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

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

Simone Ciceri[1], Lorenzo Cassani[1], Pierre Pizzochero[1,2], Matteo Osella[3], Pietro Rotondo[2], 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, via Giuria 1, 10125 Turin, Italy
[*]marco.gherardi@unimi.it

March 10, 2023

Abstract

To achieve near-zero training error in a classification problem, the layers of a deep 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 remarkably 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 [1–6]. 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 [7–11] and underlies several approaches aimed at quantifying expressivity and generalisation, in artificial neural networks as well as in neuroscience [12–17]. 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 dimensionality and curvature [18–22]. In Ref. [19], 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 (together with inter-manifold correlations, which we neglect in this manuscript) 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, instead of hampering,

the correct discrimination of test data [23]. 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 [24–26].

What emerges is a competition between learning invariant features and disentangling explanatory factors [27]. 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, in light of its relation to generalisation, is the focus of this manuscript.

In the spirit of statistical physics [28], 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. Our analysis is carried out mainly with a two-layer fully connected network, using $P = 8192$ points, $\{x^\mu\}$, from MNIST, a dataset widely employed in computer vision. MNIST is not linearly separable directly, but high accuracy can be achieved already with shallow networks. We anticipate that the phenomenology emerging in this simple setting is carried on to wider and deeper networks, and to more challenging data sets (see Results and Discussion).

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 $\mathcal{S}^{n-1}$: $\hat{h}_t(x^\mu) = h_t(x^\mu)/\|h_t(x^\mu)\|$. Such normalisation is justified 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

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

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 results obtained for the perceptron [13, 29].

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 observables recapitulating the geometry of the internal representations $\{h_t(x^\mu)\}$ (see the Methods section for precise definitions).

Results

Class manifold segregation in shallow networks

As discussed above, the internal representations of data points belonging to the same class are expected to move closer to one another as a consequence of training. This is persuasively shown in [19], 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 (i) the multi-class nature of the object recognition tasks, (ii) special properties of the complex ImageNet data set, (iii) the heavily convolutional architecture of the neural networks, (iv) the specific choice of observables. To address these questions, we trained a shallow network on a simple task (Methods) and compared $R_\pm$ and D before and after training. Figure 1a 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 identifies two distinct training periods

This experiment confirms that the segregation of class manifolds is not a special feature of deep convolutional networks, but rather a general mechanism, operating in simple settings as well. Only the

initial and final snapshots of the model are observed in this analysis. By contemplating the temporal dimension as well, one can address questions regarding the dynamics of manifold segregation. In particular, do the metric observables evolve monotonically?

Figure 1b 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 period, 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; their asymptotic values at convergence are, respectively, $R_\pm(T) < R_\pm(0)$ and $D(T) > D(0)$, as stated above.

Invariance of the training error at the inversion point

The location of the inversion epoch t_* where the time derivatives of the metric observables change sign (i) is not appreciably different between R_+, R_-, and D (we obtain $t_* \approx 12$–15, corresponding to the shaded 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. 1b).

Since the inversion is an intrinsically dynamical phenomenon, an important question is how its location depends on the optimisation dynamics. In order to address this question we considered the metric observables as functions of the training error $\epsilon_{\rm tr}$, by computing them from the sets $\mathcal{M}_\pm(t(\epsilon_{\rm tr}))$. This procedure yields approximately single-valued functions $R_\pm(\epsilon_{\rm tr})$ and $D(\epsilon_{\rm tr})$, as the average training error is monotonically decreasing in time. Figure 1d 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 SGD with different learning rates, with and without momentum and weight decay.

Similarly, we can ask whether the training error at inversion is sensitive to sampling noise in the training data. Figure 1c shows that the dynamics, and the inversion point in particular, is quite independent of the specific subset of MNIST employed for training.

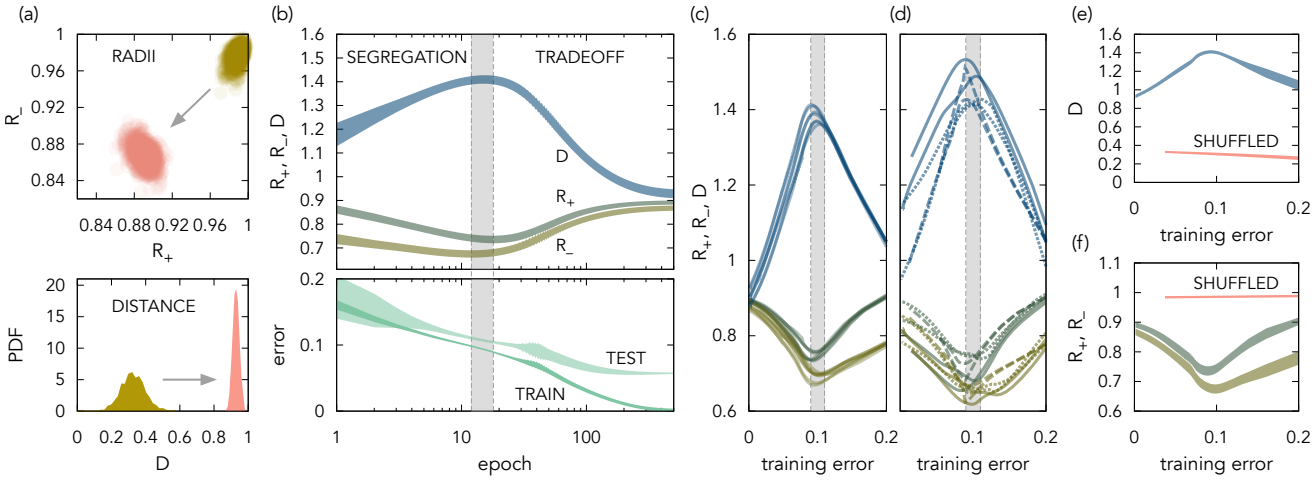

Figure 1: (a) Training disentangles class manifolds. Density plot of the two radii $R_\pm$ (top) and histograms of the distance D (bottom) from 1000 independent samples, at initialization (yellow) and after training (pink). (b) Class manifold dynamics is non-monotonic. Radii and distance (top) and train and test errors (bottom) as functions of training epoch (on x axis, in log scale); inversion happens in the grey shaded regions; curve widths are 2 standard deviations. (c) Dynamics is robust to sub-sampling. The three metric observables as functions of training error (only means shown); 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: SGD with weight decay ($\lambda = 0.01$ and 0.05); dotted lines: SGD with momentum ($\mu, \eta = 0.5, 0.5$ and $0.9, 0.2$). (e,f) Randomised labels (pink curves) remove the non-monotonicity.

Manifold expansion is elicited by structure in the data

What is causing the expansion phase? Is it due to shape of the data manifold (the way data points are arranged in input space) or is it related to the dependence between each data point and its label within the data set? We repeated the same experiments as above, this time with randomly chosen labels for each input. The expansion phase disappears (Fig. 1e), 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.

Inversion dynamics reveals tradeoffs due to straggling data points

What happens at the inversion? Some insight can be gained by watching which subset of the training data is still classified incorrectly at t_*. Just after the segregation period, the model classifies correctly most of the training data. After this point, further optimization 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 all 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. (6). Does the expansion period persist if we remove the elements of $\mathcal{S}(t_*)$ from the training set? Figure 2(a) shows how the metric observable R_+ behaves when retraining the network on the reduced training set $\mathcal{T} \setminus \mathcal{S}(t_*)$ (R_- and D are similar). Removal of $\mathcal{S}(t_*)$ completely deleted the expansion period, in favour of a longer, and more seamless, segregation phase. (Instead, removal of a random subset of the same cardinality did not affect the radii appreciably.) Not only did the radius decrease monotonically on average: no inversion point could be identified in any single training run.

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 2a).

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

We name "stragglers" the elements of $\mathcal{S}(t_*)$, owing to their being late to catch up with the rest of the training set. In MNIST, with $P = 8192$ and a two-layer network, their number is $|\mathcal{S}(t_*)| \approx 800$ (how this number changes for different tasks and architectures is explored below). Similarly to the inversion epoch t_*, the identity of the stragglers is conserved across network initialisations; we checked this by comparison with a null hypergeometric model (see the Methods). Remarkably, among all the sets $\mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$, stragglers are maximally conserved (Fig. 2d).

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. Entanglement, in turn, is expected to have an impact on generalisation, since, as discussed in the Introduction, the similarity structure of internal representations was found to correlate with testing accuracy. Is it possible to quantify the influence of stragglers on generalisation?

To answer this question, 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 dataset depends, is shown in Fig. 2c. 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 $\mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$. Perhaps unexpectedly, the magnitude of the increase in ϵ_{test} is not a smooth, 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 function. 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

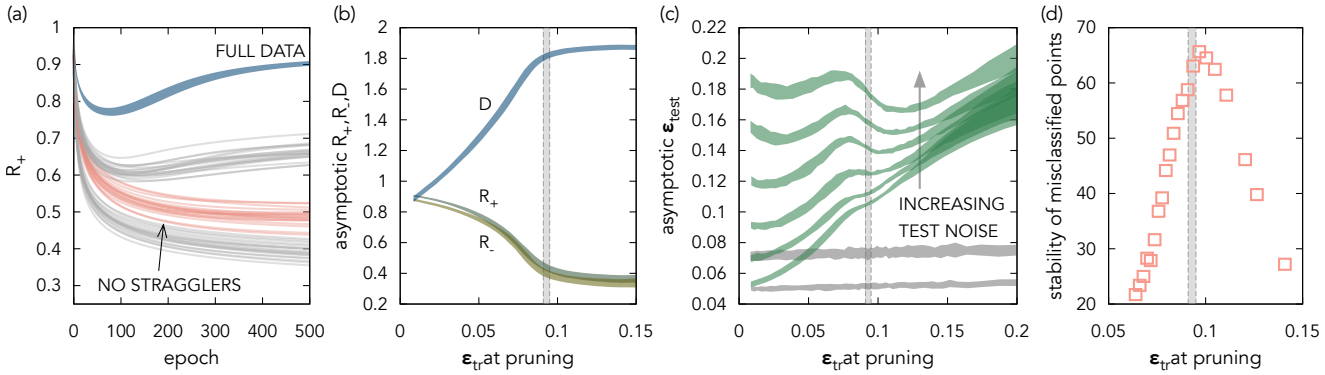

Figure 2: (a) Training without stragglers removes the inversion of R_+ (y axis) (similar behaviour occurs for R_- and D, not shown). The blue curve is obtained by training with the full dataset (shaded region corresponds to 2 sigmas); pink curves (indicated by the arrow) are 20 runs with the pruned dataset $\mathcal{T} \setminus \mathcal{S}(t_*)$; grey curves above and below the pink ones are obtained with pruned datasets $\mathcal{T} \setminus \mathcal{S}(t)$, with $t > t_*$ and $t < t_*$ respectively. (b) Metric observables at convergence (y axis) using training set $\mathcal{T} \setminus \mathcal{S}\left(t\left(\epsilon_{\rm tr}\right)\right)$, as functions of $\epsilon_{\rm tr}$ (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 $\epsilon_{\rm tr}$, a random set of points, of the same cardinality as $\mathcal{S}\left(t\left(\epsilon_{\rm tr}\right)\right)$ (only the two smallest values of σ are shown). (d) The inversion point marks a maximally stable set of misclassified points. Pink crosses are a measure of the stability of the set $\mathcal{S}\left(t\left(\epsilon_{\rm tr}\right)\right)$ (y axis; see Methods) under fluctuations in the initialisations, as a function of $\epsilon_{\rm tr}$. In all plots, $\mathcal{T}$ contains $P = 8192$ elements from MNIST, the architecture is a two-layer network with 20 hidden units.

between the pruned subsets and the testing accuracy. At sufficiently low signal-to-noise ratios, removing stragglers even reduces the test error with respect to the values for $\epsilon_{\rm tr} \to 0$. The interpretation of these results is not straightforward, and we propose an attempt in the Discussion.

Inversion dynamics in other data sets

The nonmonotonic segregation dynamics, as discussed above, is due to the specific dependence of the labels from the input images. Is this a peculiarity of MNIST, or is the phenomenology more general? Figure 3a shows that other commonly-employed data sets engender a similar inversion dynamics of the metric observables (see the Methods). While KMNIST and fashion MNIST are simple enough that a two-layer network can rapidly reach the training error where the inversion is observed, CIFAR-10 is more challenging and required a more expressive network (see the caption).

The more complex data set CIFAR-10 allows 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. Indeed, for simple data sets such as MNIST, plotting $R_\pm$ and D as a function of epoch or training error bears no qualitative impact on the observed behavior. On the contrary, when an 8-layer architecture is trained on CIFAR-10, the dependence of the geometric observables on the training error is much sharper (less fluctuating). Figures 3c and 3d show a comparison between the use of epochs and training error as independent variables.

Stragglers depend weakly on network depth and width

The training error at the inversion epoch $\phi = \epsilon_{\rm tr}(t_*)$ is the fraction of stragglers in the data set. Above, we have used a fixed number of training examples P. How does the fraction 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. Strikingly, this is not the case. Figure 3b shows ϕ as a function of P for MNIST, KMNIST, and fashion MNIST. For all three data sets, ϕ saturates to a relatively small fraction. By fitting a tentative scaling form for the fraction of stragglers as a function of P, one can attempt an extrapolation to infinite data set size. A simple form, inspired

by the theory of finite-size scaling in statistical physics [30], is effective:

$$\phi(P) = \phi_\infty \left[1 - \left(\frac{P}{P_0} \right)^{-\gamma} + o\big(P^{-\gamma}\big) \right]. \tag{3}$$

Fits are performed by varying ϕ_∞, P_0, and γ. The fitted curves are in Fig. 3; we obtain $\phi_\infty \approx 11\%$ (MNIST), 20% (KMNIST), 4% (fashion MNIST).

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. The asymptotic value ϕ_∞ was obtained by fitting Eq. (3) to data with $P = 4096, 8192, 16384, 32768$. Figure 3e 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.

Finally, 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).

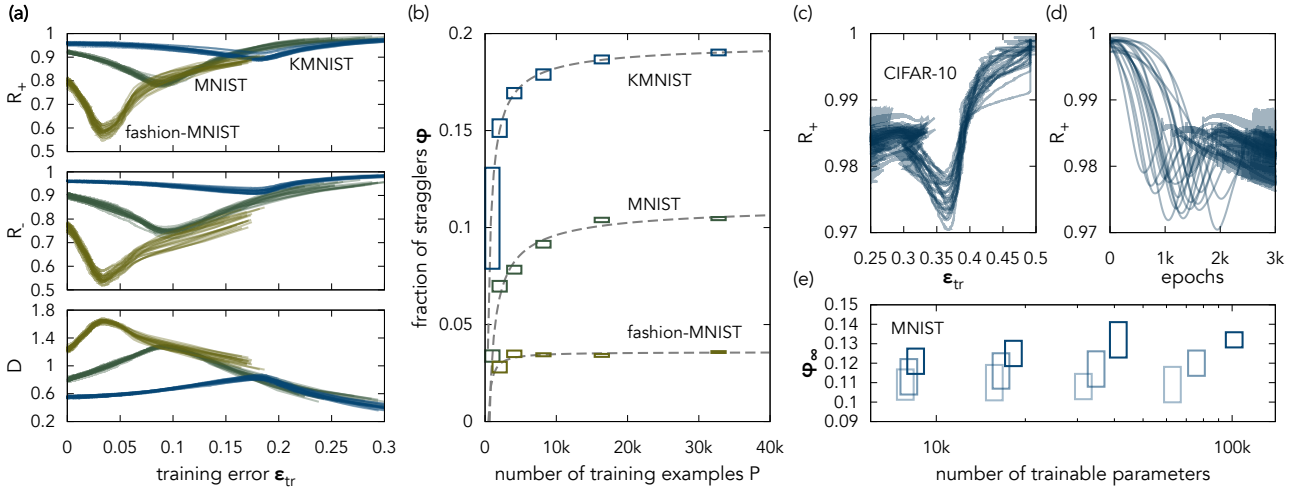

Figure 3: (a) Inversion dynamics is common across data sets. The three metric observables (y axes) as functions of training error (x axis) for MNIST, KMNIST, and fashion MNIST. (b) Fraction of stragglers has a well-defined large-dataset limit. Dashed lines are fits of Eq. (3) to these data. (c), (d) Inversion 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).

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 [31, 32] or of the objective landscape [33], and to other specifics of data structure [34, 35], is an interesting open question.

One aspect that was not mentioned above is the dependence on the activation function. We observed that the inversion point, when using nonlinearities other than tanh, is slightly more fluctuating, but 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). Interestingly, the phenomenology remains the same also 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 [36].

In spite of the robustness presented above, we were able to find one way to disrupt the behaviour. Increasing the variance of the weight initialisations 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 [37].

The response of the test accuracy to removal of the sets $\mathcal{S}\left(t\left(\epsilon_{\mathrm{tr}}\right)\right)$ is heterogeneous. When interpreting Fig. 2c, some insight can be obtained by considering that the training error on the x axis is exactly the fraction of points removed from the training set. Let us suppose that the pruned sets are ordered, such that $\mathcal{S}\left(t\left(\epsilon_0\right)\right) \subset \mathcal{S}\left(t\left(\epsilon_1\right)\right)$ whenever $\epsilon_0 < \epsilon_1$; this is not the case in general, but it is a good approximation, as the number of points leaving $\mathcal{S}(t)$ between epochs t and $t+1$ is generally much less than the number of points entering it. Then, the curves in Fig. 2c can be loosely interpreted as expressing the effect of the cumulative removal of points from the training set. This suggests to interpret the slope of the curves at ϵ_{tr} as the impact, on the test error, brought about by those points that enter the set $\mathcal{S}\left(t\left(\epsilon_0\right)\right)$ exactly at epoch t. For finite and infinite signal-to-noise ratios, the largest and the smallest slopes both occur just on the left of the inversion. This may be condensed in a qualitative statement: some stragglers are exceptionally beneficial for generalisation while others are exceptionally detrimental.

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. (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. 2c, is sensitive to the choice of architecture and training set. A more systematic exploration of these matters is left for future work.

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 (which is, in other contexts, a well documented feature of empirical data sets [38]) play a role? The inversion dynamics indicates that two different compartments of $\mathcal{T}$ are involved in shaping distinct periods of training, and appear to have distinct contributions to generalisation. In this regard, the identification of stragglers may lend itself to applications as a dataset-distillation aid, for the design of dataset-specific curriculum learning protocols [39], and it can inspire, in the field of explainable AI, post-hoc methods for the selection of exceptionally influential instances within the data set [40].

Methods

Models and training Most of our analysis was carried out on a shallow network with weights $w \in \mathbb{R}^{N \times H}$, $v \in \mathbb{R}^{2 \times H}$, and biases b^a, $a = \pm 1$, and c_i, $i = 1, \ldots, H$. The forward function is

$$f^a(x) = \sum_{i=1}^{H} v_i^a \left[h\left(x^\mu\right)\right]_i + b^a, \quad a = \pm 1, \tag{4}$$

where the vector $h(x^\mu)$ is the internal representation, whose components are

$$\left[h(x^\mu)\right]_i = \sigma\left(\sum_{j=1}^{N} w_{ij} x_j^\mu + c_i\right). \tag{5}$$

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

$$\hat{y}(x) = \underset{a}{\operatorname{argmax}} \left(\{ f^a(x) \}_a \right). \tag{6}$$

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

$$L = -\sum_{\mu=1}^{P} \log \left(\operatorname{softmax} \{ f^a(x) \}_a \right)_{y(x^\mu)}. \tag{7}$$

Weights and biases were initialised as independent random variables with the uniform distribution $\mathcal{U}(-1/\sqrt{n}, 1/\sqrt{n})$, 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 have used the following data sets:

- MNIST, handwritten digits, 28x28 greyscale images [41]
- Kuzushiji-MNIST, or KMNIST; cursive Japanese characters, 28x28 greyscale images [42],
- fashion MNIST; Zalando's article images, 28x28 greyscale images [43],
- CIFAR-10, 32x32 RGB images; the three channels were averaged down to grayscale [44].

In all cases, unless specified otherwise, we constructed our training sets by using the first $P = 8192$ elements of the full data set, and we binarised the classification task by using label $y = -1$ for odd classes and $y = 1$ for even classes. All inputs 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.

Observables The train and test errors were computed as

$$\epsilon_{\text{tr,test}} = \frac{1}{|\mathcal{T}_{\text{tr,test}}|} \sum_{x \in \mathcal{T}_{\text{tr,test}}} \delta_{\hat{y}(x), y(x)}, \tag{8}$$

where $\mathcal{T}_{\text{tr,test}}$ 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:

$$R_\pm^2(t) = \frac{1}{2n_\pm^2} \sum_{x,y \in \mathcal{M}_\pm(t)} \|x - y\|^2, \tag{9}$$

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

$$D(t) = \left\| \frac{1}{n_+} \sum_{x \in \mathcal{M}_+(t)} x - \frac{1}{n_-} \sum_{x \in \mathcal{M}_-(t)} x \right\|. \tag{10}$$

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

$$\begin{aligned} t_*(R_\pm) &= \underset{t}{\operatorname{argmin}}\, R_\pm(t), \\ t_*(D) &= \underset{t}{\operatorname{argmax}}\, D(t) \end{aligned} \tag{11}$$

Operatively, we computed t_* separately for R_+, R_-, and D; then $\phi = \epsilon_{\text{tr}}(t_*)$ 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 $\mathcal{S}_\alpha(t_*)$ 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 $\mathcal{S}\left(t\left(\epsilon_{\rm tr}\right)\right)$, we used the z-score

$$z = \frac{\langle M \rangle - \langle \hat{M} \rangle}{\sigma_M}, \tag{12}$$

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 the experiment described above. The z-score for $\mathcal{S}\left(t\left(\epsilon_{\rm tr}\right)\right)$, as a function of $\epsilon_{\rm tr}$, 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).

Data availability

The datasets analysed during the current study are available in public repositories; links are in the corresponding publications [41–44].

Code availability

The code produced and used in the current study is available on Google Colaboratory under the GNU General Public License, version 3 (GPL-3.0), at the following URL:
`https://colab.research.google.com/drive/1AmU2wzZuPpgJHDyfyMqSuPDlWLCb8ZB3?usp=sharing`.

References

[1] 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.

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

[3] 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.

[4] 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.

[5] 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.

[6] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. ICLR 2017.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

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

[14] 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.

[15] 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.

[16] 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, 2013.

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

[18] 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.

[19] 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.

[20] 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.

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

[22] 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.

[23] 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, 2019.

[24] Alessandro Achille, Giovanni Paolini, Stefano Soatto. Where is the information in a deep neural network? *arXiv:1905.12213 [cs.LG]*, 2019.

[25] 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.

[26] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv:1703.00810 [cs.LG]*, 2017.

[27] 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.

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

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

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

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

[32] 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.

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

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

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

[36] 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.

[37] 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.

[38] 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.

[39] 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.

[40] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable AI: A review of machine learning interpretability methods. *Entropy*, 23(1), 2021.

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

[42] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep learning for classical Japanese literature. *arXiv:1812.01718 [cs.CV]*, 2018.

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

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