Inversion dynamics of class manifolds in deep learning reveals tradeoffs underlying generalisation

Simone Ciceri[1], Lorenzo Cassani[1], Matteo Osella[3], Pietro Rotondo[2], Filippo Valle[3], and Marco Gherardi[1,2,*]

[1]Università degli Studi di Milano, via Celoria 16, 20133 Milan, Italy
[2]Istituto Nazionale di Fisica Nucleare — Sezione di Milano, via Celoria 16, 20133 Milan, Italy
[3]Università degli Studi di Torino and INFN, Sezione di Torino, via Giuria 1, 10125 Turin, Italy
[*]marco.gherardi@unimi.it

February 23, 2024

Abstract

To achieve near-zero training error in a classification problem, the layers of a feed-forward network have to disentangle the manifolds of data points with different labels, to facilitate the discrimination. However, excessive class separation can bring to overfitting since good generalisation requires learning invariant features, which involve some level of entanglement. We report on numerical experiments showing how the optimisation dynamics finds representations that balance these opposing tendencies with a non-monotonic trend. After a fast segregation phase, a slower rearrangement (conserved across data sets and architectures) increases the class entanglement. The training error at the inversion is stable under subsampling, and across network initialisations and optimisers, which characterises it as a property solely of the data structure and (very weakly) of the architecture. The inversion is the manifestation of tradeoffs elicited by well-defined and maximally stable elements of the training set, coined "stragglers", particularly influential for generalisation.

Introduction

Supervised deep learning excels in the baffling task of disentangling the training data, so as to reach near-zero training error, while still achieving good accuracy on the classification of unseen data. How this feat is achieved, particularly in relation to the geometry and structure of the training data, is currently a topic of debate and partly still an open question [1–12]. Activations of hidden layers in response to input examples, i.e., the internal representations of the data, evolve during training to facilitate eventual linear separation in the last layer. This requires a gradual segregation of points belonging to different classes, in what can be pictured as a disentangling motion between their class manifolds.

Segregation of class manifolds is a powerful conceptualisation that informs the design of distance-based losses in metric learning and contrastive learning [13–17] and underlies several approaches aimed at quantifying expressivity and generalisation, in artificial neural networks as well as in neuroscience [18–23]. Several recent efforts have leveraged this picture to characterise information processing along the layers of a deep network, particularly focusing on metrics such as intrinsic dimension and curvature [24–29]. In Ref. [25], for instance, two descriptors of manifold geometry, related to the intrinsic dimension and to the extension of the manifolds, are shown to undergo dramatic reduction as a result of training in deep convolutional neural networks. Such shrinking decisively supports the model's capacity in a memorisation task.

Yet, this appears to be just one side of the coin. There are indications that entanglement of class manifolds in the internal representations of deep neural networks promotes the correct discrimination of test data [30]. This fact appears counterintuitive, as more entangled representations should correspond to smaller margins. Still, manifold entanglement may encourage compression (in information-theoretic, rather than geometric, meaning) by reducing the number of discriminative features and by minimising the information about the input data that gets propagated through the network, effectively acting as a regularisation [31–33].

What emerges is a competition between learning invariant features and disentangling explanatory factors [34]. In this perspective, the classic bias-variance tradeoff, and the tension between train and test accuracy, translate to opposing tendencies for the optimisation dynamics: segregation of class manifolds on the one hand, and their entanglement on the other. How this tradeoff is realised dynamically through training is the focus of this manuscript.

In the spirit of statistical physics [35], we explore these questions in simple models, where patterns are more likely to emerge clearly, and exploration of their causes and consequences is less hampered by confounding factors. As an illustrative example, we consider a two-layer fully connected network, using $P = 8192$ points, $\{x^\mu\}$, from MNIST, a dataset widely employed in computer vision, containing 28×28 greyscale images of handwritten digits. We train the network to solve the parity classification task, where the label is $+1$ for even digits and -1 for odd ones. However, we anticipate that the phenomenology we will describe using this simple setting is more general: it is present also when training wider and deeper networks, as well as in more challenging data sets, such as KMNIST and CIFAR-10, and for different classification tasks (see Results and Supplementary Information).

At each epoch t during training, the activation of the hidden layer is a function h_t mapping elements of $\mathbb{R}^N$ to elements of $\mathbb{R}^H$, where N is the dimension of the input space and H is the width of the hidden layer. Our goal is to observe the evolution, throughout training, of the internal representations $h_t(x^\mu)$ of the training data $x^\mu \in \mathcal{T}$. In particular, we focus on the overall dispersion of points belonging to the same class y, i.e., of the images under h_t of equally-labelled elements of the training set. The projective nature of linear separability suggests to consider projections onto the unit sphere $\mathcal{S}^{n-1}$: $\hat{h}_t(x^\mu) = h_t(x^\mu)/\|h_t(x^\mu)\|$ (see the Methods for further explanation). Such normalisation is natural when h_t is the representation at the last layer, but we employ the same definition even when considering the first layer in a deep network. Thus, the internal representations of the two classes, or “class manifolds”, at each epoch t, are the two sets

$$\mathcal{M}_\pm(t) = \{\hat{h}_t(x^\mu) \mid y(x^\mu) = \pm 1\} \subset \mathbb{R}^H, \tag{1}$$

where $y(x^\mu)$ is the label of x^μ. Intuitively, separation of the two classes by the last layer is facilitated whenever $\mathcal{M}_+(T)$ and $\mathcal{M}_-(T)$, at the final epoch T, are small or far apart. This intuition is confirmed by analytical computations [19, 36].

Our analysis is based on a simple descriptor of manifold extension, the gyration radius, a metric proxy of the set's extension in Euclidean space. The two radii $R_\pm(t)$, together with the distance $D(t)$ between the two centres of mass of the two sets $\mathcal{M}_\pm(t)$, are three metric quantities recapitulating the geometry of the internal representations $\{h_t(x^\mu)\}$ (see Methods).

Results

Class manifold segregation in shallow networks

The internal representations of data points belonging to the same class are expected to move closer to one another after training. This is persuasively shown in [25], where the authors focus on state-of-the-art models (AlexNet and VGG-16) and on a sophisticated data set such as ImageNet. It is not obvious whether the systematic compaction that they observe is due specifically to special properties of the complex ImageNet data set, or to the heavily convolutional architectures probed. To address these questions, we trained a shallow network on the simple task described above, and compared $R_\pm$ and D before and after training. Figure 1(a) shows that the internal representations $\mathcal{M}_\pm$ in the trained model are always less entangled than at initialisation, i.e, they are more compact ($R_\pm$ is smaller) and further apart (D is larger).

Dynamics of class manifolds is non-monotonic

By contemplating the temporal dimension as well, one can address questions regarding the dynamics of manifold segregation. In particular, do the metric quantities evolve monotonically?

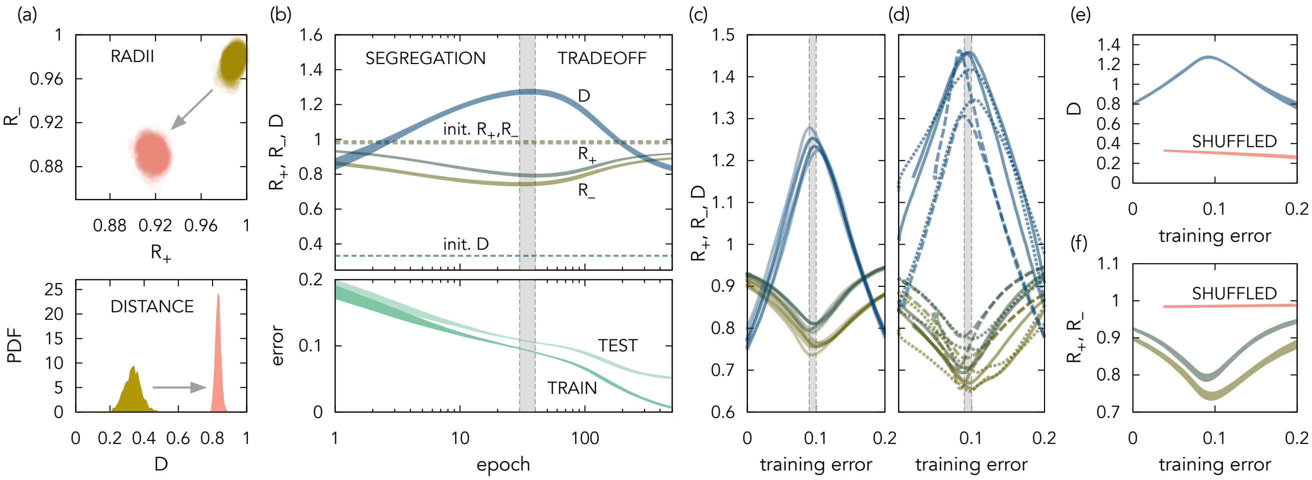

Figure 1: **Non-monotonic learning dynamics.** (a) Training disentangles the class manifolds. Scatter plot of the two radii $R_\pm$ (top) and histograms of the distance D (bottom) from 1000 independent runs, at initialization (yellow) and after training (pink). (b) Class manifold dynamics is non-monotonic. Radii and distance (top) and train and test errors (bottom) as functions of training epoch (on the x axis, in log scale); the dashed horizontal lines are the mean values at initialisation; inversion happens in the grey shaded regions; curve widths are 2 standard deviations. (c) Dynamics is robust to sub-sampling. The three metric quantities as functions of training error (only means shown, computed over 20 runs); different curves are obtained by training on non-overlapping subsets of MNIST. (d) Dynamics is similar across optimisers and hyperparameters. Solid lines: Adam (learning rates 0.001 and 0.005); dashed lines: GD with weight decay ($\lambda = 0.01$ and 0.05); dotted lines: GD with momentum ($\mu, \eta = 0.5, 0.5$ and $0.9, 0.2$). Curves are averages over 20 runs. (e,f) Randomised labels (pink curves) remove the non-monotonicity.

Figure 1(b) shows that the answer is negative: $R_\pm$ and D significantly overshoot before converging to their asymptotic values. An “inversion epoch” t_* marks the separation between two qualitatively different training periods. During the first, which happens fairly quickly, the internal representations of points belonging to the same class are brought closer to one another, while the representations of points belonging to different classes move further away from each other. After t_*, when the radii $R_\pm$ stop decreasing and the distance D stops increasing, training proceeds by a slow expansion of the manifolds and a gradual drift of their centres of mass, bringing them closer together. During the latter “expansion” phase, neither the radii nor the distance get back to their pre-training values.

Invariance of the training error at the inversion point

The location of the inversion epoch t_* where the time derivatives of the metric quantities change sign (i) is not appreciably different between R_+, R_-, and D (we obtain $t_* \approx 30$–40, corresponding to the grey area in Fig. 1); (ii) it barely fluctuates between runs started from different initialisations; (iii) it is not a special point for either the test or the training errors (Fig. 1(b)); see the Methods for definitions.

Since the inversion is an intrinsically dynamical phenomenon, an important question is how it depends on the optimisation dynamics. To address this question, we considered the metric quantities as functions of the training error $\epsilon_{\rm tr}$, by computing them from the sets $\mathcal{M}_\pm(t(\epsilon_{\rm tr}))$. The epoch $t(\epsilon_{\rm tr})$ here is defined as the first epoch when the training error crosses the value $\epsilon_{\rm tr}$. Figure 1(d) shows that, although t_* itself can be very different for different optimisers, the training error at the inversion epoch $\epsilon_{\rm tr}(t_*)$ is approximately invariant; the figure collects trajectories obtained by running Adam and gradient descent (GD) with different learning rates, with and without momentum and weight decay. Using stochastic GD with a batch size much smaller than the training set gives similar results, but shifts the inversion slightly towards smaller training errors (Supplementary Figure 1).

Similarly, we can ask whether the training error at the inversion is sensitive to sampling noise in the training data. Figure 1(c) shows that the dynamics, and $\epsilon_{\rm tr}(t_*)$ in particular, is quite independent of the specific subset of the training set employed for training.

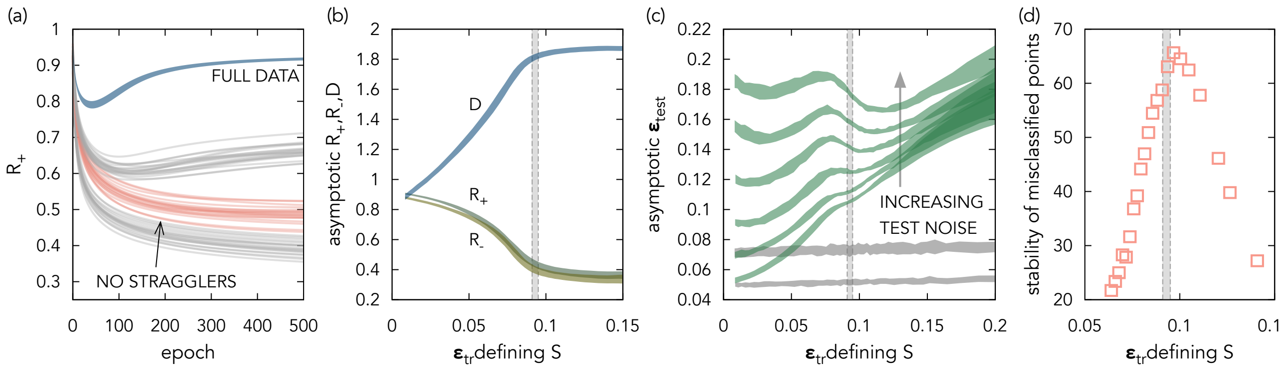

Figure 2: **Stragglers shape the dynamics and influence generalisation.** (a) Training without stragglers removes the inversion. The blue curve is obtained by training with the full dataset (shaded region corresponds to 2 sigmas); pink curves (indicated by the arrow) are 20 runs with the pruned training set $\mathcal{T} \setminus \mathcal{S}(t_*)$; the variability is due to the different initialisations, which affect both the dynamics and the elements of $\mathcal{S}(t)$; grey curves above and below the pink ones are obtained with pruned training sets $\mathcal{T} \setminus \mathcal{S}(t)$, with $t = 100 > t_*$ and $t = 10 < t_*$ respectively. (b) Metric quantities at convergence (y axis) using training sets $\mathcal{T} \setminus \mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$, as functions of ϵ_{tr} (x axis). (c) Removal of stragglers affects the test error at convergence (y axis). The green curves, from bottom to top, are obtained from noisy test sets, obtained by adding white noise, independently to each pixel, with standard deviation $\sigma = 0, 0.5, 0.75, 1., 1.2, 1.5$ respectively (inputs are standardised, see Methods); shaded regions correspond to 2 sigmas. Grey curves are obtained by removing, for each ϵ_{tr}, a random set of points, of the same cardinality as $\mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$ (only the two smallest values of σ are shown). (d) The inversion point marks a maximally stable set of misclassified points. Pink crosses are z-scores of the stability of the set $\mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$ (y axis; see Methods) under fluctuations in the initialisations, as a function of ϵ_{tr}. In all plots, $\mathcal{T}$ contains $P = 8192$ elements from MNIST, the architecture is a two-layer network with 20 hidden units.

Manifold expansion is elicited by structure in the data

What is causing the expansion phase? We will give an answer in the upcoming sections. A preparatory question is the following: does the inversion dynamics persist if one destroys the dependences between the data points and their labels? We repeated the same experiments as above, this time with randomly chosen labels for each input. The expansion phase disappears (Fig. 1(e,f)), giving way to a single slow segregation mode: the distance between the latent manifolds increases monotonically, while the two radii remain roughly constant throughout training. This result suggests that the non-monotonic dynamics is elicited by data structure, i.e., by the relation between the geometry of the data manifolds and the labels [8, 19, 37, 38].

Non-monotonic dynamics reveals trade-offs due to stragglers

What happens at the inversion? Some insight can be gained by watching which subset of the training set is still classified incorrectly at t_*. At t_*, the model classifies correctly most of the training set. Further optimisation of the loss function requires to trade off the overall segregation of this bulk for the separability of the few data points that are still misclassified.

Consider the set of misclassified points at epoch t:

$$\mathcal{S}(t) = \{x^\mu \in \mathcal{T} \mid \hat{y}_t(x^\mu) \neq y^\mu\}, \tag{2}$$

where $\hat{y}_t(x^\mu)$ is the label predicted by the network trained up to epoch t, Eq. (5). We name "stragglers" the elements of $\mathcal{S}(t_*)$, owing to their being late to catch up with the rest of the training set.

Does the expansion period persist if we remove the stragglers from the training set? Figure 2(a) shows how R_+ behaves when retraining the network on the reduced training set $\mathcal{T} \setminus \mathcal{S}(t_*)$. Removal of $\mathcal{S}(t_*)$ completely deleted the expansion period, in favour of a longer, and more seamless, segregation phase. Not only did the radius decrease monotonically on average: no inversion point could be identified in any single training run. Instead, removal of a random subset of the same cardinality did not affect the radii appreciably. Pruning the dataset by removing sets $\mathcal{S}(t < t_*)$, which are generally larger than $\mathcal{S}(t_*)$, had the same effect, but t_* was the largest epoch at which this happened: removing sets

$\mathcal{S}(t > t_*)$ did not destroy the non-monotonicity (bottom grey lines in Fig. 2(a)). Note that the sets $\mathcal{S}(t)$ depend on the initialisation; all statements made here were checked for 20 different initialisations.

The individuality of the inversion epoch t_* is emphasised by yet another experiment. We used the pruned dataset $\mathcal{T} \setminus \mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$, and measured the metric quantities at convergence, as functions of ϵ_{tr} (Fig. 2(b)). The training error at the inversion, $\epsilon_{\mathrm{tr}}(t_*)$, marks the boundary between two qualitatively different phases: when ϵ_{tr} is larger than $\epsilon_{\mathrm{tr}}(t_*)$ the asymptotic geometry of class manifolds is approximately independent of ϵ_{tr}.

In MNIST, with $P = 8192$ and a two-layer network with 20 hidden units, the number of stragglers is $|\mathcal{S}(t_*)| \approx 800$ (how this number changes for different tasks and architectures is reported below). Similarly to the inversion epoch t_*, the identity of the stragglers is conserved across network initialisations and, when training with stochastic GD, for different shuffles of the training set; we checked this by comparison with a null hypergeometric model (see Methods). Remarkably, among all the sets $\mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$, stragglers are maximally conserved (Fig. 2(d)).

Stragglers influence generalisation and noise robustness

The experiments above elucidated how stragglers shape the dynamics of the class manifolds, by triggering a tradeoff phase where entanglement between different classes, as measured via their metric properties, increases. As mentioned in the Introduction, entanglement between class manifolds is expected, in turn, to facilitate good generalisation, because it promotes learning of common invariant features. Persuasive clues that this is in fact the case were identified in Ref. [30]. Hence, is it possible to quantify the influence of stragglers on generalisation?

We trained a two-layer network on the pruned training sets $\mathcal{T} \setminus \mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$, and measured the test error at convergence, ϵ_{test}. The resulting ϵ_{test} as a function of ϵ_{tr}, from which the training set depends, is shown in Fig. 2(c). Testing accuracy deteriorates (ϵ_{test} increases) when removing any subset $\mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$ from $\mathcal{T}$. The magnitude of the deterioration is much larger than that obtained when removing a random subset of $\mathcal{T}$ of the same size as $\mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$ (grey regions in Fig. 2(c)).

The magnitude of the increase in ϵ_{test} is not a featureless function of ϵ_{tr}: the training error at the inversion epoch (grey vertical band in the figure) appears to separate two different branches of the curve. This contrast was accentuated when we repeated the same experiment with noisy versions of the test set, obtained by adding white noise to the input images with increasingly large variances. At low signal-to-noise ratios (large variances), the curves become non-monotonic, signalling a complex relation between the pruned subsets and the testing accuracy. Interestingly, at sufficiently low signal-to-noise ratios, removing stragglers reduces the test error to values below the original ones.

Given their impact on generalisation, it is natural to ask whether the role of stragglers is different on the two sides of the double-descent curve [39,40]. We repeated the preceding experiments with training set sizes and number of learnable parameters well above and well below the interpolation threshold, finding no appreciable differences (Supplementary Figure 2).

Non-monotonic dynamics and stragglers in other data sets

The non-monotonic segregation dynamics, as discussed above, is due to data structure. Is this a peculiarity of MNIST, or is the phenomenology more general? Figure 3(a) shows that KMNIST and fashion-MNIST, other commonly-employed data sets (see Methods), engender a similar non-monotonic dynamics. In addition, pruning the training set has similar consequences to those observed in MNIST. Stragglers, the misclassified examples at the inversion point, are again the most conserved among all subsets $\mathcal{S}(t)$ (Supplementary Figure 3).

The more complex data set CIFAR-10, which required a more expressive network (see the caption to Fig. 3), allows us to make an interesting observation. The goal of defining the inversion point as a function of training error (as opposed to the epoch) was to enable a fair comparison between optimisers. For simple data sets such as MNIST, plotting $R_{\pm}$ and D as functions of epoch or training error has no qualitative impact on the observed behavior. On the contrary, when an 8-layer architecture is trained on CIFAR-10, the dependence on the training error is much sharper (less fluctuating). Figures 3(c) and 3(d) show a comparison between the use of epochs and training error as independent variables.

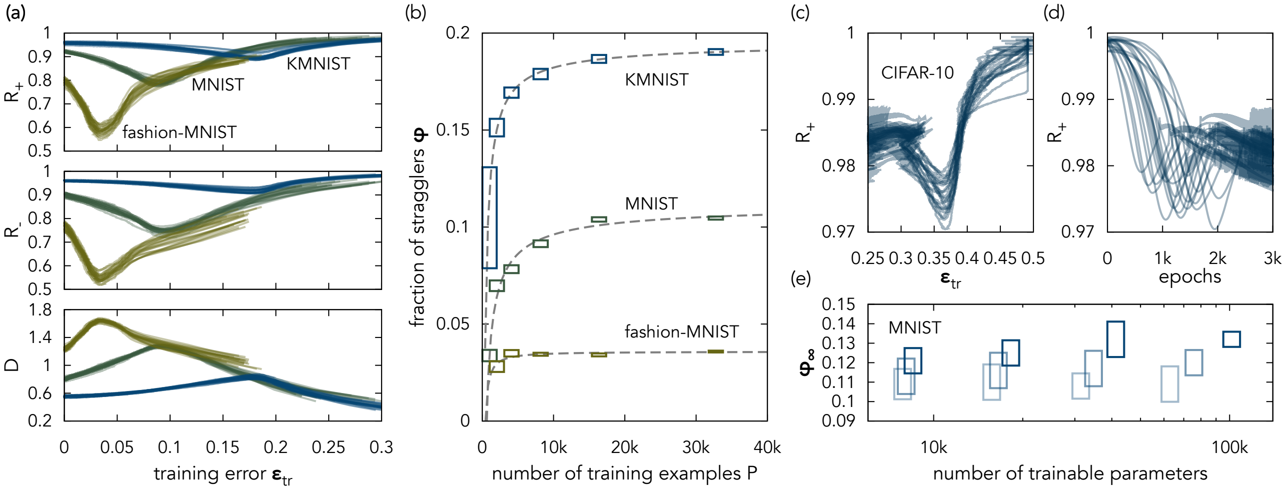

Figure 3: **Stragglers across data sets and architectures.** (a) The three metric quantities (y axes) as functions of the training error (x axis) for MNIST, KMNIST, and fashion MNIST. (b) Fraction of stragglers has a well-defined large-dataset limit. Dashed lines are fits of Eq. (12) to these data. (c), (d) Non-monotonic dynamics of R_+ (y axes) in CIFAR-10, as a function of training error in (c) and epochs in (d). (e) The asymptotic (large-dataset) fraction of stragglers (y axis) depends only weakly on the depth, and negligibly on the width, of the architecture. The four groups of boxes correspond to increasing widths from left to right; darker shades of blue correspond to deeper architectures. The curves in (a),(c), and (d) are 20 runs for each data set. Box heights in (b) and (e) correspond to 2 standard deviations. Architectures and parameters: 2 layers with 20 hidden units each in (a) and (b); 8 layers (fully connected) with 20 hidden units each, learning rate $\eta = 0.02$, in (c) and (d); 2,4, and 8 layers, each with 10,20,40, and 80 hidden units, $\eta = 0.1$, in (e).

Weak dependence on data size, depth, width, activation function

The training error at the inversion epoch $\phi = \epsilon_{\mathrm{tr}}(t_*)$ is the fraction of stragglers in the data set. Above, we have used a fixed number of training examples P. How does ϕ depend on this choice? Increasing P makes the training more and more difficult by adding new constraints in the optimisation problem, thus potentially also influencing the inversion point. However, this is not the case. Figure 3(b) shows that ϕ, as a function of P, saturates to a relatively small fraction for MNIST, KMNIST, and fashion MNIST. By fitting a tentative scaling form for ϕ as a function of P, one can attempt an extrapolation to infinite data set size, thus obtaining an estimate of the asymptotic fraction ϕ_∞ (see Methods, Eq. (12)). The fitted curves are in Fig. 3(b); we obtained $\phi_\infty \approx 4\%$ (fashion MNIST), 11% (MNIST), 20% (KMNIST).

The arrangement of these values and the higher inversion point for CIFAR-10 (Fig. 3(c)) indicate a relation between the complexity of the data set and the proportion of stragglers. A similar relation was reported between the fraction of critical samples, defined via the concept of adversarial examples, and data-set complexity [41].

We explored the dependence of the fraction of stragglers on the architecture by computing ϕ_∞ for fully connected networks with 2,4, and 8 layers, and 10,20,40, and 80 units per hidden layer. Figure 3(e) shows ϕ_∞ as a function of the total number of trainable parameters. In MNIST, about 11–13% of the training set is composed of stragglers, this figure being approximately constant over the range of depths and widths considered, encompassing more than an order of magnitude in total number of parameters. A weak systematic dependence emerges, mainly as a function of depth.

We checked the stability of the stragglers' identity across architectures, by comparing the sets $\mathcal{S}(t_*)$ obtained in models with different widths and depths. Stragglers are strongly conserved, with z-scores lying close to those obtained by comparing different training runs of a single shallow network (see Methods).

Finally, we observed that the inversion point, when using nonlinearities other than tanh, is slightly more fluctuating, but it still occurs around the same value of the training error. For a 4-layer network with 20 hidden units per layer, trained on 8192 examples from MNIST, we found $\phi = 0.089 \pm 0.009$ (reLU), $\phi = 0.088 \pm 0.007$ (leaky reLU with negative slope 0.1), and $\phi = 0.097 \pm 0.014$ (siLU), to be compared with $\phi = 0.098 \pm 0.002$ (tanh). The phenomenology persists in the fully linear case where the activation function is the identity, for which we found $\phi = 0.100 \pm 0.002$. This suggests that theoretical insight into the segregation dynamics of class manifolds may be gained by employing the theory of deep linear networks, which allows for analytical computations [42].

Specificity of stragglers within the data set

Do stragglers occupy special places with respect to the data manifold? While visual inspection does not reveal striking peculiarities, we found that stragglers, compared to other training examples, are significantly further away from the respective class centres. This analysis was done using all 10 MNIST classes, even though the classification problem is binary. By embedding the data in 2-dimensional space by t-SNE, it becomes visually clear that stragglers lie preferentially close to the class boundaries (see Supplementary Figure 4). This result suggests that it may be useful to think of stragglers as the "support vectors" for the non-linear classification problem.

Discussion

The nonmonotonic dynamics, and its inversion point in terms of training error, proved to be remarkably robust to changes in the hyperparameters and to perturbations. The fraction of stragglers appears to be an invariant property of the data set, characterising its complexity in terms of the tradeoffs discussed above. How this measure relates to other metrics of task difficulty, such as the intrinsic dimensions of the data set [43,44] or of the objective landscape [45], and to other specifics of data structure [41,46–48], is an open question.

In spite of the robustness presented above, we were able to find one way to disrupt the behaviour. Increasing the variance of the weight initialisation kept ϕ unchanged but pushed the radii towards 1 and made the minimum shallower. When the variance far exceeded the inverse of the number of units in hidden layers, the minimum disappeared abruptly and the radii became monotonic. This may be the manifestation of a transition between the feature learning and the lazy training regimes [49].

We list here some limitations of our work. (i) However robust, the phenomenology found in small fully-connected architectures should not be expected to arise immediately, or to be as clearcut, in state-of-the-art deep convolutional neural networks or transformers. (ii) We focussed solely on the internal representations at the first layer, even in deeper architectures. The dynamics in immediately downstream layers is not dissimilar, but representations closer to the output display different patterns. (iii) It is not evident how much the behaviour of the test error under pruning, Fig. 2(c), is sensitive to the choice of architecture. A more systematic exploration of these matters is left for future work. To address, at least partially, the limitations (i) and (iii), we have explored a set of 2-layers convolutional neural networks (CNN). Their behaviour varies with the choice of hyperparameters. Some architectures (e.g., with large kernel sizes) behave in the same way as fully connected ones, while in others the dynamics is qualitatively the same only for D (see the Supplementary Figure 5 for an example with 4×4 kernels, on MNIST). Even in the latter architecture, removal of stragglers (as identified with either a CNN or a FC network) has a profound impact on the training dynamics, making the inversion less marked. The results of Fig. 2(c), and 2(d) are also still valid (see Supplementary Figure 6).

Our empirical results shed light also on separate questions regarding the role of different examples during the training. Once architecture, optimiser, and training objective are fixed, thus establishing implicit inductive biases, the ability to generalise to unseen, possibly out-of-distribution, data is acquired by relying solely on the training set. Do all training data coherently cooperate in maximising train and test accuracy? Or does heterogeneity, a well documented feature of empirical data sets [50, 51], play a role? The inversion dynamics presented here indicates that two different compartments of $\mathcal{T}$ are involved in shaping distinct periods of training, and appear to have distinct contributions to generalisation. Stragglers emerge as a set of challenging instances located at the outskirts of class manifolds, which are memorised during later stages of training. We hypothesize that they carry information about the geometry of the data distribution, thereby contributing to the fine-grained properties of the learned discrimination boundaries. Omitting these examples from the training set results in more compact class representations but at the cost of decreased generalization. Conversely, when the geometric details of the data distribution are blurred by the presence of noise, as in our experiments with noisy test sets (Fig. 2c and Supplementary Fig. 2), removing the stragglers can instead enhance the out-of-distribution generalization.

Previous literature supports the observation that training examples are consistently classified at different learning stages, implying the existence of easy and hard examples [41]. Accurate metrics for assessing example difficulty are essential for designing data set pruning strategies [52] and curriculum learning protocols [53]. Exploring the potential role of stragglers as challenging examples in these contexts is an avenue for future research.

Methods

Models and training Most of our analysis was carried out on a shallow network with weights $w \in \mathbb{R}^{H\times N}$ and $v \in \mathbb{R}^{2\times H}$, and biases $b \in \mathbb{R}^2$ and $c \in \mathbb{R}^H$. We denote w_{ij}, v_i^a, b^a, and c_i the elements of these vectors, where $i = 1, \dots, H$; $j = 1, \dots, N$; $a = \pm 1$. The forward function is

$$f^a(x) = \sum_{i=1}^{H} v_i^a \left[h\left(x^\mu\right)\right]_i + b^a, \quad a = \pm 1, \tag{3}$$

where the vector $h(x^\mu)$ is the internal representation of x^μ; its components are

$$\left[h(x^\mu)\right]_i = \sigma\left(\sum_{j=1}^{N} w_{ij} x_j^\mu + c_i\right). \tag{4}$$

The transfer function σ was tanh for most of our analyisis. With these definitions, the predictor is

$$\hat{y}(x) = \underset{a}{\operatorname{argmax}}\left(\left\{f^a(x)\right\}_a\right). \tag{5}$$

We use the subscript t, as in h_t or $\hat{y}_t$, to specify that weights and biases are those evaluated at epoch t during training.

All models were trained using full-batch gradient descent with learning rate $\eta = 0.2$ (except where stated otherwise), with loss function

$$L = -\sum_{\mu=1}^{P} \log\left[\operatorname{softmax}\left\{f^a(x)\right\}_a\right]_{y(x^\mu)}, \tag{6}$$

where $y(x^\mu)$ is the label of x^μ in the training set. Weights and biases were initialised as independent random variables with the uniform distribution $\mathcal{U}\left(-1/\sqrt{n}, 1/\sqrt{n}\right)$, where n is the number of weights in the layer. Using other initialisation schemes, such as He or Xavier initialisation, does not change the results presented above, but see the comment regarding initialisation in the Discussion.

Datasets and standardisation We used the following data sets:

- MNIST, handwritten digits, 28x28 greyscale images [54],
- Kuzushiji-MNIST, or KMNIST; cursive Japanese characters, 28x28 greyscale images [55],
- Fashion MNIST; Zalando's article images, 28x28 greyscale images [56],
- CIFAR-10, 32x32 RGB images; the three channels were averaged down to greyscale [57].

All data sets are natively divided into training and test subsets. In all cases, unless specified otherwise, we constructed our training sets by using the first $P = 8192$ elements of the training subset. For computing the test errors, Eq. (7) below, we used the full test subsets. In both training and test sets, we binarised the classification task by using label $y = -1$ for odd classes and $y = 1$ for even classes, except when stated otherwise. All inputs in our training sets (respectively, test sets) were standardised by removing the mean and dividing by the standard deviation, separately for each pixel i: $x_i \rightsquigarrow (x_i - \langle x_i \rangle)/(\langle x_i^2 \rangle - \langle x_i \rangle^2)^{1/2}$, where the means $\langle x_i \rangle$ and $\langle x_i^2 \rangle$ are computed on the training set (respectively, test set).

Projection onto the unit sphere The problem of linear separation of a set of points is projective, in the following sense. Let us consider a linear separator identified by the vector $w \in \mathbb{R}^H$. A dichotomy f of the points $z^\mu \in \mathbb{R}^H$, $\mu = 1, \dots, P$, can be defined by setting $f(z^\mu) = \mathrm{sign}(w \cdot z^\mu)$ for all μ. There is a large class of transformations of the points z^μ under which the dichotomy is invariant. In particular, rescaling each point by a (possibly different) positive factor, $\tilde{z}^\mu = \lambda^\mu z^\mu$, $\lambda^\mu > 0$, gives $f(\tilde{z}^\mu) = f(z^\mu)$.

This fact shows that projection onto the unit sphere of the internal representations $h_t(x^\mu)$, as defined above Eq. (1), does not affect the final linear readout. We perform this normalisation step during evaluation of the metric quantities. However, we do it for both shallow and deep networks, even if the invariance does not hold for the latter. Without this transformation, the non-monotonicity in the learning dynamics would be less evident (see e.g. [18]).

Definitions of the quantities measured The train and test errors were computed as

$$\epsilon_{\mathrm{tr,test}} = 1 - \frac{1}{|\mathcal{T}_{\mathrm{tr,test}}|} \sum_{x \in \mathcal{T}_{\mathrm{tr,test}}} \delta_{\hat{y}(x), y(x)}, \tag{7}$$

where $\mathcal{T}_{\mathrm{tr,test}}$ is the training or test set respectively, and $y(x) = \pm 1$ is the label of x.

The squared gyration radii of the class manifolds, Eq. (1), are defined as follows:

$$R_\pm^2(t) = \frac{1}{2n_\pm^2} \sum_{x,y \in \mathcal{M}_\pm(t)} \|x - y\|^2, \tag{8}$$

where $n_+ = |\mathcal{M}_+(t)|$ is the number of elements with label +1 (and similarly for n_-). The distance $D(t)$ between the centres of mass of $\mathcal{M}_+(t)$ and $\mathcal{M}_-(t)$ is

$$D(t) = \left\| \frac{1}{n_+} \sum_{x \in \mathcal{M}_+(t)} x - \frac{1}{n_-} \sum_{x \in \mathcal{M}_-(t)} x \right\|. \tag{9}$$

The inversion epoch t_* is the epoch corresponding to the stationary value of each metric quantity:

$$\begin{aligned} t_*^{R_\pm} &= \underset{t}{\mathrm{argmin}}\; R_\pm(t), \\ t_*^{D} &= \underset{t}{\mathrm{argmax}}\; D(t) \end{aligned} \tag{10}$$

Operatively, we computed t_* separately for R_+, R_-, and D; then $\phi = \epsilon_{\mathrm{tr}}(t_*)$ was computed by averaging the training errors corresponding to these values of t_*. The values of ϕ reported are averages over 100 training runs.

Identity of stragglers To check that the identity of stragglers is conserved across initialisations, we performed the following experiment. We trained a two-layer neural network with 20 hidden units (on MNIST with $P = 8192$) starting from two random initialisations. For each of the two runs, $\alpha = 0, 1$, we identified the set $\mathcal{S}_\alpha(t_*)$ containing the misclassified elements of $\mathcal{T}$ at the inversion epoch; in addition, we picked two random subsets $\widehat{\mathcal{S}}_\alpha \subset \mathcal{T}$, such that $|\widehat{\mathcal{S}}_\alpha| = |\mathcal{S}_\alpha(t_*)|$. We computed the numbers of common points $M = |\mathcal{S}_0(t_*) \cap \mathcal{S}_1(t_*)|$ and $\hat{M} = |\hat{\mathcal{S}}_0 \cap \hat{\mathcal{S}}_1|$. The distributions of M and $\hat{M}$, over 10k repetitions, were peaked around $M \approx 680$ and $\hat{M} \approx 70$, with standard deviations $\sigma_M \approx 10$ and $\sigma_{\hat{M}} \approx 8$. Comparing these numbers to the number of stragglers for this setting (around 800) shows that around 85% of them are conserved, as opposed to 9% in the null model.

To quantify the stability of the stragglers, or more in general of the sets $\mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$, we used the z-score

$$z = \frac{\langle M \rangle - \langle \hat{M} \rangle}{\sigma_M}, \tag{11}$$

where $\langle M \rangle$ and $\langle \hat{M} \rangle$ are the averages of M and $\hat{M}$, and σ_M is the standard deviation of M, obtained with a similar experiment as the one described above, but for different epochs $t(\epsilon_{\mathrm{tr}})$ instead of t_*. The z-score for $\mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$, as a function of ϵ_{tr}, is plotted in Fig. 2d.

The z-score can be used to measure the conservation of the stragglers' identity between different models. To this aim, we used the same definition as above, and performed the two runs $\alpha = 0, 1$ on two possibly different architectures. We obtained the following z-scores; in parentheses, the depths L_α and widths H_α of the two architectures, $(L_0/H_0, L_1/H_1)$: $z = 50$ (2/20, 4/20), $z = 41$ (2/20, 8/20), $z = 40$ (4/20, 8/20). By comparison, the same-architecture z-scores for the deeper models are $z = 51$ (4/20, 4/20) and $z = 45$ (8/20, 8/20). The same method can be used to compare the identity of the stragglers for two different shuffles of the training set, when training is performed using stochastic gradient descent. For a shallow network with 20 hidden units, and batch size 32, we obtained $z \approx 11$.

Scaling with training set size A simple form, inspired by the theory of finite-size scaling in statistical physics [58], is effective in capturing the dependence of the fraction of stragglers, ϕ, on the size of the training set P:

$$\phi(P) = \phi_\infty \left[1 - \left(\frac{P}{P_0} \right)^{-\gamma} + o\big(P^{-\gamma}\big) \right]. \tag{12}$$

Fits are performed by varying ϕ_∞, P_0, and γ. The asymptotic values ϕ_∞ in Fig. 3(e) were obtained by fitting Eq. (12) to data with $P = 4096, 8192, 16384, 32768$.

Data availability

The datasets analysed during the current study are available in public repositories; links are in the corresponding publications [54–57].

Code availability

The code produced and used in the current study [59] is available on GitHub, under the GNU General Public License, version 3 (GPL-3.0), at `https://github.com/marco-gherardi/stragglers`

Acknowledgements

P.R .acknowledges funding from the Fellini program under the H2020-MSCA-COFUND action, Grant Agreement No. 754496, INFN (IT).

Author contributions statement

S.C. and M.G. discovered the stragglers. M.G., M.O., and P.R. conceived and designed the experiments. L.C., S.C., M.G., and F.V. performed the experiments. All authors analysed the results and wrote the paper. M.G. and M.O. supervised the analysis. M.G. coordinated the project.

References

[1] R. Pacelli, S. Ariosto, M. Pastore, F. Ginelli, M. Gherardi, and P. Rotondo. A statistical mechanics framework for Bayesian deep neural networks beyond the infinite-width limit. *Nature Machine Intelligence*, 5(12):1497–1507, 2023.

[2] Albert J. Wakhloo, Tamara J. Sussman, and SueYeon Chung. Linear classification of neural manifolds with correlated variability. *Phys. Rev. Lett.*, 131:027301, Jul 2023.

[3] Leonardo Petrini, Francesco Cagnetta, Umberto M. Tomasini, Alessandro Favero, and Matthieu Wyart. How deep neural networks learn compositional data: The random hierarchy model, 2023.

[4] Yu Feng, Wei Zhang, and Yuhai Tu. Activity–weight duality in feed-forward neural networks reveals two co-determinants for generalization. *Nature Machine Intelligence*, 5(8):908–918, 2023.

[5] Carlo Baldassi, Clarissa Lauditi, Enrico M. Malatesta, Rosalba Pacelli, Gabriele Perugini, and Riccardo Zecchina. Learning through atypical phase transitions in overparameterized neural networks. *Phys. Rev. E*, 106:014116, Jul 2022.

[6] Alessandro Ingrosso and Sebastian Goldt. Data-driven emergence of convolutional structure in neural networks. *Proceedings of the National Academy of Sciences*, 119(40):e2201854119, 2022.

[7] Madhu S. Advani, Andrew M. Saxe, and Haim Sompolinsky. High-dimensional dynamics of generalization error in neural networks. *Neural Networks*, 132:428–446, 2020.

[8] Sebastian Goldt, Marc Mézard, Florent Krzakala, and Lenka Zdeborová. Modeling the influence of data structure on learning in neural networks: The hidden manifold model. *Phys. Rev. X*, 10:041044, Dec 2020.

[9] Marc Mézard. Mean-field message-passing equations in the hopfield model and its generalizations. *Phys. Rev. E*, 95:022117, Feb 2017.

[10] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 5949–5958, Red Hook, NY, USA, 2017. Curran Associates Inc.

[11] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. 2016. cite arxiv:1611.03530Comment: Published in ICLR 2017.

[12] Charles H. Martin and Michael W. Mahoney. Rethinking generalization requires revisiting old ideas: statistical mechanics approaches and complex learning behavior. *arXiv:1710.09553 [cs.LG]*, 2017.

[13] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc., 2020.

[14] Konstantinos Kamnitsas, Daniel Castro, Loic Le Folgoc, Ian Walker, Ryutaro Tanno, Daniel Rueckert, Ben Glocker, Antonio Criminisi, and Aditya Nori. Semi-supervised learning via compact latent space clustering. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2459–2468. PMLR, 10–15 Jul 2018.

[15] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In Aasa Feragen, Marcello Pelillo, and Marco Loog, editors, *Similarity-Based Pattern Recognition*, pages 84–92, Cham, 2015. Springer International Publishing.

[16] Ruslan Salakhutdinov and Geoff Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 412–419, San Juan, Puerto Rico, 21–24 Mar 2007. PMLR.

[17] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, 2005.

[18] Achim Schilling, Andreas Maier, Richard Gerum, Claus Metzner, and Patrick Krauss. Quantifying the separability of data classes in neural networks. *Neural Networks*, 139:278–293, 2021.

[19] SueYeon Chung, Daniel D. Lee, and Haim Sompolinsky. Classification and geometry of general perceptual manifolds. *Phys. Rev. X*, 8:031003, 2018.

[20] Abigail A. Russo, Sean R. Bittner, Sean M. Perkins, Jeffrey S. Seely, Brian M. London, Antonio H. Lara, Andrew Miri, Najja J. Marshall, Adam Kohn, Thomas M. Jessell, Laurence F. Abbott, John P. Cunningham, and Mark M. Churchland. Motor cortex embeds muscle-like commands in an untangled population response. *Neuron*, 97(4), 2018.

[21] Jonathan Kadmon and Haim Sompolinsky. Optimal architectures in a solvable model of deep networks. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[22] Marino Pagan, Luke S. Urban, Margot P. Wohl, and Nicole C. Rust. Signals in inferotemporal and perirhinal cortex suggest an untangling of visual target information. *Nature Neuroscience*, 16:1132+, 2022/9/6/ 2013.

[23] James J. DiCarlo and David D. Cox. Untangling invariant object recognition. *Trends in Cognitive Sciences*, 11(8):333–341, 2007.

[24] Matthew Farrell, Stefano Recanatesi, Timothy Moore, Guillaume Lajoie, and Eric Shea-Brown. Gradient-based learning drives robust representations in recurrent neural networks by balancing compression and expansion. *Nature Machine Intelligence*, 4(6):564–573, 2022.

[25] Uri Cohen, SueYeon Chung, Daniel D. Lee, and Haim Sompolinsky. Separability and geometry of object manifolds in deep neural networks. *Nature Communications*, 11(1):746, 2020.

[26] 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*, 2019.

[27] Matthew Farrell, Stefano Recanatesi, Guillaume Lajoie, and Eric Shea-Brown. Recurrent neural networks learn robust representations by dynamically balancing compression and expansion. In *Real Neurons & Hidden Units: Future directions at the intersection of neuroscience and artificial intelligence @ NeurIPS 2019*, 2019.

[28] Stefano Recanatesi, Matthew Farrell, Madhu Advani, Timothy Moore, Guillaume Lajoie, and Eric Shea-Brown. Dimensionality compression and expansion in deep neural networks, 06 2019.

[29] Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[30] Nicholas Frosst, Nicolas Papernot, and Geoffrey Hinton. Analyzing and improving representations with the soft nearest neighbor loss. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2012–2020. PMLR, 09–15 Jun 2019.

[31] Alessandro Achille and Stefano Soatto. Where is the information in a deep neural network? *CoRR*, abs/1905.12213, 2019.

[32] Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–9, 2018.

[33] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *CoRR*, abs/1703.00810, 2017.

[34] Yoshua Bengio. Deep learning of representations: Looking forward. In Adrian-Horia Dediu, Carlos Martín-Vide, Ruslan Mitkov, and Bianca Truthe, editors, *Statistical Language and Speech Processing*, pages 1–37, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[35] Lenka Zdeborová. Understanding deep learning is also a job for physicists. *Nature Physics*, 16(6):602–604, May 2020.

[36] Marco Gherardi. Solvable model for the linear separability of structured data. *Entropy*, 23(3), 2021.

[37] Marc Mézard. Spin glass theory and its new challenge: structured disorder, 2023.

[38] Pietro Rotondo, Marco Cosentino Lagomarsino, and Marco Gherardi. Counting the learnable functions of geometrically structured data. *Phys. Rev. Res.*, 2:023169, May 2020.

[39] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

[40] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: where bigger models and more data hurt*. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, dec 2021.

[41] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International conference on machine learning*, pages 233–242. PMLR, 2017.

[42] Andrew M. Saxe, James L. Mcclelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural network. In *In International Conference on Learning Representations*, 2014.

[43] Vittorio Erba, Marco Gherardi, and Pietro Rotondo. Intrinsic dimension estimation for locally undersampled data. *Scientific Reports*, 9(1):17133, 2019.

[44] Elena Facco, Maria d'Errico, Alex Rodriguez, and Alessandro Laio. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific Reports*, 7(1):12140, 2017.

[45] Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. In *International Conference on Learning Representations*, 2018.

[46] Pietro Rotondo, Mauro Pastore, and Marco Gherardi. Beyond the storage capacity: Data-driven satisfiability transition. *Phys. Rev. Lett.*, 125:120601, Sep 2020.

[47] Mauro Pastore, Pietro Rotondo, Vittorio Erba, and Marco Gherardi. Statistical learning theory of structured data. *Phys. Rev. E*, 102:032119, 2020.

[48] Marco Gherardi and Pietro Rotondo. Measuring logic complexity can guide pattern discovery in empirical systems. *Complexity*, 21(S2):397–408, 2016.

[49] Mario Geiger, Stefano Spigler, Arthur Jacot, and Matthieu Wyart. Disentangling feature and lazy training in deep neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(11):113301, nov 2020.

[50] Andrea Mazzolini, Marco Gherardi, Michele Caselle, Marco Cosentino Lagomarsino, and Matteo Osella. Statistics of shared components in complex component systems. *Phys. Rev. X*, 8:021023, 2018.

[51] Andrea Mazzolini, Jacopo Grilli, Eleonora De Lazzari, Matteo Osella, Marco Cosentino Lagomarsino, and Marco Gherardi. Zipf and heaps laws from dependency structures in component systems. *Phys. Rev. E*, 98:012315, Jul 2018.

[52] Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536, 2022.

[53] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[54] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[55] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical japanese literature, 2018. cite arxiv:1812.01718Comment: To appear at Neural Information Processing Systems 2018 Workshop on Machine Learning for Creativity and Design.

[56] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[57] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009.

[58] J.L. Cardy. *Finite-size Scaling*. Current physics. North-Holland, 1988.

[59] Marco Gherardi. Inversion dynamics of class manifolds in deep learning reveals tradeoffs underlying generalisation. DOI: 10.5281/zenodo.8355859, 2023.