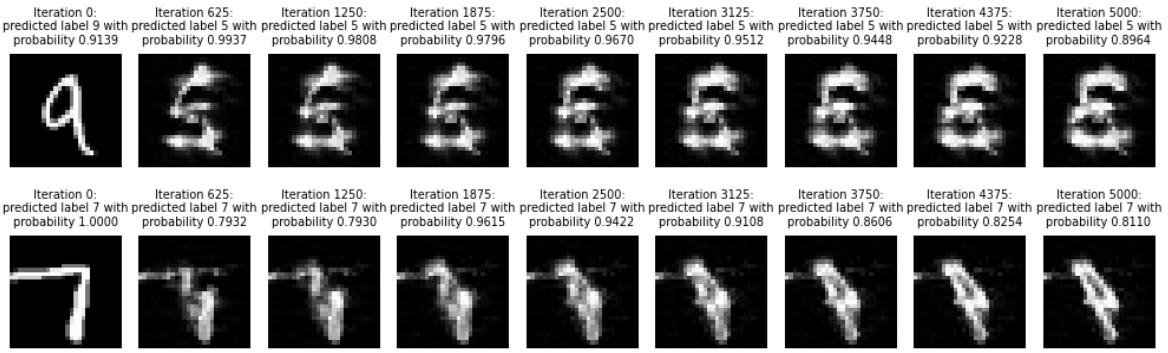

Figure 5. *Horizontal paths between a valid image and a mirrored image from MNIST test set. The mirrored images are chosen to resemble letters (E and P). Notice how in the first sequence the path passes through the digit* 5 *before reaching the destination point. Also images not in MNIST appear to be connected by a horizontal path.*

The observation of the paths on the data leaf gives us a novel point of view on the generalization property of the model. First of all, while we could expect to find all training data on the same leaf, it is remarkable that test data are placed on the same leaf too. Furthermore, we observe that the data leaf is not limited to digits and transition points between them. In fact, we can find paths connecting valid images with images of nonexistent digits similar to letters, as shown in Figure 5. This fact suggests that the model-centric data manifold is somewhat more general than the classification task on which the model is trained.

Our final experiment gives us another remarkable confirmation of our theoretical findings: it is impossible to link with a horizontal path a noisy image outside the data leaf and a valid data image. A noisy image is generated using the same strategy presented in section 4.2, i.e., modifying a valid image along a direction in $\ker G(x, w)$. Figure 6 shows that, even after 10000 iterations of Algorithm 4.1, it is impossible to converge to the valid destination point. Indeed,

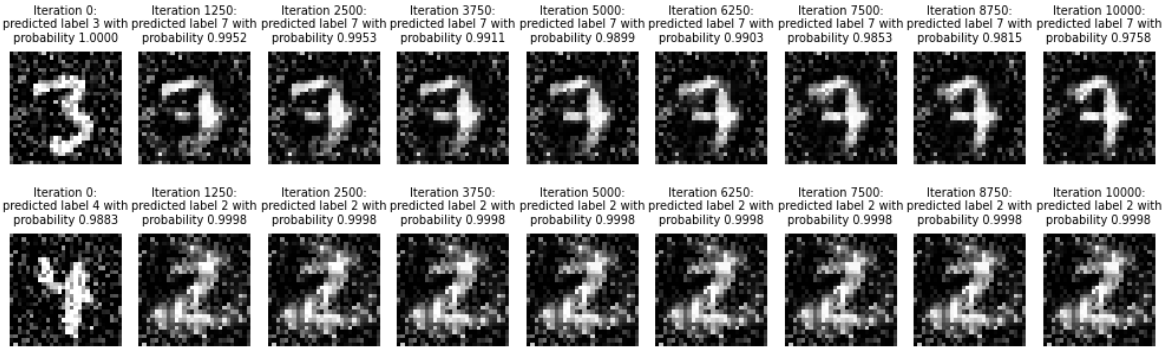

Figure 6. *Horizontal paths unable to reach a valid image in MNIST test set from a noisy image.*

the noise is preserved along the path, and the noise pattern stabilizes after 5000 iterations. The final point reached is a noisy version of the actual destination point.

All those experiments confirm that there exists a model-centric data manifold, the data leaf of the foliation. In addition, they exhibit that the leaves in the foliation of the data domain are characterized by the noise content.

5. Related works.

Fisher information matrix. In [2] the authors discuss the information content of the Fisher matrix, and they show that such content changes during the training of a neural network, decreasing rapidly toward the end of it. This shows that the Fisher–Rao metric acquires importance during the training phase only (see also [20]). More on the geometry of the parameter space, as a Riemannian manifold, is found in [32]. In this paper, we take an analogue of the Fisher–Rao metric but on the data domain. In our examples, we see a phenomenon similar to the one observed in [2]: the trace of the local data matrix decreases rapidly as the model completes its training. Other metrics on datasets were suggested (for example, see [10] and references within) but with different purposes. Here, our philosophy is the same as in [6]: we believe that data themselves are not equipped with a geometric structure, but such structure emerges only when the model views data, with a given classification task.

Intrinsic dimension. [4] measures the intrinsic dimension of layer representations for many common neural network architectures. At the same time, they measure the intrinsic dimension of MNIST, CIFAR-10 [21], and ImageNet [8] datasets. Our objective is similar, but we do not specifically quantify the dimension of the data manifold. The data leaf reflects how a classifier sees a geometric structure on the discrete data points from a dataset. Its dimension is intimately linked to the classification task. For this reason the results are not directly comparable.

[4] shows that MNIST and neural networks trained on it behave very differently from networks trained on CIFAR-10 or ImageNet. While our experiments focus on MNIST, additional experiments on CIFAR-10 are shown in Appendix A.

Adversarial attacks. Our method to navigate the leaves of the foliation is very similar to common adversarial attack methods, like fast gradient sign method [15] or projected gradient descent. Adversarial attacks and our navigation algorithm both rely on gradients $\nabla_x \log p_i$, but adversarial generation algorithms perturb the original image by $\text{sign}(\nabla_x \log p_i)$. In general,

$\operatorname{sign}(\nabla_x \log p_i) \notin \ker(G(x,w))^{\perp}$, so adversarial examples are created perturbing the image outside the data leaf.

6. Conclusions. In this paper, we introduce the local data matrix, a novel mathematical object that sheds light on the internal workings of a neural network classifier. We prove that the model organizes the data domain according to the geometric structure of a foliation. Experiments show that valid data are placed on the same leaf of the foliation; thus the model sees the data on a low-dimensional submanifold of the data domain. Such a submanifold appears more general than the model itself because it includes meaningless but visually similar images together with training and test data.

In the future, we aim to characterize the data leaf and to study the Riemannian metric given by the local data matrix. If we could analytically characterize the leaves of the foliation by the degree of noise, we could distinguish noisy data from valid examples. That can be used in the inference phase to exclude examples on which model predictions are not reliable. Furthermore, it could pave the way to the creation of novel generic denoising algorithms applicable to every kind of data.

Appendix A. CIFAR-10 experiments.

We perform some experiments with the CIFAR-10 dataset (shown in Figure 7) and a neural network similar to VGG-11. The horizontal paths found are hard to interpret, and they are very similar to interpolations.

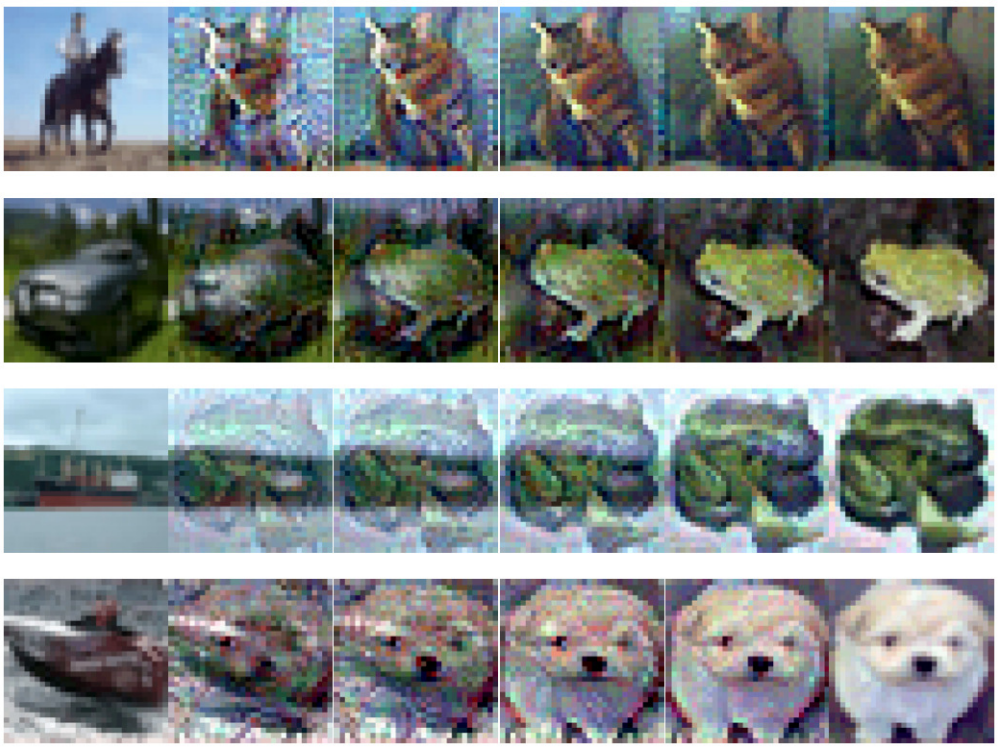

Figure 7. *Horizontal paths between images in CIFAR-10 test set.*

Acknowledgments. We would like to thank Prof. S. Soatto and Dr. A. Achille for helpful comments and discussions. We also would like to thank our anonymous referees for helpful comments.

REFERENCES

[1] A. G. Ache and M. W. Warren, *Ricci curvature and the manifold learning problem*, Adv. Math., 342 (2019), pp. 14–66.
[2] A. Achille, M. Rovere, and S. Soatto, *Critical learning periods in deep networks*, in Proceedings of the 7th International Conference on Learning Representations, OpenReview.net, 2019.
[3] S.-I. Amari, *Natural gradient works efficiently in learning*, Neural Comput., 10 (1998), pp. 251–276.
[4] A. Ansuini, A. Laio, J. H. Macke, and D. Zoccolan, *Intrinsic dimension of data representations in deep neural networks*, in Advances in Neural Information Processing Systems 32, Curran Associates, Red Hook, NY, 2019, pp. 6111–6122.
[5] M. Arjovsky and L. Bottou, *Towards principled methods for training generative adversarial networks*, in Proceedings of the 5th International Conference on Learning Representations, OpenReview.net, 2017.
[6] M. G. Bergomi, P. Frosini, D. Giorgi, and N. Quercioli, *Towards a topological–geometrical theory of group equivariant non-expansive operators for data analysis and machine learning*, Nature Mach. Intell., 1 (2019), pp. 423–433.
[7] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, *Natural language processing (almost) from scratch*, J. Mach. Learn. Res., 12 (2011), pp. 2493–2537.
[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, *ImageNet: A large-scale hierarchical image database*, in Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 248–255.
[9] P. Dollár, V. Rabaud, and S. Belongie, *Learning to traverse image manifolds*, in Advances in Neural Information Processing Systems 19, MIT Press, Cambridge, MA, 2007, pp. 361–368.
[10] Y. Dukler, W. Li, A. Lin, and G. Montufar, *Wasserstein of Wasserstein loss for learning generative models*, in Proceedings of the 36th International Conference on Machine Learning, Proc. Mach. Learn. Res. (PMLR) 97, JMLR, 2019, pp. 1716–1725.
[11] C. Fefferman, S. Mitter, and H. Narayanan, *Testing the manifold hypothesis*, J. Amer. Math. Soc., 29 (2016), pp. 983–1049.
[12] N. Frosst, N. Papernot, and G. E. Hinton, *Analyzing and improving representations with the soft nearest neighbor loss*, in Proceedings of the 36th International Conference on Machine Learning, Proc. Mach. Learn. Res. (PMLR) 97, JMLR, 2019.
[13] D. Gabay, *Minimizing a differentiable function over a differential manifold*, J. Optim. Theory Appl., 37 (1982), pp. 177–219.
[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial nets*, in Advances in Neural Information Processing Systems 27, Curran Associates, Red Hook, NY, 2015, pp. 2672–2680.
[15] I. J. Goodfellow, J. Shlens, and C. Szegedy, *Explaining and harnessing adversarial examples*, in Proceedings of the 3rd International Conference on Learning Representations, Schloss Dagstuhl, 2015.
[16] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al., *Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups*, IEEE Signal Process. Mag., 29 (2012), pp. 82–97.
[17] R. Karakida, S. Akaho, and S. Amari, *Pathological spectra of the Fisher information metric and its variants in deep neural networks*, Neural Comput., 33 (2021), pp. 2274–2307.
[18] D. P. Kingma and P. Dhariwal, *Glow: Generative flow with invertible* $1x1$ *convolutions*, in Advances in Neural Information Processing Systems 31, Curran Associates, Red Hook, NY, 2018, pp. 10215–10224.

[19] D. P. Kingma and M. Welling, *Auto-encoding Variational Bayes*, in Proceedings of the 2nd International Conference on Learning Representations, ICLR, 2014.
[20] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al., *Overcoming catastrophic forgetting in neural networks*, Proc. Natl. Acad. Sci. USA, 114 (2017), pp. 3521–3526.
[21] A. Krizhevsky, V. Nair, and G. Hinton, *CIFAR-10 (Canadian Institute for Advanced Research)*, 2010, http://www.cs.toronto.edu/~kriz/cifar.html.
[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *ImageNet classification with deep convolutional neural networks*, in Advances in Neural Information Processing Systems 25, Curran Associates, Red Hook, NY, 2012, pp. 1097–1105.
[23] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proc. IEEE, 86 (1998), pp. 2278–2324.
[24] Y. Ma and Y. Fu, *Manifold Learning Theory and Applications*, CRC Press, Boca Raton, FL, 2012.
[25] J. Martens, *New insights and perspectives on the natural gradient method*, J. Mach. Learn. Res., 21 (2020), pp. 1–76.
[26] H. Narayanan and S. Mitter, *Sample complexity of testing the manifold hypothesis*, in Proceedings of the 23rd International Conference on Neural Information Processing Systems, Vol. 2, Curran Associates, 2010, pp. 1786–1794.
[27] F. Nielsen, *An Elementary Introduction to Information Geometry*, preprint, arXiv:1808.08271, 2018.
[28] C. Olah, A. Mordvintsev, and L. Schubert, *Feature visualization*, Distill, 2 (2017), e7, https://doi.org/10.23915/distill.00007.
[29] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, *PyTorch: An imperative style, high-performance deep learning library*, in Advances in Neural Information Processing Systems 32, Curran Associates, Red Hook, NY, 2019, pp. 8024–8035.
[30] W. Peebles, J. Peebles, J.-Y. Zhu, A. A. Efros, and A. Torralba, *The Hessian penalty: A weak prior for unsupervised disentanglement*, in Proceedings of the 16th European Conference on Computer Vision (ECCV), Springer, 2020.
[31] Y. Rubing, M. Jialin, and C. Pratik, *Does the Data Induce Capacity Control in Deep Learning?*, preprint, arXiv:2110.14163[cs.LG], 2021.
[32] S. Sommer and A. M. Bronstein, *Horizontal flows and manifold stochastics in geometric deep learning*, IEEE Trans. Pattern Anal. Mach. Intell., 44 (2020), pp. 811–822.
[33] A. Van den Oord, N. Kalchbrenner, L. Espeholt, K. Kavukcuoglu, O. Vinyals, and A. Graves, *Conditional image generation with PixelCNN decoders*, in Advances in Neural Information Processing Systems 29, Curran Associates, Red Hook, NY, 2016, pp. 4790–4798.
[34] Z. Zhang and H. Zha, *Principal manifolds and nonlinear dimensionality reduction via tangent space alignment*, SIAM J. Sci. Comput., 26 (2004), pp. 313–338.