

High-dimensional dynamics of generalization error in neural networks

Madhu S. Advani^{a,1}, Andrew M. Saxe^{a,2,*}, Haim Sompolinsky^{a,b}

^a Center for Brain Science, Harvard University, Cambridge, MA 02138, United States of America

^b Edmond and Lily Safra Center for Brain Sciences, Hebrew University, Jerusalem 91904, Israel

ARTICLE INFO

Article history:

Received 4 January 2020

Received in revised form 18 August 2020

Accepted 24 August 2020

Available online 5 September 2020

Keywords:

Neural networks

Generalization error

Random matrix theory

ABSTRACT

We perform an analysis of the average generalization dynamics of large neural networks trained using gradient descent. We study the practically-relevant “high-dimensional” regime where the number of free parameters in the network is on the order of or even larger than the number of examples in the dataset. Using random matrix theory and exact solutions in linear models, we derive the generalization error and training error dynamics of learning and analyze how they depend on the dimensionality of data and signal to noise ratio of the learning problem. We find that the dynamics of gradient descent learning naturally protect against overtraining and overfitting in large networks. Overtraining is worst at intermediate network sizes, when the effective number of free parameters equals the number of samples, and thus can be reduced by making a network smaller or *larger*. Additionally, in the high-dimensional regime, low generalization error requires starting with small initial weights. We then turn to non-linear neural networks, and show that making networks very large does not harm their generalization performance. On the contrary, it can in fact reduce overtraining, even without early stopping or regularization of any sort. We identify two novel phenomena underlying this behavior in overcomplete models: first, there is a frozen subspace of the weights in which no learning occurs under gradient descent; and second, the statistical properties of the high-dimensional regime yield better-conditioned input correlations which protect against overtraining. We demonstrate that standard application of theories such as Rademacher complexity are inaccurate in predicting the generalization performance of deep neural networks, and derive an alternative bound which incorporates the frozen subspace and conditioning effects and qualitatively matches the behavior observed in simulation.

© 2020 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Deep learning approaches have attained high performance in a variety of tasks (LeCun, Bengio, & Hinton, 2015; Schmidhuber, 2015), yet their learning behavior remains opaque. Strikingly, even very large networks can attain good generalization performance from a relatively limited number of examples (Canziani, Paszke, & Culurciello, 2017; He, Zhang, Ren, & Sun, 2016; Simonyan & Zisserman, 2015). For instance, the VGG deep object recognition network contains 155 million parameters, yet generalizes well when trained with the 1.2 million examples of ImageNet (Simonyan & Zisserman, 2015). This observation leads to important questions. What theoretical principles explain this good performance in this limited-data regime? Given a fixed dataset size, how complex should a model be for optimal generalization error?

In this paper we study the average generalization error dynamics of various simple training scenarios in the “high-dimensional” regime where the number of samples P and parameters N are both large ($P, N \rightarrow \infty$), but their ratio $\alpha = P/N$ is finite (e.g. Advani, Lahiri, & Ganguli, 2013). We start with simple models, where analytical results are possible, and progressively add complexity to verify that our findings approximately hold in more realistic settings. We mainly consider a student–teacher scenario, where a “teacher” neural network generates possibly noisy samples for a “student” network to learn from Saad and Solla (1995), Seung, Sompolinsky, and Tishby (1992) and Watkin, Rau, and Biehl (1993). This simple setting retains many of the trade-offs seen in more complicated problems: under what conditions will the student overfit to the specific samples generated by the teacher network? How should the complexity (e.g. number of hidden units) of the student network relate to the complexity of the teacher network?

First, in Section 2, we investigate linear neural networks. For shallow networks, it is possible to write exact solutions for the dynamics of batch gradient descent learning as a function of the amount of data and signal to noise ratio by integrating random matrix theory results and the dynamics of learning. This

* Corresponding author.

E-mail address: andrew.saxe@psy.ox.ac.uk (A.M. Saxe).

¹ Both authors contributed equally to this work.

² Current address: Department of Experimental Psychology, University of Oxford, Oxford OX2 6GG, United Kingdom.

is related to a large body of prior work on shallow networks (Bai & Silverstein, 2010; Baldi & Chauvin, 1991; Baldi & Hornik, 1995; Benaych-Georges & Rao, 2011, 2012; Chauvin, 1990; Dodier, 1995; Dunmur & Wallace, 1999; Hoyle & Rattray, 2007; Kinouchi & Caticha, 1995; Krogh & Hertz, 1992; LeCun, Kanter, & Solla, 1991; Seung et al., 1992). Our results reveal that the combination of early stopping and initializing with small-norm weights successfully combats overtraining, and that overtraining is worst when the number of samples equals the number of model parameters.

For deep linear neural networks, in Section 3 we derive a reduction of the full coupled, nonlinear gradient descent dynamics to a much smaller set of parameters coupled only through a global scalar. Our reduction is applicable to any depth network, and yields insight into the dynamical impact of depth on training time. The derivation here differs from previous analytic works on the dynamics of deep linear networks (Saxe, McClelland, & Ganguli, 2014) in that we do not assume simultaneous diagonalizability of the input and input–output correlations. It shows that generalization error in this simple deep linear case behaves qualitatively similarly to the shallow linear case.

Next, in Section 4 we turn to nonlinear networks. We consider a nonlinear student network which learns from a dataset generated by a nonlinear teacher network. We show through simulation that the qualitative intuitions gained from the linear analysis transfer well to the nonlinear setting. Remarkably, we find that catastrophic overtraining is a symptom of a model whose complexity is exactly matched to the size of the training set, and can be combated either by making the model smaller or *larger*. Moreover, we find no evidence of overfitting: the optimal early-stopping generalization error decreases as the student grows larger, even when the student network is much larger than the teacher and contains many more hidden neurons than the number of training samples. Our findings agree both with early numerical studies of neural networks (Caruana, Lawrence, & Giles, 2001) and the good generalization performance of massive neural networks in recent years (e.g. Belkin, Hsu, Ma and Mandal, 2019; Simonyan & Zisserman, 2015; Spigler et al., 2018). In this setting, we find that bigger networks are better, and the only cost is the computational expense of the larger model.

We also analyze a two layer network to gain theoretical intuition for these numerical results. A student network with a wide hidden layer can be highly expressive (Barron, 1993), even if we choose the first layer to be random and fixed during training. However, in this setting, when the number of hidden units N_h is larger than the number of samples P , our linear solution implies that there will be no learning in the $N_h - P$ zero-eigenvalue directions of the hidden layer covariance matrix. It follows that further increasing the network size will not actually increase the complexity of the functions the network can learn. Furthermore, there seems to be an implicit reduction in model complexity and overfitting at late stopping times when N_h is increased above P because the non-zero eigenvalues are actually pushed away from the origin, thus reducing the maximum norm learned by the student weights. Finally, through simulations with the MNIST dataset, we show that these qualitative phenomena are recapitulated on real-world datasets.

The excellent performance of complex models may seem to contradict straightforward applications of VC dimension and Rademacher bounds on generalization performance (Zhang, Bengio, Hardt, Recht, & Vinyals, 2017). We take up these issues in Sections 6 and 7. Our results show that the effective complexity of the model is regularized by the combined strategy of early stopping and initialization with small norm weights. This two-part strategy limits the norm of the weights in the network, thereby limiting the Rademacher complexity. This strategy is analogous to

that used in support vector machines to allow generalization from kernels with infinite VC dimension: generalization is preserved provided the training process finds a large-margin (low-norm weight vector) solution on the training data. Hence, while a large deep network can indeed fit random labels, gradient-trained DNNs initialized with small-norm weights learn simpler functions first and hence generalize well if there is a consistent rule to be learned.

Remarkably, even without early stopping, generalization can still be better in larger networks (Belkin, Hsu, Ma et al., 2019). To understand this, we derive an alternative bound on the Rademacher complexity of a two layer non-linear neural network with fixed first layer weights that incorporates the dynamics of gradient descent. We show that complexity is limited because of a frozen subspace in which no learning occurs, and overtraining is prevented by a larger gap in the eigenspectrum of the data in the hidden layer in overcomplete models. Our bound provides new intuitions for generalization behavior in large networks, and qualitatively matches the findings of our simulations and prior work: even without early stopping, overtraining is reduced by increasing the number of hidden units N_h when $N_h > P$.

Our results begin to shed light on the diverse learning behaviors observed in deep networks, and support an emerging view that suggests erring on the side of large models before pruning them later.

2. Generalization dynamics in shallow linear neural networks

To begin, we study the simple case of generalization dynamics in a shallow linear neural network receiving N -dimensional inputs. We consider a standard student–teacher formulation, a setting that has been widely studied using statistical mechanics approaches (see Engel & Van Den Broeck, 2001; Seung et al., 1992; Watkin et al., 1993 for reviews, and Baldi & Chauvin, 1991; Baldi, Chauvin, & Hornik, 1990; Baldi & Hornik, 1995; Chauvin, 1990; Dunmur & Wallace, 1999; Kinouchi & Caticha, 1995; Krogh & Hertz, 1992; LeCun et al., 1991 for settings more closely related to the linear analysis adopted in this section). Here a student network with weight vector $w(t) \in \mathbb{R}^{1 \times N}$ trains on examples generated by a teacher network with weights $\bar{w} \in \mathbb{R}^{1 \times N}$ (see Fig. 1). The teacher implements a noisy linear mapping between P inputs $X \in \mathbb{R}^{N \times P}$ and associated scalar outputs $y \in \mathbb{R}^{1 \times P}$,

$$y = \bar{w}X + \epsilon. \quad (1)$$

In the equation above, $\epsilon \in \mathbb{R}^{1 \times P}$ denotes noise in the teacher's output. We will model both the noise ϵ and the teacher weights \bar{w} as drawn *i.i.d.* from a random distribution with zero mean and variance σ_ϵ^2 and $\sigma_{\bar{w}}^2$ respectively. The signal-to-noise ratio $\text{SNR} \equiv \sigma_{\bar{w}}^2 / \sigma_\epsilon^2$ parametrizes the strength of the rule underlying the dataset relative to the noise in the teacher's output. In our solutions, we will compute the generalization dynamics averaged over all possible realizations of \bar{w} to access the general features of the learning dynamics independent of the specific problems encountered. Finally, we assume that the inputs $X_{\mu j}$ are drawn *i.i.d.* from a Gaussian with mean zero and variance $\frac{1}{N}$ so that each example will have an expected norm of one: $\langle \|x^\mu\|_2^2 \rangle = 1$.

The student network is trained using the dataset $\{y, X\}$ to accurately predict outputs for novel inputs $x \in \mathbb{R}^N$. The student is a shallow linear network, such that the student's prediction $\hat{y} \in \mathbb{R}$ is simply $\hat{y}^\mu = w(t) \cdot x^\mu$. To learn its w parameters, the student network will attempt to minimize the mean squared error on the P training samples using gradient descent. The training error is

$$E_t(t) = \frac{1}{P} \sum_{\mu=1}^P \|y^\mu - \hat{y}^\mu\|_2^2, \quad (2)$$

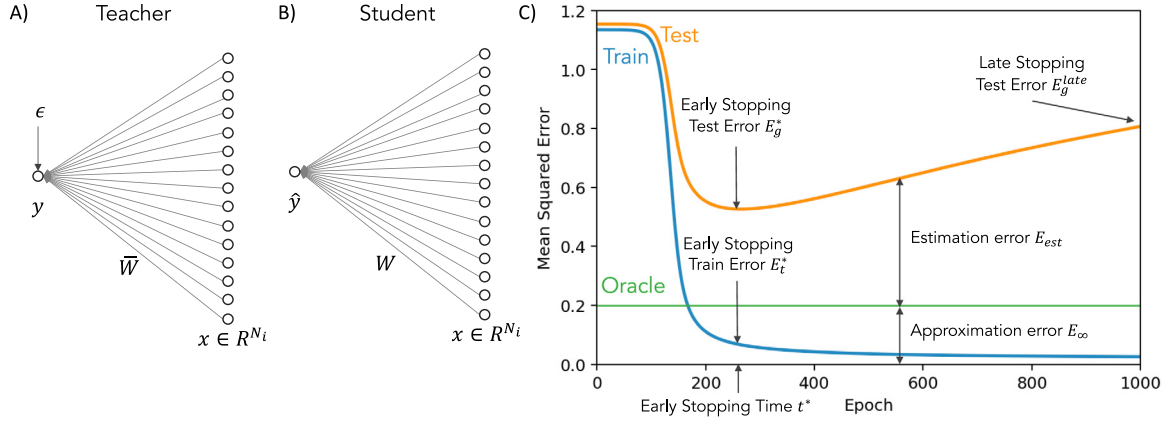


Fig. 1. Learning from a noisy linear teacher. (A) A dataset $\mathcal{D} = \{x^\mu, y^\mu\}$, $\mu = 1, \dots, P$ of P examples is created by providing random inputs x to a teacher network with a weight vector \bar{w} , and corrupting the teacher outputs with noise of variance σ_ϵ^2 . (B) A student network is then trained on this dataset \mathcal{D} . (C) Example dynamics of the student network during full batch gradient descent training. Training error (blue) decreases monotonically. Test error, also referred to as generalization error, (yellow), here computable exactly (4), decreases to a minimum E_g^e at the optimal early stopping time t^* before increasing at longer times (E_g^{late}), a phenomenon known as overtraining. Because of noise in the teacher's output, the best possible student network attains finite generalization error ("oracle", green) even with infinite training data. This error is the approximation error E_∞ . The difference between test error and this best-possible error is the estimation error E_{est} . (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

and full-batch continuous-time gradient descent on this error function (a good approximation to discrete time batch gradient descent with small step size) yields the dynamical equations:

$$\tau \dot{w}(t) = yX^T - wXX^T. \quad (3)$$

where τ is a time constant inversely proportional to the learning rate. Our primary goal is to understand the evolving generalization performance of the network over the course of training. That is, we wish to understand the generalization error which is an average error on a new, unseen example:

$$E_g(t) = \langle (y - \hat{y})^2 \rangle_{x,y} \quad (4)$$

as training proceeds. Here the average $\langle \cdot \rangle_{x,y}$ is over potential inputs x and labels y . The generalization error will be self-averaging meaning that the performance of all random instantiations of the data and model should be the same when the data is sufficiently high dimensional: $N, P \rightarrow \infty$.

2.1. Exact solutions in the high-dimensional limit

How does generalization performance evolve over the course of training? The long-term behavior is clear: if training is run for a long time and initialized with zero weights, the weight vector w will converge to the subspace minimizing the training error (2), solving the least-squares regression problem,

$$w(t \rightarrow \infty) = yX^T(XX^T)^\dagger, \quad (5)$$

where \dagger denotes the Moore–Penrose pseudoinverse. The error achieved by the pseudoinverse, along with more general dynamical results for even nonlinear students and teachers, have long been known (Krogh & Hertz, 1992; LeCun et al., 1991; Seung et al., 1992). Here we explicitly study the time course with which parameters are learned, and pursue connections to phenomena in modern deep learning.

First, we decompose the input data using the singular value decomposition $X = V\Lambda^{1/2}U^T$ (with $U \in \mathbb{R}^{P \times N}$, $\Lambda^{1/2} \in \mathbb{R}^{N \times N}$, and $V \in \mathbb{R}^{N \times N}$). The input correlation matrix is then

$$\Sigma^{xx} = XX^T = V\Lambda V^T. \quad (6)$$

Next we write the input–output correlation matrix as

$$\Sigma^{yx} = yX^T = \tilde{s}V^T, \quad (7)$$

where the row vector \tilde{s} will be called an alignment vector since it is related to the alignment between the input–output correlations and the input correlations.

Returning to the dynamics in (3), we make a change of variables to instead track the vector $z \in \mathbb{R}^{1 \times N}$ where $w = zV^T$. Applying (7) yields:

$$\tau \dot{z}(t) = \tilde{s} - z\Lambda. \quad (8)$$

To make sense of the preceding equation we compute \tilde{s} using the fact that

$$yX^T = \bar{w}XX^T + \epsilon X^T = \bar{z}\Lambda V^T + \tilde{\epsilon}\Lambda^{1/2}V^T, \quad (9)$$

where we define $\bar{z} = \bar{w}V$, and under the assumption of white Gaussian noise, $\tilde{\epsilon} \in \mathbb{R}^N$ has *i.i.d.* elements drawn from a Gaussian with variance σ_ϵ^2 . The form of Σ^{yx} implies $\tilde{s} = \bar{z}\Lambda + \tilde{\epsilon}\Lambda^{1/2}$, so we can rewrite (8) as

$$\tau \dot{z}_i = (\bar{z}_i - z_i)\lambda_i + \tilde{\epsilon}_i \sqrt{\lambda_i}, \quad i = 1, \dots, N. \quad (10)$$

The learning speed of each mode is independent of the others and depends only on the eigenvalue of the mode in question. As we will see, in the case of deep learning, there will be coupling between these modes. However in shallow neural networks, such coupling does not occur and we can solve directly for the dynamics of these modes. The error in each component of z is simply

$$\bar{z}_i - z_i = (\bar{z}_i - z_i(0))e^{-\frac{\lambda_i t}{\tau}} - \frac{\tilde{\epsilon}_i}{\sqrt{\lambda_i}}(1 - e^{-\frac{\lambda_i t}{\tau}}). \quad (11)$$

It follows from the definition of z that the generalization error as a function of training time is

$$E_g(t) = \frac{1}{N} \sum_i \langle (\bar{z}_i - z_i)^2 \rangle + \sigma_\epsilon^2 \quad (12)$$

$$= \frac{1}{N} \sum_i \left[(\sigma_{\bar{w}}^2 + (\sigma_w^0)^2) e^{-\frac{2\lambda_i t}{\tau}} + \frac{\sigma_\epsilon^2}{\lambda_i} (1 - e^{-\frac{\lambda_i t}{\tau}})^2 \right] + \sigma_\epsilon^2, \quad (13)$$

where the second equality follows from the assumption that the teacher weights \bar{w} and initial student weights $w(0)$ are drawn *i.i.d.* from Gaussian distributions with standard deviation $\sigma_{\bar{w}}$ and σ_w^0 respectively. By denoting the spectrum of data covariance

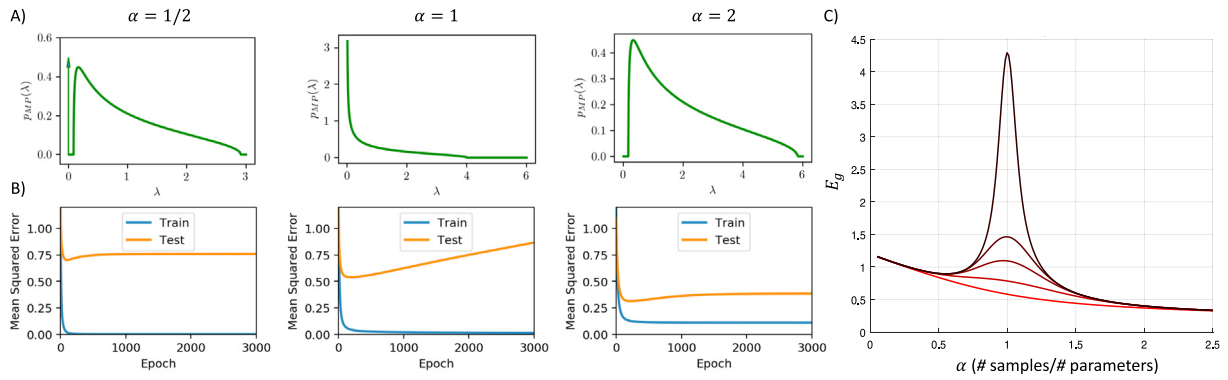


Fig. 2. The Marchenko–Pastur distribution and high-dimensional learning dynamics. (A) Different ratios of number training samples (P) to network parameters (N) ($\alpha = \frac{P}{N}$) yield different eigenvalue densities in the input correlation matrix. For large N , this density is described by the MP distribution (15), which consists of a ‘bulk’ lying between $[\lambda_-, \lambda_+]$, and, when $\alpha < 1$, an additional delta function spike at zero. When there are fewer samples than parameters ($\alpha < 1$, left column), some fraction of eigenvalues are exactly zero (delta-function arrow at origin), and the rest are appreciably greater than zero. When the number of samples is on the order of the parameters ($\alpha = 1$, center column), the distribution diverges near the origin and there are many nonzero but arbitrarily small eigenvalues. When there are more samples than parameters ($\alpha > 1$, right column), the smallest eigenvalues are appreciably greater than zero. (B) Dynamics of learning. From (13), the generalization error is harmed most by small eigenvalues; and these are the slowest to be learned. Hence for $\alpha = 1/2$ and $\alpha = 2$, the gap in the spectrum near zero protects the dynamics from overtraining substantially (eigenvalues which are exactly zero for $\alpha = 1/2$ are never learned, and hence contribute a finite error but no overtraining). For $\alpha = 1$, there are arbitrarily small eigenvalues, and overtraining is substantial. (C) Plot of generalization error versus α for several training times, revealing a clear spike near $\alpha = 1$. Other parameters: $N = 100$, $\text{INR} = 0$, $\text{SNR} = 5$. As the colors vary from red to black the training time increases $\frac{t}{\tau} = [5, 20, 50, 100, 1000]$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

eigenvalues as $\rho(\lambda)$, the generalization error dynamics can be written as:

$$\frac{E_g(t)}{\sigma_w^2} = \int \rho(\lambda) \left[(1 + \text{INR})e^{-\frac{2\lambda t}{\tau}} + \frac{1}{\lambda \cdot \text{SNR}}(1 - e^{-\frac{\lambda t}{\tau}})^2 \right] d\lambda + \frac{1}{\text{SNR}}, \quad (14)$$

where we have normalized by σ_w^2 to set the scale and defined the initialization noise ratio $\text{INR} \equiv (\sigma_w^0)^2 / \sigma_w^2$.

The generalization error expression contains two time-dependent terms. The first term exponentially decays to zero, and encodes the distance between the weight initialization and the final weights to be learned. The second term begins at zero and exponentially approaches its asymptote of $\frac{\sigma_w^2}{\lambda_i}$. This term corresponds to overfitting the noise present in the particular batch of samples. We note two important points: first, eigenvalues which are exactly zero ($\lambda_i = 0$) correspond to directions with no learning dynamics (the right hand side of Eq. (10) is exactly zero) so that the parameters z_i will remain at $z_i(0)$ indefinitely. These directions form a *frozen subspace* in which no learning occurs. While in the original basis all weights typically change during learning, rotation into the principal component directions of the input data reveals these exactly frozen directions. Hence, if there are zero eigenvalues, weight initializations can have a lasting impact on generalization performance even after arbitrarily long training. Second, smaller eigenvalues lead to the most serious overfitting due to the $\frac{\sigma_w^2}{\lambda_i}$ factor in the second term of the generalization error expression. The origin of this phenomena is that noise projected onto these smaller modes cause a large increase in the network weights (see (11)). Hence a large *eigengap* between zero and the smallest nonzero eigenvalue can naturally protect against overfitting. Moreover, smaller eigenvalues are also the slowest to learn, suggesting that early stopping can be an effective strategy, as we demonstrate in more detail later. Finally, the expression provides insight into the time required for convergence of the training dynamics. Non-zero but small eigenvalues of the sample covariance lead to very slow dynamics, so that it will take on the order of $t = \frac{\tau}{\lambda_{\min}}$ for gradient descent to minimize the training error.

The generalization dynamics (13) reveal the critical role played by the eigenvalue spectrum of the sample input covariance matrix. If there are many small eigenvalues generalization performance will be poor due to the growth in the norm of the student weights.

In many interesting cases, the distribution of the eigenspectrum of the sample covariance matrix is known in the high-dimensional limit where the input dimension N and number of examples P jointly go to infinity, while their measurement density $\alpha = P/N$ remains finite. In particular, in the setting in which the inputs are drawn iid from a gaussian with zero mean and variance $1/N$, the eigenvalue distribution of XX^T approaches the Marchenko–Pastur distribution (LeCun et al., 1991; Marchenko & Pastur, 1967) in the high dimensional limit:

$$\rho^{\text{MP}}(\lambda) = \frac{1}{2\pi} \frac{\sqrt{(\lambda_+ - \lambda)(\lambda - \lambda_-)}}{\lambda} + 1_{\alpha < 1}(1 - \alpha)\delta(\lambda), \quad (15)$$

for $\lambda = 0$ or $\lambda \in [\lambda_-, \lambda_+]$, and is zero elsewhere. Here the edges of the distribution take the values $\lambda_{\pm} = (\sqrt{\alpha} \pm 1)^2$ and hence depend on the number of examples relative to the input dimension. Fig. 2A depicts this eigenvalue density for three different values of the measurement density α . In the under-sampled regime when $\alpha < 1$, there are fewer examples than input dimensions and many eigenvalues are exactly zero, yielding the delta function at the origin. This corresponds to a regime where data is scarce relative to the size of the model. In the critically sampled regime $\alpha = 1$, there are exactly as many parameters as examples. Here the distribution diverges near the origin, with increasing probability of extremely small eigenvalues. This situation yields catastrophic overtraining. Finally in the oversampled regime when $\alpha > 1$, there are more examples than input dimensions, yielding the traditional asymptotic regime where data is abundant. The eigenvalue distribution in this case is shifted away from the origin.

Fig. 2B shows the generalization dynamics resulting from substituting (15) into (14) in the under-, critically- and over-sampled regimes. While all three exhibit overfitting, this is substantially worse at the intermediate point $\alpha = 1$. Fig. 2C systematically traces out generalization performance as a function of α at several training times, showing a clear divergence at $\alpha = 1$. Hence

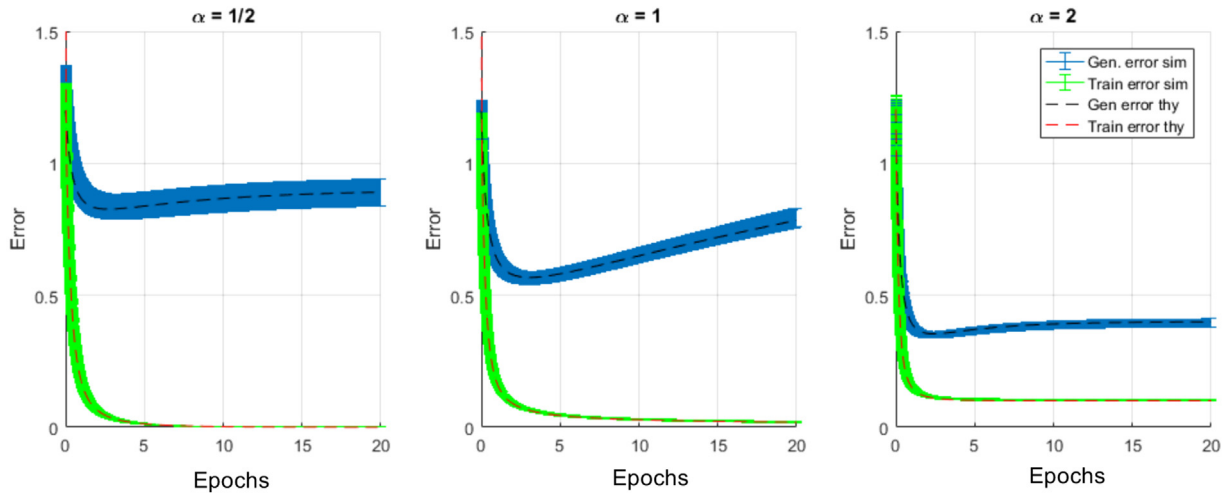


Fig. 3. Generalization and training error dynamics compared with simulations. Here we demonstrate that our theoretical predictions for generalization and training error dynamics at different measurement densities ($\alpha = 1/2$, $\alpha = 1$, and $\alpha = 2$) match with simulations. The simulations for generalization (blue) and training (green) errors have a width of ± 2 standard deviations generated from 20 trials using $N = 300$, $P = \alpha N$ and Gaussian input, noise, and parameters with $\sigma_w = 1$, $\sigma_w^0 = 0$, and $\text{SNR} = 5$. The simulations show excellent agreement with our theoretical predictions for generalization and training error provided in (14) and (26). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

overtraining can lead to complete failure of generalization performance when the measurement density α is close to 1. The generalization error prediction (14) is also validated in Fig. 3 where we compare theory with simulations.

Remarkably, even this simple setting of a linear student and teacher yields generalization dynamics with complex overtraining phenomena that depend on the parameters of the dataset. In the following sections we explore aspects of these solutions in greater detail, and show that early stopping provides an effective remedy to combat overtraining.

2.2. Effectiveness of early stopping

The generalization curves in Fig. 2B improve for a period of time before beginning to worsen and converge to the performance of $\lim_{t \rightarrow \infty} w(t)$. Thus, there will be an optimal stopping time at which simply ending training early would yield improved generalization compared to training for longer times. The intuitive explanation for this is that by limiting time we are effectively regularizing the parameters being learned. This early stopping strategy is widely used in practice, and has been studied theoretically (Amari, Murata, & Müller, 1996; Amari, Murata, Müller, Finke, & Yang, 1995; Baldi & Hornik, 1995; Biehl, Caticha, & Riegler, 2009; Chauvin, 1990; Dodier, 1995; Yao, Rosasco, & Caponnetto, 2007). To solve for the optimal stopping time numerically, we can differentiate (14) with respect to t and set the result equal to zero to solve for t^{opt} . To gain qualitative insights into this problem, we solve for the stopping time that minimizes the error of each mode, yielding:

$$t^{opt} = \frac{\tau}{\lambda} \log(\text{SNR} \cdot \lambda + 1). \quad (16)$$

where we assume for simplicity $w(0) = 0$. This qualitative scaling is verified in Fig. 4A. The intuition behind the logarithmic growth of optimal stopping time with SNR is that high quality data requires less regularization. However optimal stopping time does not achieve optimal stopping on every mode if there is a spread of eigenvalues in the input covariance, and to accurately select the optimal stopping time more generally, we must select t to minimize Eq. (14), as done numerically in Fig. 4B. This reveals a dependence on measurement density, with longest training around $\alpha = 1$ where test error peaks.

2.3. L2 regularization

Another approach is the use of L2 regularization (also called weight decay), because the divergence of generalization error in the critical regime is due to divergence in the network weights. Here we compare the performance of early stopping of gradient descent to the performance of L2 regularization, which corresponds to solving the optimization problem:

$$\hat{w}_{L2} = \arg \min_w \frac{1}{2} \left[\|y - wX\|_2^2 + \frac{\gamma}{2} \|w\|_2^2 \right]. \quad (17)$$

This regularization yields a generalization error (Advani & Ganguli, 2016b) of the form:

$$\frac{E_g^{L2}(\gamma)}{\sigma_w^2} = \int \rho(\lambda) \left[\frac{\gamma^2}{(\lambda + \gamma)^2} + \frac{\lambda}{\text{SNR}(\lambda + \gamma)^2} \right] d\lambda + \frac{1}{\text{SNR}}. \quad (18)$$

Differentiating the above expression with respect to γ , one finds that the optimal performance of this algorithm occurs when the regularization strength is tuned to be inversely proportional to the signal-to-noise ratio: $\gamma = \frac{1}{\text{SNR}} = \frac{\sigma_w^2}{\sigma_y^2}$. In fact, under the assumption of noise, parameter, and input distributions drawn *i.i.d.* Gaussian, no algorithm can out-perform optimal L2 regularization in terms of generalization error performance, as shown in Advani and Ganguli (2016a) and Advani and Ganguli (2016c). Here we ask how close the performance of early stopping comes to this best-possible performance. In fact, if we substitute (16) for each mode into the expression for the time dependent generalization error (that is, if we could early-stop each mode individually), we obtain the same generalization performance as optimal L2 regularization, hinting at a strong relationship between the two methods. However, as mentioned above, early stopping generally performs suboptimally. We compare the two algorithms in Fig. 5A. There is a very close match between their generalization performances: here the relative error between the two is under 3 percent and is the largest around $\alpha = 1$. This near-optimality of early stopping disappears when the initialization strengths of the network weights are large as shown in 5B. Hence early stopping can be a highly effective remedy against overtraining when the initial weights are small.

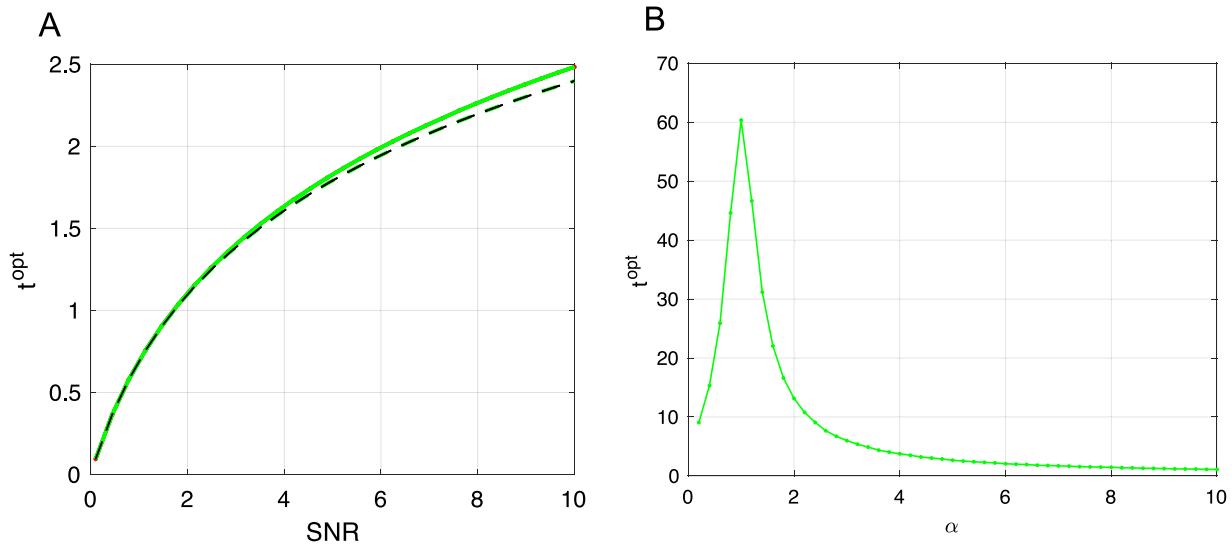


Fig. 4. Impact of data quality and dimensionality on optimal stopping time. In (A) we plot the impact of SNR on the optimal stopping time with low measurement density $\alpha = .05$, and compare this to the predictions of (16) with $\lambda = 1$ because the non-zero values of the MP distribution are highly peaked around this value at low measurement density. In (B) we plot the impact of measurement density on the optimal stopping time in the low noise (SNR = 100) limit, where longer training is required near $\alpha = 1$ because the high quality of the data makes it beneficial to learn weights even in the small eigenvalue directions. In both plots green curves are optimal stopping time numerical predictions computed via gradient descent on t using (14). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

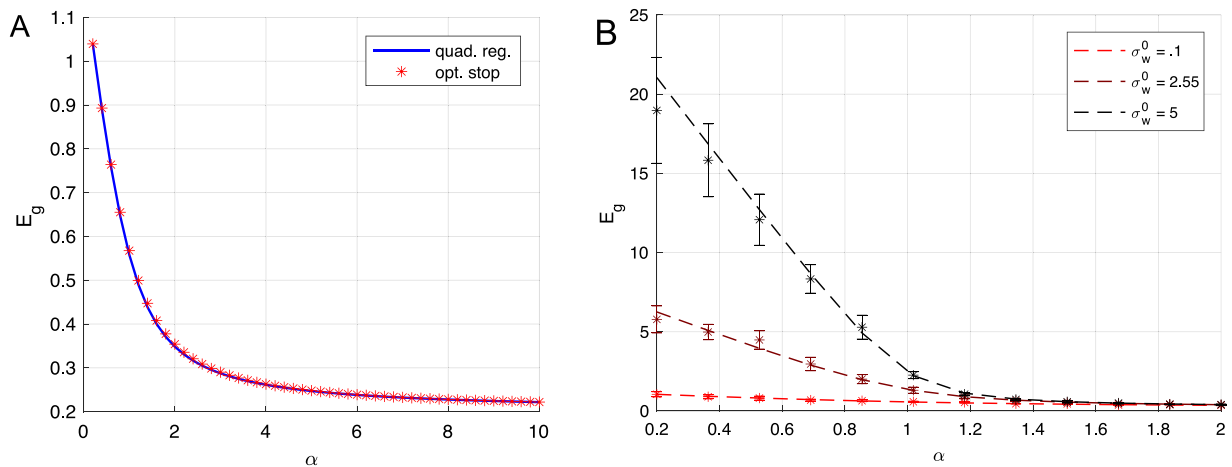


Fig. 5. Optimal stopping performance. In (A) we plot the generalization error from early stopping (red stars) versus optimal quadratically regularized regression (blue line). In this example there is no more than 3 percent relative error between the two, which peaks near $\alpha = 1$ when the spectrum of non-zero eigenvalues is broader. In (B) we plot generalization error vs. α for different initial weight norms. In the limited data regime, small initial weights are crucial for good generalization. In both plots, we have fixed SNR = 5 and $\sigma_w^2 = 1$ which implies that even at large measurement density α the generalization error asymptotes to a non-zero value due to noise $0.2 = \frac{1}{SNR}$.

2.4. Weight norm growth with SNR

Given that longer training times provide the opportunity for more weight updates, one might also expect that the norm of the weights at the optimal early stopping time would vary systematically with SNR. Here we show that this intuition is correct. Assuming zero initial student weights, the average norm of the optimal quadratic regularized weights (derived in Appendix B) increases with SNR as:

$$\langle \hat{w}_{L2}^2 \rangle = \sigma_w^2 \int \rho^{MP}(\lambda) \left(\frac{\lambda}{\lambda + \frac{1}{SNR}} \right) d\lambda. \tag{19}$$

The intuition behind this equation is that data with a lower SNR requires greater regularization which leads to a lower norm in the learned weights. The norm of the learned weights interpolates between $\langle \hat{w}^2 \rangle \rightarrow \sigma_w^2$ as $SNR \rightarrow \infty$ and $\langle \hat{w}^2 \rangle \rightarrow 0$ as $SNR \rightarrow 0$.

Thus the signal-to-noise ratio of the data plays a pivotal role in determining the norm of the learned weights using optimal regularization. In this setting, a well chosen algorithm leads to larger weights in the network when the data is easier to predict.³

³ This dependence of weight norm on SNR may help reconcile puzzles in recent empirical papers. For instance, Section 5 of Zhang et al. (2017) compares linear maps learned off of two different nonlinear transformations of the input, and finds that the linear map with larger norm sometimes generalizes better—in apparent contradiction to generalization bounds that rely on norm-based capacity control. However, based on our results, this finding is expected if one nonlinear transformation reveals more signal than the other: the transformation yielding higher SNR will require a longer training time, yield larger norm weights, and attain better generalization performance. Here the larger norm weights are justified by the stronger rule linking input and output. Comparisons between linear maps of different norms learned from the same nonlinear transformation, however, will reveal an optimal weight norm (which can be approximately found through early stopping).

2.5. Impact of initial weight size on generalization performance

In the preceding sections, we have shown the effectiveness of early stopping provided that the student is initialized with zero weights (INR ≈ 0). The early stopping strategy leaves weights closer to their initial values, and hence will only serve as effective regularization when the initial weights are small. To elucidate the impact of weight initialization on generalization error, we again leverage the similar performance of early stopping and L2 regularization when INR ≈ 0 . If the student weights are not initially zero but instead have variance $(\sigma_w^0)^2$, the optimal L2 regularization strength is $\gamma^{\text{opt}} = \frac{\sigma_\epsilon^2}{(\sigma_w^0)^2 + \sigma_w^2}$. This yields the generalization error,

$$E_g = \frac{(\sigma_w^0)^2 + \sigma_w^2}{2} \left(1 - \alpha - \gamma^{\text{opt}} + \sqrt{(\gamma^{\text{opt}} + \alpha - 1)^2 + 4\gamma^{\text{opt}}} \right) + \sigma_\epsilon^2. \quad (20)$$

In the limit of high SNR, there are three qualitatively different behaviors for the dependence of generalization performance on the initial weight values in the under-sampled, equally sampled, and oversampled regimes. There is a linear dependence on initial weight size when $\alpha < 1$, a square root dependence for $\alpha = 1$, and no dependence when $\alpha > 1$,

$$E_g \approx \begin{cases} ((\sigma_w^0)^2 + \sigma_w^2)(1 - \alpha) + \sigma_\epsilon^2, & \alpha < 1 \\ \sqrt{(\sigma_w^0)^2 + \sigma_w^2} \sigma_\epsilon + \sigma_\epsilon^2, & \alpha = 1 \\ \sigma_\epsilon^2 \frac{\alpha}{\alpha - 1}, & \alpha > 1. \end{cases}$$

Hence as more data becomes available, the impact of the weight initialization decreases. Our dynamical solutions reveal the source of this effect. When there are few examples relative to the input dimension, there are many directions in weight space that do not change because no examples lie in those directions, such that the component of weight initialization in these directions will remain indefinitely. It is therefore critical to initialize with small weights to maximize generalization performance in the overparametrized regime, as shown in Fig. 5B. Even when the number of examples is matched to the size of the model ($\alpha \approx 1$), large-norm weight initializations remain detrimental for the optimal early stopping strategy because training must terminate before the influence of the initial weights has fully decayed. Hence, based on this simple linear analysis, excellent generalization performance in large models can be obtained through the two-part strategy of early stopping and initialization with small-norm weights. Remarkably, this strategy nearly matches the performance of the best possible Bayes-optimal learning strategy (optimal L2 regularization) for the setting we consider, and remains effective in the high-dimensional regime where the number of samples can be smaller than the size of the model.

2.6. Training error dynamics

Next we turn to the dynamics of the training error during learning. As we will show, these dynamics depend strongly on the small eigenvalues of the data covariance and weakly on the SNR. To derive the training error dynamics, we begin with the form:

$$E_t(t) = \frac{1}{P} \|y - w(t)X\|_2^2. \quad (21)$$

Substituting the singular value decomposition of the data $X = V\Lambda^{1/2}U^T$, and $w(t) = z(t)V^T$ yields:

$$\frac{1}{P} \|y - z(t)\Lambda^{1/2}U^T\|_2^2. \quad (22)$$

In the oversampled setting ($P > N$), we define $\tilde{U} = (U, U^\perp) \in \mathbb{R}^{P \times P}$, and if $P \leq N$ we let $\tilde{U} = U$ so that in both cases $\tilde{U}\tilde{U}^T = I$. Rearranging the training error yields:

$$\begin{aligned} E_t(t) &= \frac{1}{P} \|y\tilde{U} - z(t)\Lambda^{1/2}U^T\tilde{U}\|_2^2 \\ &= \frac{1}{P} \|\tilde{\epsilon} + (\bar{z} - z(t))\Lambda^{1/2}\|_2^2 + \|\epsilon U^\perp\|_2^2. \end{aligned} \quad (23)$$

Here $\tilde{\epsilon} = \epsilon U$. From (11), we derived that the learning dynamics in linear networks follow:

$$\tilde{\epsilon}_i + \sqrt{\lambda_i}(\bar{z}_i - z_i(t)) = \left(\sqrt{\lambda_i}(\bar{z}_i - z_i(0)) + \tilde{\epsilon}_i \right) e^{-\frac{\lambda_i t}{\tau}}. \quad (24)$$

Thus, we may write the training error as a function of time as:

$$E_t(t) = \frac{1}{P} \left(\sum_{i=1}^N (\sqrt{\lambda_i}(\bar{z}_i - z_i(0)) + \tilde{\epsilon}_i)^2 e^{-\frac{2\lambda_i t}{\tau}} + \sum_{j=1}^{P-N} (\tilde{\epsilon}_j^\perp)^2 \right), \quad (25)$$

where $\tilde{\epsilon}^\perp = \epsilon U^\perp$ and the second sum in the expression above equals zero if $P \leq N$. Note that the training error is strictly decaying with time. If $P < N$, the training error will approach zero as each data point is memorized. If we average the training error dynamics over the noise, parameter, and data distributions, we find:

$$\begin{aligned} \langle E_t(t) \rangle &= \frac{1}{\alpha} \int \rho(\lambda) (\lambda(\sigma_w^2 + (\sigma_w^0)^2) + \sigma_\epsilon^2) e^{-\frac{2\lambda t}{\tau}} d\lambda \\ &\quad + \left(1 - \frac{1}{\alpha} \right) \sigma_\epsilon^2 \mathbf{1}[\alpha > 1]. \end{aligned} \quad (26)$$

See Fig. 3 for demonstration that the theoretical prediction above matches the results of simulations.

This formula for training error dynamics may help explain recent empirical findings (e.g. Arpit et al., 2017; Zhang et al., 2017) that neural networks do not find it difficult to memorize noisy labels and require only slightly longer training times to do so. If we consider the amount of training time before E_t reaches some pre-determined small value, this will increase slightly as noise is added to the output labels. If we consider the undersampled setting in which noise can be memorized ($\alpha < 1$), then at very late times ($t \gg \frac{\tau}{\lambda_{\min}}$) the training error will decay exponentially, so that to a good approximation:

$$E_t(t) \propto \left(\lambda_{\min} + \frac{1}{\text{SNR}} \right) e^{-\frac{2\lambda_{\min} t}{\tau}}. \quad (27)$$

It follows that the time required to reach a training error proportional to δ scales as:

$$t \approx \frac{\tau}{2\lambda_{\min}} \log \left(\frac{\lambda_{\min} + \frac{1}{\text{SNR}}}{\delta} \right). \quad (28)$$

Thus, increasing the variance in the output noise should lead to only a logarithmic increase in the time required to memorize a dataset.

3. Generalization dynamics in deep linear neural networks

The results in preceding sections have focused on the simplest case of a shallow neural network. We now turn to the question of whether similar qualitative properties occur in a class of simple deep networks. We consider a deep linear student network (Fig. 6A) with weights W_1, \dots, W_D forming a $(D + 1)$ -layer linear network. In response to an input $x \in \mathbb{R}^N$, the student produces a scalar prediction $\hat{y} \in \mathbb{R}$ following the rule $\hat{y} = W_D W_{D-1} \dots W_2 W_1 x = W^{\text{tot}} x$. We initially make no assumptions about the size of the layers, except for the final layer which

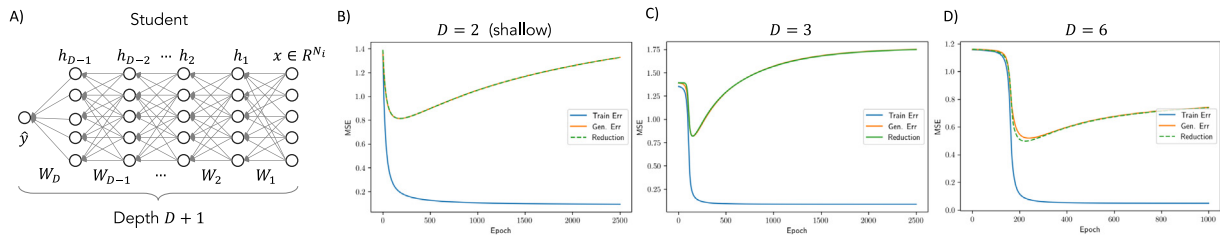


Fig. 6. Reduction of deep linear network dynamics. (A) The student network is a deep linear network of depth $D + 1$. (B–D) Comparisons of simulations of the full generalization dynamics for networks initialized with small weights (gold) to simulations of the reduced dynamics (green) for different depths and parameter settings. Training error in blue. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

produces a single scalar output. Continuous time gradient descent on the mean squared training error yields the dynamics,

$$\tau \frac{d}{dt} W_l = \left(\prod_{i=l+1}^D W_i \right)^T \left[\Sigma^{yx} - \left(\prod_{i=1}^D W_i \right) \Sigma^{xx} \right] \left(\prod_{i=1}^{l-1} W_i \right)^T. \quad (29)$$

These equations involve coupling between weights at different layers of the network, and nonlinear interactions between the weights. While they compute a simple input–output map, deep linear networks retain several important features of nonlinear networks: most notably, the learning problem is nonconvex (Baldi & Hornik, 1989; Kawaguchi, 2016), yielding nonlinear dynamical trajectories (Arora, Cohen, & Hazan, 2018; Du, Hu, & Lee, 2018; Fukumizu, 1998; Ji & Telgarsky, 2019; Saxe et al., 2014).

We wish to find a simplified expression that provides a good description of these dynamics starting from small random weights. Intuitively, the full gradient dynamics contain repulsion forces that attempt to orthogonalize the hidden units, such that no two represent the same feature of the input. When initializing with small random weights, however, hidden units will typically already be nearly orthogonal—and hence these orthogonalizing repulsive forces can be safely neglected as in Saxe et al. (2014). We begin by using an SVD-based change of variables to reduce coupling in these equations. We then will study the dynamics assuming that hidden units are initially fully decoupled—since this remains a good approximation for the full dynamics when starting from small random weights, as verified through simulation.

In particular, based on the observation that gradient descent dynamics in deep linear networks initialized with small weights extract low rank structure (Saxe et al., 2014), we make the following change of variables: $W_l(t) = r_l z(t) V^T$, where $z(t) \in \mathbb{R}^{1 \times N_i}$ is a row vector encoding the time-varying overlap with each principle axis in the input (recall $\Sigma^{xx} = V \Lambda V^T$); and $W_l(t) = d(t) r_{l+1} r_l^T$, $l = 2, \dots, D$ where the vectors $r_i \in \mathbb{R}^{N_i \times 1}$ are arbitrary unit norm vectors ($r_l^T r_l = 1$) specifying freedom in the internal representation of the network, r_{D+1} is defined to be one, and $d(t)$ is a scalar which encodes the change in representation over time. With these definitions, $W^{\text{tot}} = \left(\prod_{l=2}^D d(t) r_{l+1} r_l^T \right) r_2 z(t) V^T = d(t)^{D-1} z(t) V^T = u(t) z(t) V^T$ where we have defined the scalar $u(t) = d(t)^{D-1}$. We show in Appendix A that solutions which start in this form remain in this form, and yield the following exact reduction of the dynamics:

$$\tau \dot{u} = (D - 1) u \frac{2D-4}{D-1} (\tilde{s} z^T - u z \Lambda z^T), \quad (30)$$

$$\tau \dot{z} = u (\tilde{s} - u z \Lambda). \quad (31)$$

These equations constitute a considerable simplification of the dynamics. Note that if there are $N_h = N_i$ hidden units in each layer, then full gradient descent in a depth $D + 1$ network involves $O(N_i^2 D)$ parameters. The above reduction requires only $N_i + 1$ parameters, regardless of depth. Fig. 6 compares the predicted

generalization dynamics from these reduced dynamics to simulations of full gradient descent for networks starting with small random weights and of different depths, confirming that the reduction provides an accurate picture of the full dynamics when starting from small random weights. A line of work has shown that the full gradient descent dynamics in fact converge to the low rank structure implied by these initialization conditions, in a wide range of settings beyond the student–teacher setting we consider here (Arora et al., 2018; Du et al., 2018; Ji & Telgarsky, 2019). Our result shows that, once attracted to this low rank solution, the generalization dynamics take a particularly simple form that reveals links to the shallow case.

In the reduction, all modes are coupled together through the global scalar $u(t)$. Comparing (31) with (8), this scalar premultiplies the shallow dynamics, yielding a characteristic slow down early in training when both z and u are small (see initial plateaus in Fig. 6C–D). This behavior is a hallmark of deep dynamics initialized close to the saddle point where all weights are zero, and has been found for training error dynamics as well (Goodfellow, Vinyals, & Saxe, 2015; Saxe et al., 2014). Remarkably, in the reduction the entire effect of depth is compressed into the scalar u which sets a global speed of the dynamics shared by all modes. Otherwise, each mode’s dynamics is analogous to the shallow case and driven primarily by the size of its associated eigenvalue λ_i , with smaller eigenvalues being learned later. This suggests that optimal stopping will again be effective in deep networks, with comparable results to the shallow case. And as in the shallow case, eigenvalues that are zero make no progress, yielding a frozen subspace in which no learning occurs. Hence again, large initial weight norms will harm generalization error in the limited data regime in deep linear networks.

Thus for deep linear networks producing scalar outputs, our reduction predicts behavior qualitatively like their shallow counterparts in terms of their early stopping error and sensitivity to weight initializations. However, depth introduces a potential tension between training speed and generalization performance: small initial weights dramatically slow learning in deep networks, but large initial weights can harm generalization in the high-dimensional regime. This may underlie observations in full nonlinear networks that certain initializations can converge quickly but yield poorer generalization performance (e.g., Mishkin & Matas, 2016).

4. Generalization dynamics and optimal size in neural nets with frozen first layer weights

In practical settings, nonlinear activations are essential since they allow networks to express complex functions. In this section we explore the degree to which the qualitative intuitions obtained through studying linear networks transfer to the nonlinear setting by introducing a fixed, random first layer with non-linear activations in the hidden layer. Nonlinearity introduces the crucial question of model complexity: is there an optimal size of the

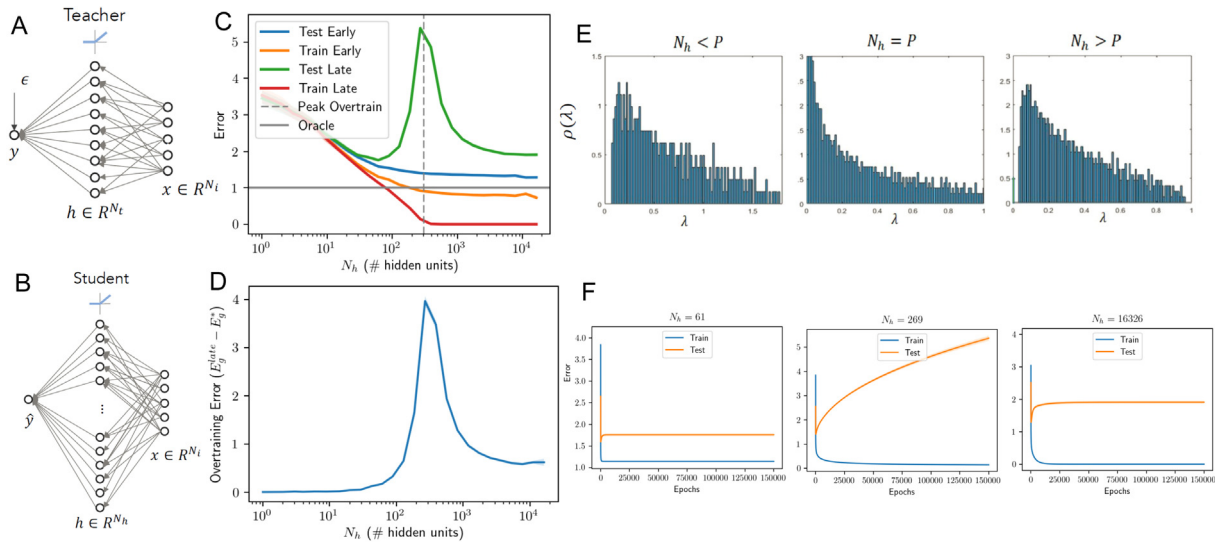


Fig. 7. Learning from a nonlinear teacher. (A) The teacher network (N_t ReLUs). (B) The student network (N_h ReLUs, but only the output weights are trained). (C) Effect of model complexity. Optimal early stopping errors as a function of number of hidden units N_h for the case SNR = 1, INR = 0, $N_i = 15$, $N_t = 30$ and $P = 300$ training samples. Shaded regions show ± 1 standard error of the mean over 50 random seeds. (D) Overtraining peaks at an intermediate level of complexity near the number of training samples: when the number of free parameters in the student network equals the number of samples (300). (E) The eigenspectrum of the hidden layer of a random non-linear neural network with $P = 1000$ samples and an $N_i = 100$ dimensional input space. We consider three cases and find a similar eigenvalue density to a rescaled Marchenko–Pastur distribution when we concentrate only on the small eigenvalues and ignore a secondary cluster of N_i eigenvalues farther from the origin. Left: Fewer hidden nodes than samples ($N_h = 500$ hidden units) leads to a gap near the origin and no zero eigenvalues. Center: An equal number of hidden nodes and samples ($N_h = 1000$) leads to no gap near the origin so that eigenvalues become more probable as we approach the origin. Right: More hidden nodes than samples ($N_h = 2000$) leads to a delta function spike of probability 0.5 at the origin with a gap to the next eigenvalue. (F) Average training dynamics for several models illustrating overtraining at intermediate levels of complexity. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

student, given the complexity of the problem and the number of training samples available?

We begin by simulating the dynamics in a student–teacher scenario (Engel & Van Den Broeck, 2001; Saad & Solla, 1995; Seung et al., 1992; Watkin et al., 1993; Wei & Amari, 2008; Wei, Zhang, Cousseau, Ozeki, & Amari, 2008), as depicted in Fig. 7A–B. The teacher is now a single hidden layer neural network with rectified linear unit (ReLU) activation functions and N_t hidden neurons. We draw the parameters of this network randomly. To account for the compression in variance of the ReLU nonlinearity, we scale the variance of the input-to-hidden weights by a factor $\sqrt{2}$ (He et al., 2016; Saxe et al., 2014). We generate a dataset as before by drawing P random Gaussian inputs, obtaining the teacher’s output y , and adding noise. The student network, itself a single hidden layer ReLU network with N_h hidden units, is then trained to minimize the MSE on this dataset. The procedure is to train only the second layer weights of the student network, leaving the first layer weights randomly drawn from the same distribution as the teacher. This setting most closely recreates the setting of the shallow linear case, as only the hidden-to-output weights of the student change during learning. Due to the nonlinearity, analytical results are challenging so we instead investigate the dynamics through simulations. For the simulations in Fig. 7, we fixed the teacher parameters to SNR = 1, $N_t = 30$ and $N_i = 15$ and the number of training samples to $P = 300$, and then trained student models at a set of hidden layer sizes $N_h = [1, \dots, 16326]$ ranging from $300\times$ smaller than the number of samples to $54\times$ larger. We average results at each size over 50 random seeds. Fig. 7C–D shows average errors as a function of the number of neurons in the student. Additionally, while we can only estimate the generalization error through cross validation in the setting without a teacher network, we find that the qualitative trends observed in the teacher–student framework hold when we instead use the MNIST dataset in Fig. 8.

Our goal is to understand whether the qualitative patterns of overtraining obtained from the linear analysis hold for this nonlinear case, and we find the following empirical observations:

Overtraining occurs at intermediate model complexity. There is a peak in overtraining near where the number of samples in the dataset equals the number of hidden units/free parameters, with large models overtraining less than intermediate-sized models (Fig. 7C–D). Hence qualitatively, the network behaves akin to the linear case, with a peak in overtraining when the number of trainable parameters is matched to the number of training samples. To understand this behavior, we computed the eigenspectrum of the hidden layer activity in Fig. 7E. Despite the nonlinearity, we see qualitatively similar behavior to the linear case: the small eigenvalues approximately follow the Marchenko–Pastur distribution (c.f. Fig. 2). The asymptotic distribution of the eigenvalues of this covariance matrix has been derived in Pennington and Worah (2017), and a similar observation about the small eigenvalues computed from the hidden layer was also used in a slightly different context to study training error in Pennington and Bahri (2017). We provide additional simulations in Fig. 14 showing this overtraining peak for different SNRs, which demonstrate that noisy data will amplify overtraining causing it to grow approximately proportionally to the inverse SNR as should be expected from (13).

Larger models are better when early-stopped. Strikingly, we find no evidence of overfitting: larger models are always better in terms of their early stopping test error. Here the teacher network has just 30 hidden neurons and the dataset consists of just 300 examples. Nevertheless, the best student model is the largest one tested, containing 16326 hidden units, or $344\times$ as many hidden neurons as the teacher and $54\times$ more neurons than training samples. This benefit of large models requires early stopping and initialization with small random weights, and hence reflects regularization through limiting the norm of the solution (see extended comments in Section 7). If instead training is continued to much longer times (green curve Fig. 7C), the optimal model size is smaller than the number of samples (≈ 60 hidden units), consistent with standard intuitions about model complexity. We note that in this setting, although the late test error does not

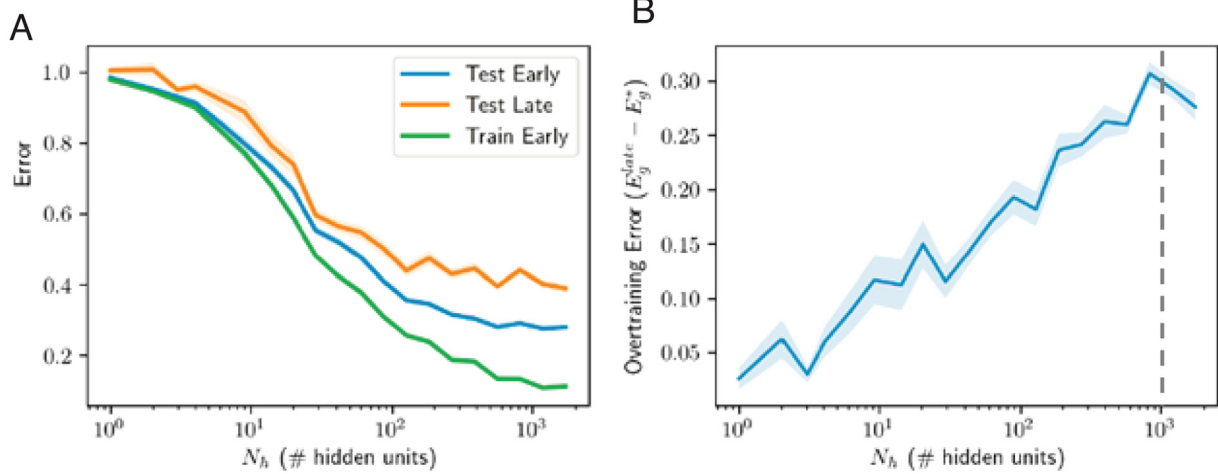


Fig. 8. Training directly on data: two layer ReLU network trained on binary MNIST classification of 7s vs. 9s with 1024 training samples. First layer weights of student are random and fixed. The less pronounced peak is likely due to computational restrictions which limited training time and maximum model size (see text).

diverge to infinity in large models, it is nevertheless better for smaller models, which may appear to conflict with observations in other recent work (Belkin, Hsu, Ma et al., 2019). As we discuss in Section 6, whether an overparametrized model will generalize better than smaller models in our setting depends on the signal-to-noise ratio of the dataset.

Mathematical decomposition of generalization error

To gain insight into this more complex nonlinear case, we derive a decomposition of the test error into several interpretable components. In the setting of a frozen first layer and trained second layer, the generalization error can be decomposed into three parts: an approximation error, an over-fitting error, and a null-space error. This decomposition helps to explain how varying the number of parameters in a network can improve generalization performance when the number of examples is held fixed. It also provides one interpretation of how the student-teacher framework might apply to more realistic settings where the model is not in the same class as the data. In essence, the best possible neural network acts as an effective teacher, with the label noise capturing the remainder of the transformation that the neural network cannot represent. That is, we find that approximation error behaves very similarly to an external noise from the perspective of the neural network in estimating overtraining error. This implies that even in settings when there is no label noise in the problem, there may be an effective noise based on the student network architecture.

We can write a two layer neural network in the form:

$$\hat{y}_a(x) = \sum_{i=1}^{N_h} w_{ai} \phi_i(x). \quad (32)$$

In general ϕ_i is the mapping from the input to a hidden representation and can be nonlinear. Here $x \in \mathbb{R}^{N_i}$, $y \in \mathbb{R}^{N_o}$, and $\phi_i : \mathbb{R}^{N_i} \rightarrow \mathbb{R}$. We leave the specific choice of ϕ_i general, but in a typical two layer network it would have the form: $\phi_i(x) = \sigma(\sum_j J_{ij} x_j)$, where the matrix J represents fixed input to hidden layer weights.

We then train the network using a set of input-output data (x^μ, y^μ) for $\mu = 1, \dots, P$ generated from some process which can be deterministic or stochastic. Assuming a MSE loss function the training error has the form:

$$E_t = \frac{1}{2} \sum_{a,\mu} \left(y_a^\mu - \sum_{i=1}^{N_h} w_{ai} \phi_i(x^\mu) \right)^2. \quad (33)$$

We define the hidden layer activity matrix $\Phi_{i\mu} = \phi_i(x^\mu)$ and the output matrix $Y_{a\mu} = y_a^\mu$.

Assuming initial output weights of zero, the weights that will be learned at late time by the network \hat{w} are:

$$\hat{w} = \hat{R} \hat{C}^\dagger, \quad (34)$$

where we define:

$$\hat{R} = Y \Phi^T, \quad \hat{C} = \Phi \Phi^T. \quad (35)$$

The network weights which minimize the generalization error for the given architecture are simply:

$$w^* = RC^\dagger, \quad (36)$$

where $C_{ij} = \langle \phi_i(x) \phi_j(x) \rangle_x$, and $R_{ai} = \langle y_a(x) \phi_i(x) \rangle_x$.

However, when training on finite data, the student network weights only grow in directions in which the hidden layer covariance is not null, so it is useful to define w_{\parallel}^* to be the weights which minimize generalization error subject to the constraint of being zero in the null space of the empirical hidden layer covariance. If we define the projection onto the space in which the data lies in the hidden layer as $P_{\parallel} = (\Phi \Phi^T)(\Phi \Phi^T)^\dagger$, then we can show that:

$$w_{\parallel}^* = RC^\dagger P_{\parallel}, \quad (37)$$

so that the best possible weights restricted to the subspace parallel to the data are simply the projection of w^* onto this subspace.

We find one can decompose the generalization error into three terms to explain the behavior of late stopping generalization error as:

$$E_g^{late} = \mathcal{E} \Phi^T (\Phi \Phi^T)^\dagger C (\Phi \Phi^T)^\dagger \Phi \mathcal{E}^T + \text{Tr} [RC^\dagger P_{\perp} C P_{\perp}^T C^\dagger R^T] + \langle \epsilon(x)^2 \rangle_x, \quad (38)$$

where we refer to the first term as the *overfitting error* $E_g^{overfit}$, the second term as the *null-space error* E_g^{null} , and the third term as the *approximation error* E_g^{approx} , and we define an approximation noise matrix:

$$\mathcal{E}_{i\mu} = Y_{i\mu} - \sum_j w_{ij}^* \Phi_{j\mu}. \quad (39)$$

The *approximation error* $\langle \epsilon(x)^2 \rangle_x$ is defined as the minimal generalization error that can be achieved given a particular network architecture, by optimizing over the weights. If the network

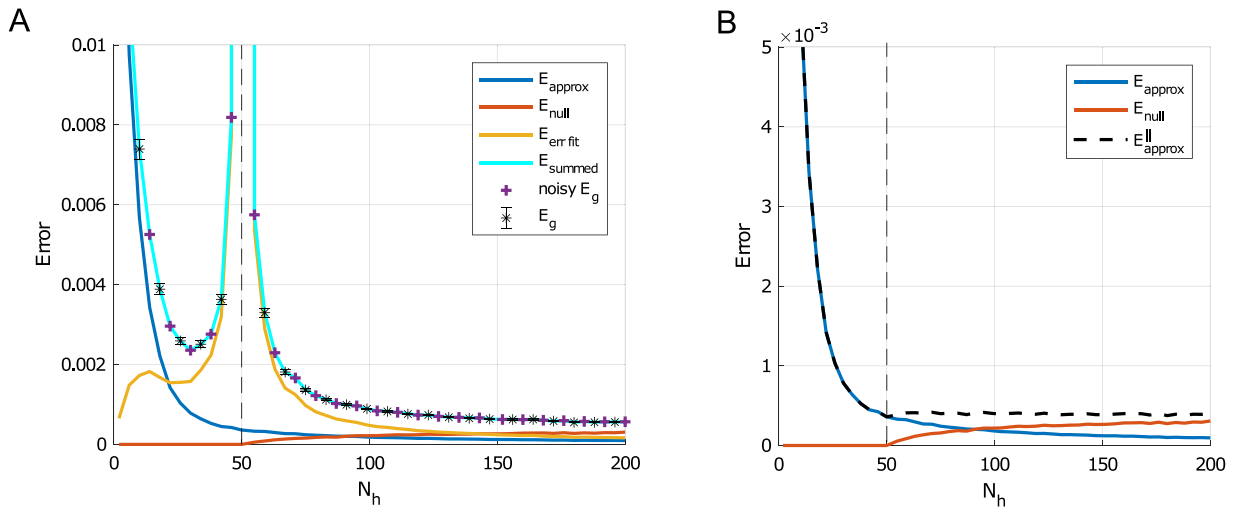


Fig. 9. Two layer network generalization error decomposition: Here we use data generated from a two layer teacher network with ReLU activations, 20 hidden units, and no noise. The student network has fixed random first layer weights and ReLU activations. (A) Shows the approximation error in blue, the null-space error in red, the over-fitting error in gold, and the sum of all three in cyan. Generalization error computed from 5000 new examples are black error bars. Plus sign denotes generalization error with a two layer teacher with w_{ij}^* weights and Gaussian noise with a variance matched to the sum of approximation and null-space error. The number of examples used in training is $P = 50$ (dashed vertical line) and number of trials used to compute error bars is 500. (B) Shows the approximation error (blue) and null space error (red) as well as the sum of the two (dashed black). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

is too small or simple relative to the rule being learned the approximation error will be large.

The *null-space error* can be understood as additional approximation error due to the fact that no learning can take place in the null space of the hidden layer activity, and only occurs in this setting when the number of hidden units exceeds the number of examples.

Finally, the *over-fitting error* is the error due to the fact the data set is finite and depends strongly on the ratio of number of parameters to examples through the eigenvalue decomposition of the empirical hidden covariance matrix. This over-fitting error is highly non-monotonic and can lead to a massive peak in generalization error at the point where the number of examples and free parameters matches. Derivation details on the decomposition calculation may be found in [Appendix C](#). The decomposition is plotted in [Fig. 9A](#) and the sum is compared to the generalization error.

Interestingly, if we replace the actual data with new data generated from a model where noise is restricted to the data subspace:

$$\tilde{y}^\mu = \sum_{j=1}^{N_h} w_{ij}^* \phi_j(x^\mu) + z_a^\mu, \quad (40)$$

where $z_a^\mu \sim \mathcal{N}(0, \langle \epsilon^\parallel(x)^2 \rangle)$, we find a good agreement with the predictions of overfitting error and generalization from the linear network calculations (see [Fig. 9A](#) ‘noisy E_g ’). Here the variance of the effective noise is due to the approximation error restricted to the data subspace:

$$\langle \epsilon^\parallel(x)^2 \rangle = \text{Tr} [RC^\dagger P_\perp C P_\perp^\dagger R^T] + \langle \epsilon(x)^2 \rangle_x. \quad (41)$$

This suggests that we can think of the approximation error as label noise which decays as the architecture grows until the number of parameters added exceeds the number of training examples at which point the restricted approximation error stops decreasing and begins to saturate (see [Fig. 9B](#)). Further generalization error improvement with wider networks in this context is not due to the networks reduced restricted approximation error, but to the eigenspectrum of the empirical co-variance, since the gap between small and zero eigenvalues grows with increased

network width. Thus, the decomposition helps to explain how, even without regularization or early stopping, large neural networks avoid over-fitting and can achieve better generalization error when they have more parameters than examples.

With some additional modifications, it also helps one understand the impact of initialization weights and true label noise on the performance of non-linear networks of different sizes. [Eq. \(38\)](#) gives the generalization error at late times starting with zero initial weights. If instead initial weights $w(0)$ are drawn iid with zero mean and variance $(\sigma_w^0)^2$, then for large networks we can average over this initialization to find:

$$E_g^{\text{late}} = \mathcal{E} \Phi^T (\Phi \Phi^T)^\dagger C (\Phi \Phi^T)^\dagger \Phi \mathcal{E}^T + \text{Tr} [RC^\dagger P_\perp C P_\perp^\dagger R^T] + (\sigma_w^0)^2 \text{Tr} [P_\perp C P_\perp^\dagger] + \langle \epsilon(x)^2 \rangle_x, \quad (42)$$

where we can consider both the second and third terms part of the null-space error because the initialization weights are frozen during training, so that large initialization scales increase the generalization error when $N_h > P$. In addition, *i.i.d.* label noise will appear in the generalization error under the transformation $\langle \epsilon(x)^2 \rangle \rightarrow \langle \epsilon(x)^2 \rangle + \sigma_\epsilon^2$.

We demonstrate in [Fig. 10](#) how label noise makes the shifts in approximation error due to the network architecture less important so that the over-fitting error and hidden layer covariance spectrum are more relevant in determining generalization error. Thus, this setting is most similar to the single layer network dynamics. We also demonstrate how initial weight scale can impact performance in large over-parameterized networks more than under-parameterized ones in [Fig. 11](#). In this case, the null-space error grows with the variance of initialization and causes the generalization error to increase in the $N_h > P$ setting.

5. Fully trained two-layer network

Next, we allowed both layers of the student network to train, as is typically the case in practice. [Fig. 12](#) shows that similar dynamics are obtained in this setting. Overtraining is transient and peaked at intermediate levels of complexity, roughly where the number of free parameters ($N_i \cdot N_h + N_h$) equal the number of examples P . This is the point at which training error hits

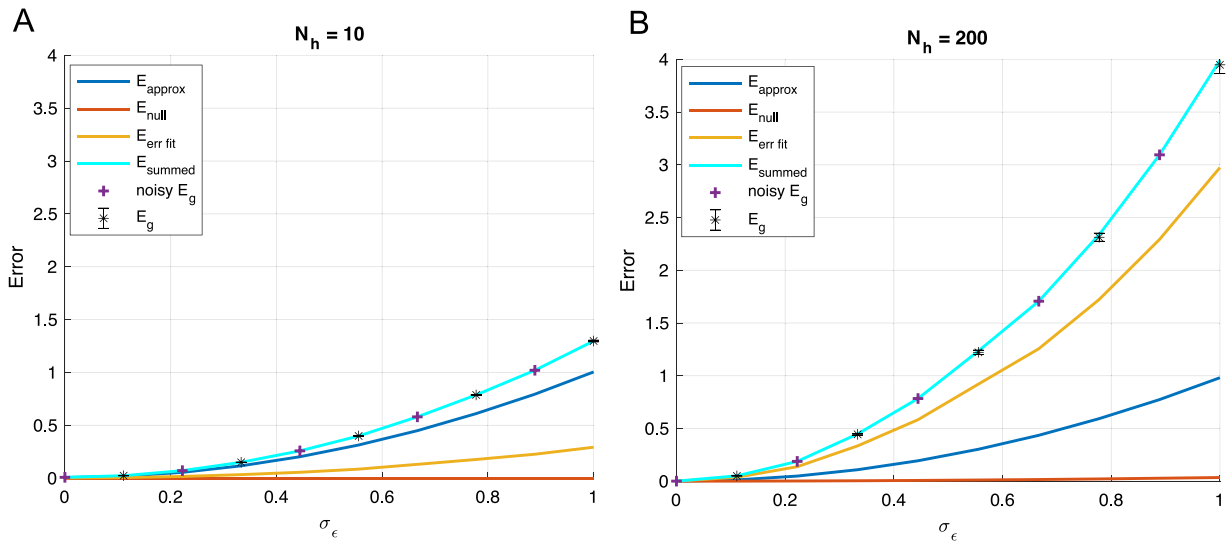


Fig. 10. Impact of label noise on performance of wide networks: We plot the generalization error decomposition as in Fig. 9 for (A) an under-parameterized $N_h = 10$ and (B) an over-parameterized $N_h = 200$ network. Because output noise is added to both networks, approximation of the function is difficult even for the larger network because the function it is learning is not deterministic. However, the fitting error is larger in the larger network due to the eigenvalue spectrum of the hidden layer covariance. This is in contrast to our finding that larger networks perform better in the deterministic, low noise setting depicted in Fig. 9. Other parameters: $P = 50$, $N_t = 20$, $\sigma_w^0 = 0$, $\sigma_{\bar{w}} = 1$.

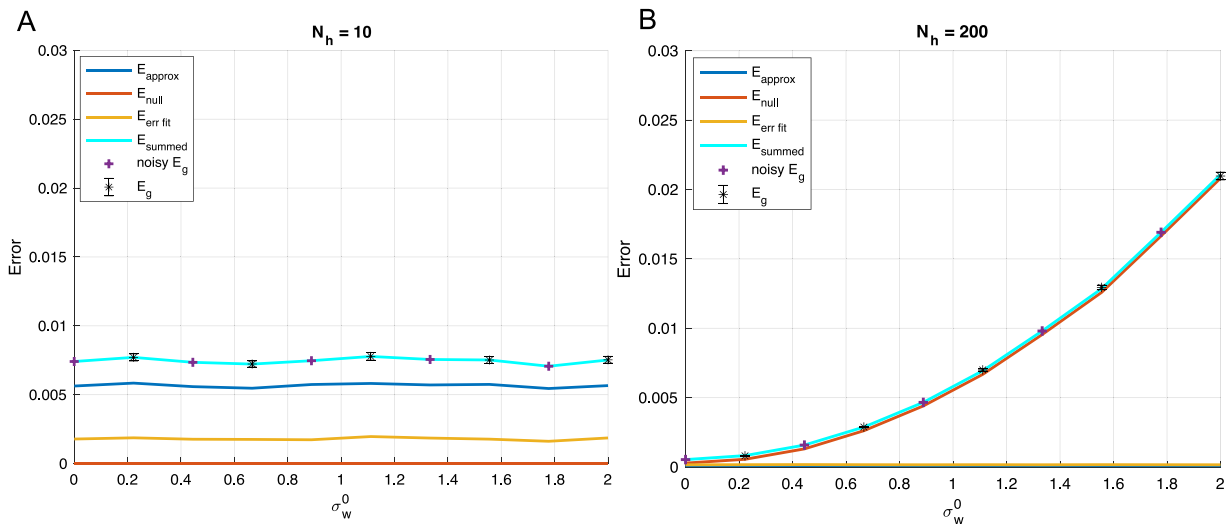


Fig. 11. Impact of initialization on performance of wide networks. We plot the generalization error decomposition as in Fig. 9 for (A) an under-parameterized $N_h = 10$ and (B) an over-parameterized $N_h = 200$ network. We demonstrate that large initializations have a detrimental impact on generalization performance in wide neural networks which are over-parameterized and not on smaller networks. This effect is due to the frozen subspace observed when training with fewer examples than free parameters, and again is in contrast to our finding that larger networks perform better in the low initialization setting depicted in Fig. 9. Other parameters: $P = 50$, $N_t = 20$, $\sigma_\epsilon = 0$, $\sigma_{\bar{w}} = 1$.

zero at long training times (red curve, Fig. 12A), consistent with recent results on the error landscape of ReLU networks (Soudry & Hoffer, 2017; Spigler et al., 2018). Regarding model complexity, again we find that massive models relative to the size of the teacher and the number of samples are in fact optimal, if training is terminated correctly and the initial weights are small. In particular, good performance is obtained by a student with 34,430 hidden neurons, or $1147\times$ the number of teacher neurons and $115\times$ the number of examples. In contrast, without early stopping, the long-time test error has a unique minimum at a small model complexity of ≈ 5 hidden neurons in this instance. Hence as before, early stopping enables successful generalization in very large models, and the qualitative behavior is analogous to the linear case. We caution that the linear analysis can only be expected to hold approximately. As one example of an apparent difference in behavior, we note that the test error curve when the

number of parameters equals the number of samples (Fig. 12C middle) diverges to infinity in the linear and fix hidden weights cases (see Fig. 2B middle and Fig. 7F middle), while it appears to asymptote in the two-layer trained case (see Fig. 12C middle). Nevertheless, this asymptote is much larger at the predicted peak (when the number of free parameters and samples are equal) than at other network sizes.

5.1. Real-world datasets: MNIST

Finally, to investigate the degree to which our findings extend to real world datasets, we trained nonlinear networks on a binary digit recognition task using the MNIST dataset (LeCun, Cortes, & Burges, 1998). We trained networks to discriminate between $P = 1024$ images of 7s and 9s, following the setting of

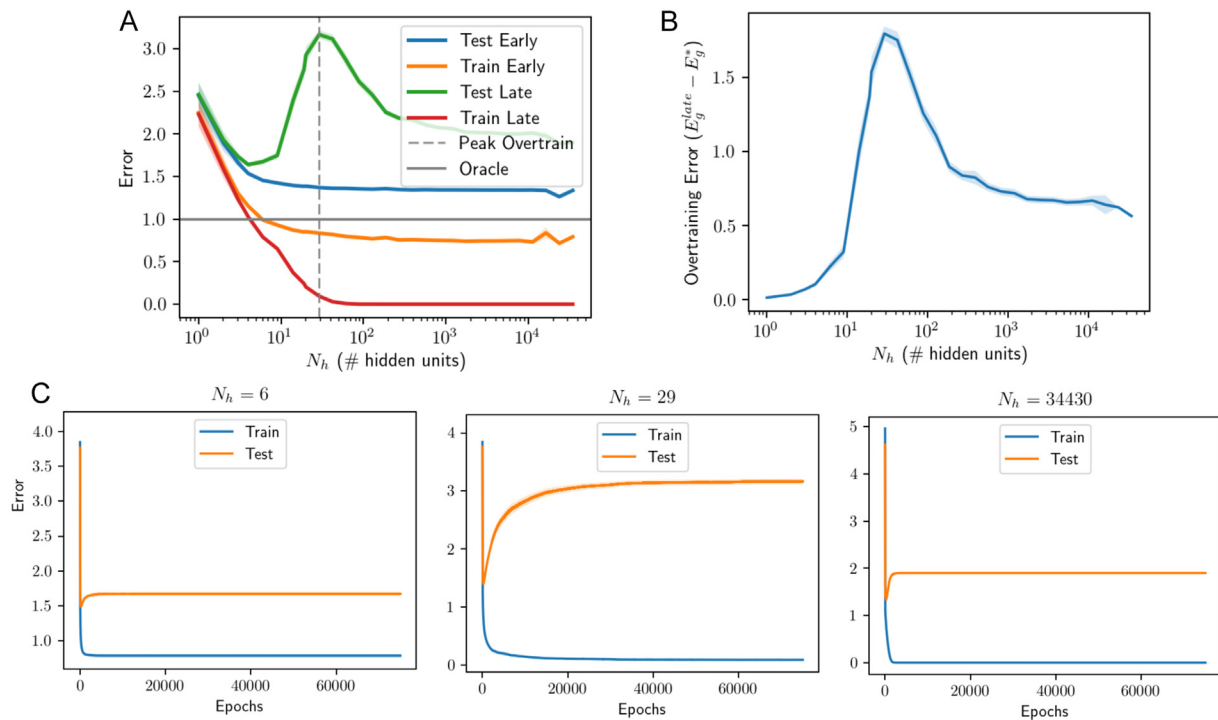


Fig. 12. Training both layers of nonlinear student. These simulations are of the same setting as that of Fig. 7, but with both layers of the student network trained. The number of hidden units for which the total number of free parameters is equal to the number of samples ($N_h + N_l N_h = P$) is marked with a dashed line and aligns well with the peak in overtraining observed. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

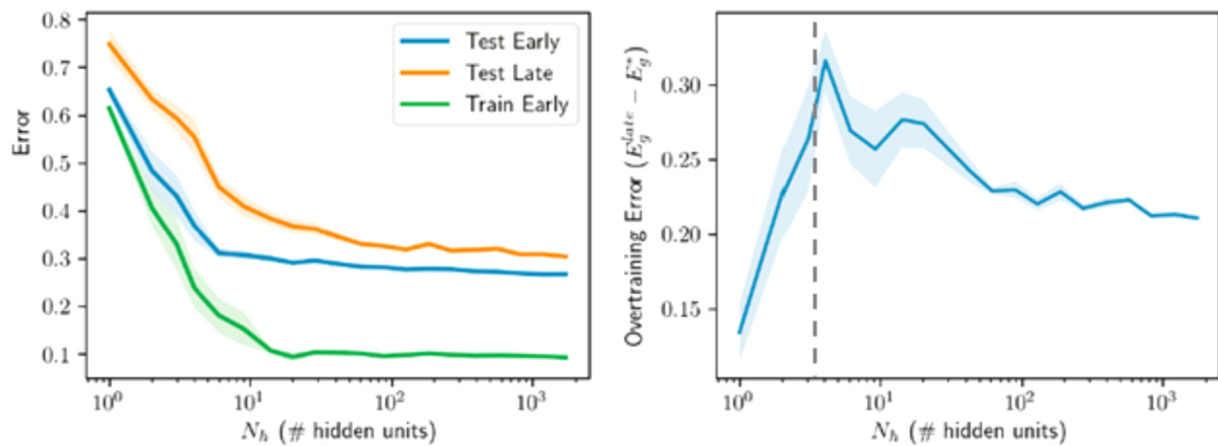


Fig. 13. Two layer network trained (without frozen weights) on binary MNIST classification of 7s vs. 9s with 1024 training samples. The qualitative trends identified earlier hold: over-training is a transient phenomenon at intermediate levels of complexity, and large models work well: no over-fitting is observed given optimal early stopping. Note that the over-training is less distinct than before primarily because these experiments were run for only 5000 epochs as opposed to tens of thousands of epochs as in Figs. 7 and 12, due to computational constraints.

Louart, Liao, and Couillet (2017). Each input consists of a 28×28 -pixel gray-scale image, flattened into a vector of length $N_i = 784$. Inputs were scaled to lie between 0 and 1 by dividing by 255 and each element was shifted to have zero mean across the training dataset. The target output was ± 1 depending on the class of the input, and the loss function remained the mean squared error on the training dataset. We train models using batch gradient descent with a single hidden layer for a fixed first layer and with both layers trained. Fig. 13 shows the resulting train and test errors for different model sizes and as a function of early

stopping. Again, all qualitative features appear to hold: overtraining peaks when the number of effective parameters matches the number of samples, and larger models perform better overall. We note that in these experiments, due to computational restrictions, training was continued for only 5000 epochs of batch gradient descent (c.f. 75 – 150k epochs for the other experiments reported in this paper), yielding less pronounced overtraining peaks; and the maximum model size was 1740, making the peak in overtraining less identifiable, particularly for the fixed first layer case. There is additionally no evidence of overfitting, as

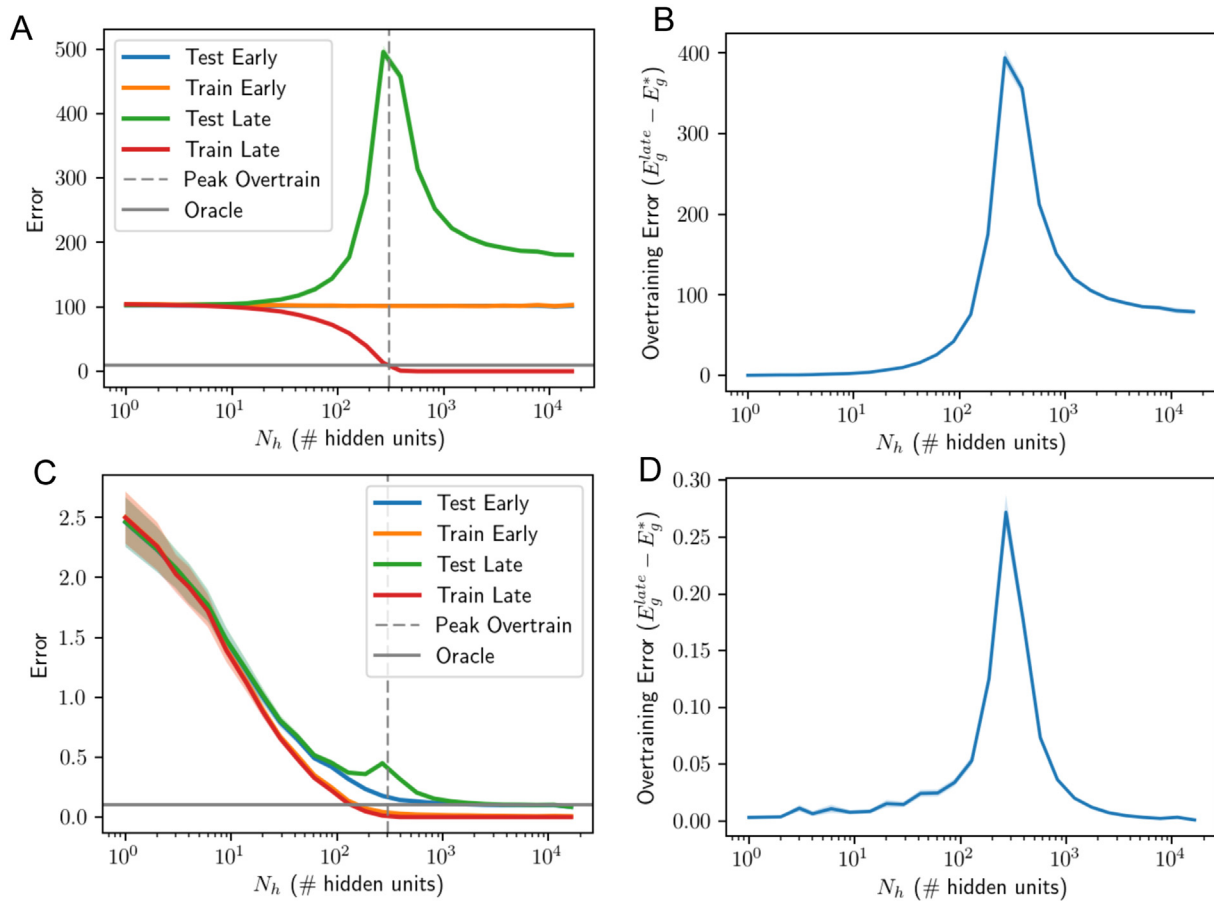


Fig. 14. Memorization, generalization, and SNR. Nonlinear ReLU networks with fixed first layer (same setting as Fig. 7) are trained on target labels of varying SNR. (A-B) Nearly random target labels (SNR = 0.01). In the high dimensional regime in which the model size outstrips the number of examples (here $N_h > 300$, indicated by vertical dotted line), all networks easily attain zero training error when trained long enough, thereby showing the ability to memorize an arbitrary dataset. (C-D) These exact same model sizes, however, generalize well when applied to data with nearly noise-free labels (SNR = 10). In fact in the low noise regime, even early stopping becomes unnecessary for large enough model sizes. The dynamics of gradient descent naturally protect generalization performance provided the network is initialized with small weights, because directions with no information have no dynamics. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the largest models outperformed smaller models. Overall, the qualitative findings accord well with those from the student–teacher paradigms, suggesting that our analysis accesses general behavior of generalization dynamics in gradient-trained neural networks.

6. Double descent, interpolation, memorization, and generalization

In the interpolation regime where training error can be driven to zero, influential work by Belkin, Hsu, Ma et al. (2019) and Belkin, Hsu and Xu (2019) has shown that very large models can achieve better test error than smaller models. We find similar behavior in our Fig. 9, for a setting with little label noise. However, it is important to note that larger models do not always perform better after extensive training in our setting. Fig. 14 shows test error as a function of model size for two SNR levels. While we generically observe a double descent curve in which late test error is non-monotonic in hidden layer size (Belkin, Hsu, Ma et al., 2019), whether a smaller or larger model will achieve lower test error depends on the SNR. In the high label noise regime (Fig. 14A–B), smaller models perform better; while in the low noise regime (Fig. 14C–D), overparametrized models perform better. To understand the source of this difference, we turn to the error decomposition shown in Fig. 10: as noise increases, approximation error increases in small and large models because there

is no way to exactly implement the non-deterministic teacher. However, the larger network has a larger error contribution from the error-fitting term, as shown in Fig. 10B (gold). Hence in the high noise regime, overparametrized models do not generalize as well as smaller models after extensive training.

To generalize well, a model must identify the underlying rule behind a dataset rather than simply memorizing each training example in its particulars. An empirical approach to test for memorization is to see if a deep neural network can fit a training set with randomly scrambled labels rather than the true labels (Zhang et al., 2017). If a network can fit arbitrary random labels, it would seem to be able to memorize any arbitrary training dataset, and therefore, have excessive capacity and poor generalization performance. This approach has been taken by Zhang et al. (2017), and extended by Arpit et al. (2017). Their main empirical findings conflict with this intuitive story: while large deep networks do readily achieve zero training error on a dataset that has been randomly labeled, they nevertheless generalize well when trained on the true labels, even without any regularization or early stopping. Our results provide a straightforward explanation of these phenomena in terms of the signal-to-noise ratio of the dataset and the high dimensional dynamics of gradient descent.

In our student–teacher setting, a randomly-labeled training set corresponds to a situation where there is no rule linking input to output, and is realized in the regime where $\text{SNR} \rightarrow 0$. As shown

in Fig. 14A (red curve), nonlinear networks with more parameters than samples can easily attain zero training error on this pure-noise dataset. However, they do not generalize well after substantial training, or even after optimal early stopping, because the model has fit pure noise. When the exact same networks are instead trained on a high SNR dataset (Fig. 14C), they generalize extremely well, nearly saturating the oracle lower bound. Moreover, in the high-SNR regime, the performance gain from early stopping nearly disappears for larger network sizes. To take a specific example, the largest network size considered ($N_h = 16\,326$) can easily fit noise labels in the low-SNR regime, but attains the best generalization performance of all models in the high-SNR regime, even without any early stopping. This behavior arises due to two important phenomena that arise in the high-dimensional setting: First, in overcomplete networks, gradient descent dynamics do not move in the subspace where no data lies. This frozen subspace implicitly regularizes the complexity of the network when networks are initialized with small norm weights. Regardless of network size, only a P -dimensional subspace remains active in the learning dynamics. Second, as overcompleteness increases, the eigengap (smallest nonzero eigenvalue) of the hidden layer correlation matrix also increases due to the nature of the Marchenko–Pastur distribution, which protects against overtraining even at long training times. This increasing eigengap is a fundamental property of the high dimensional setting. Thus large neural networks can generalize well even without early stopping, provided that the rule to be learned in the dataset is prominent, the network is large, and the network is initialized with small weights. We provide more details on this finding through the lens of Rademacher complexity in the following section.

7. Rademacher complexity and avoiding overfitting in non-linear neural networks

We want to understand why traditional Rademacher complexity bounds do not qualitatively match the overtraining picture observed in neural networks. In particular, they do not seem to square with the excellent generalization performance that can be achieved by large neural networks. To this end, we sketch how the dynamics of gradient descent learning may be included in the Rademacher complexity, yielding a bound which is more intuitively useful for understanding overtraining in neural networks. One particularly nice property of the bound we derive is that it shows how generalization can occur without early stopping: the Rademacher complexity can be expected to decrease with the number of hidden units N_h for $N_h > P$ even when training is continued for long times.

As noted by Zhang et al. (2017), on the surface the excellent generalization ability of large networks seems to contradict the intuition from traditional measures such as Rademacher complexity and VC dimension. We find that a straightforward application of these measures yields not only loose bounds, but even the opposite qualitative behavior from what is observed in simulations.

A simple illustration of this can be seen in Fig. 7C where the task is fitting a random non-linear teacher. Here the size of the student network is increased well beyond the number of samples in the dataset and the number of hidden units in the teacher network, yet the generalization performance of early stopping continues to improve. The Rademacher complexity and VC dimension of the network is growing with the number of hidden units due to the fact that larger neural networks can more easily fit randomly labeled data, as discussed in Zhang et al. (2017). If we consider training a large non-linear neural network with a random first layer, the empirical Rademacher complexity of a

classification problem (see e.g. Mohri, Rostamizadeh, & Talwalkar, 2012) is

$$\mathcal{R}(H) = \left\langle \sup_{h \in H} \frac{1}{P} \sum_{\mu=1}^P \sigma^\mu h(x^\mu) \right\rangle_{\sigma, x}, \tag{43}$$

where H is the class of functions the network or classifier can have (mapping input x^μ to an output), and $\sigma^\mu = \pm 1$ are randomly chosen labels. The Rademacher complexity measures the ability of a network to learn random labels, and leads to a bound on the gap between the probability of correct classification of a new example or training example, denoted as generalization error and training error respectively below. The bound states that with probability at least $1 - \delta$,

$$E_g - E_t \leq 2\mathcal{R}(H) + \sqrt{\frac{\log \frac{1}{\delta}}{2P}}. \tag{44}$$

As we increase the width of a large random network, we see empirically that the number of points which may be fit exactly grows with the size of the network. Thus the bound above becomes trivial when $N_h > P$ since $\mathcal{R}(H) \geq O(1)$ in this limit and any random classification can be realized because solving for the weights in the final layer requires solving for N_h unknowns with P random constraint equations. This argument shows that the bound is loose, but this is to be expected from a worst-case analysis. More surprising is the difference in qualitative behavior: larger networks in fact do better than intermediate size networks at avoiding overtraining in our simulations.

However, we will see that a more careful bounding of the Rademacher complexity which uses the learning dynamics of wide networks does allow us to capture the overtraining behavior we observe in practice. For a rectified linear network with one hidden layer, a bound on the Rademacher complexity (Bartlett & Mendelson, 2002) is

$$\mathcal{R}(H) \leq \frac{B_2 B_1 C \sqrt{N_h}}{\sqrt{P}}, \tag{45}$$

where B_2 is the norm of the output weights, B_1 is the maximum L2 norm of the input weights to any hidden neuron, and C is the maximum norm of the input data. From this it is clear that, no matter how many hidden neurons there are, the hypothesis complexity can be controlled by reducing the weight norm of the second layer.

Both early stopping and L2 regularization are capable of bounding the second layer weights and thus reducing overfitting, as we observe in simulations. In the case of early stopping, the dynamics scan through successively more expressive models with larger norms, and early stopping is used to pick an appropriate complexity.

Yet remarkably, even in the absence of early stopping we find that our analytic solutions provide a bound on the second layer weights in terms of the eigenspectrum of the hidden layer. The essential intuition is that, first, when a model is massively overcomplete, there is a large *frozen subspace* in which no learning occurs because no training samples lie in these directions; and second, the eigengap between the minimum eigenvalue and zero increases, protecting against overtraining and limiting the overall growth of the weights. To derive new bounds on the Rademacher complexity that account for these facts, we consider analytically how the weight norm depends on training time in our formulation. In the non-linear, fixed random first layer setting we are considering a model where

$$\hat{y}(x) = \sum_{a=1}^{N_h} \frac{w_a}{\sqrt{N_h}} \phi_a(x). \tag{46}$$

We will abuse notation because of the similarity with the single layer case, letting $X_h^{a\mu} = \frac{1}{\sqrt{N_h}}\phi_a(W_a^1 \cdot x^\mu)$. If we then diagonalize the covariance of the hidden layer s.t. $X_h(X_h)^T = V_h\Lambda_hV_h^T$ and solve for the dynamics of the weights, we find:

$$z_i^h(t) = \frac{\tilde{z}_i^h}{\lambda_i^h}(1 - e^{-\frac{\lambda_i^h t}{\tau}}) + z_i^h(0)e^{-\frac{\lambda_i^h t}{\tau}}. \quad (47)$$

Here we define the vectors: $z^h = wV_h^T$, $\tilde{z}^h = y(X_h)^TV_h$, and $z_i^h(0) = w(0)V_h^T$, and we denote the eigenvalues of the hidden layer covariance as $\lambda_1^h, \dots, \lambda_{N_h}^h$. Thus, the average squared norm of the output layer weights is

$$\begin{aligned} \langle \|w(t)\|^2 \rangle &= \sum_i z_i^2(t) \\ &= \sum_i \left(\frac{\|\tilde{z}_i^h\|^2}{(\lambda_i^h)^2} (1 - e^{-\frac{\lambda_i^h t}{\tau}})^2 + \|z_i^h(0)\|^2 e^{-2\frac{\lambda_i^h t}{\tau}} \right). \end{aligned} \quad (48)$$

In the case of a linear shallow network, without a hidden layer, the expression for the growth of the norm of weights is very similar, we simply substitute $\tilde{z}_i^h = z_i^h + \frac{\tilde{\epsilon}_i}{\sqrt{\lambda_i^h}}$, and in this case the distribution of eigenvalues approach the Marchenko–Pastur distribution in the high dimensional limit.

We now make the assumption that the initial weights are zero ($\sigma_w^0 = 0$), since this is the setting which will minimize the error and our large scale simulations are close to this limit (note that in deep networks we cannot set the initial weights exactly to zero because it would freeze the learning dynamics). It follows that $\langle \|w(t)\|^2 \rangle$ will monotonically increase as a function of time implying the Rademacher complexity bound is increasing with time.

The smallest non-zero eigenvalue constrains the maximum size that the norm of the weights can achieve, and zero eigenvalues result in no dynamics and thus do not impact the norm of the weights. Thus, even without early stopping, we can bound the norm of the weights in the hidden layer by an upper bound on the late-time behavior of (48) in the large P, N_h limit:

$$\frac{\|w\|}{\sqrt{N_h}} \leq \sqrt{\frac{\max_i \|\tilde{z}_i^h\|^2 \min(P, N_h)}{\min_{i, \lambda_i^h > 0} (\lambda_i^h)^2 N_h}} = B_2. \quad (49)$$

The minimum over P and N_h arises because there will be no impact on learning from eigenvalues of strength zero, corresponding to the frozen subspace. The presence of the minimum eigenvalue in the denominator indicates that, as the eigengap grows, the bound will improve. Substituting this into (45) yields a bound on the Rademacher complexity in terms of the eigenspectrum of the hidden layer:

$$\mathcal{R}(H) \leq B_1 C \sqrt{\frac{\max_i \|\tilde{z}_i^h\|^2 \min(P, N_h)}{\min_{i, \lambda_i^h > 0} (\lambda_i^h)^2 P}}. \quad (50)$$

The above bound qualitatively matches our simulations with a non-linear network and fixed first layer weights, which show that the gap between training and generalization error at late stopping times drops as we increase the number of network parameters beyond the number of examples (see Fig. 7C-D). In the equations above, as we increase N_h above P , $\min(P, N_h)$ remains fixed at P , but we do increase the minimum non-zero eigenvalue of the hidden layer covariance matrix (see e.g. Fig. 7E). This gap in the eigenspectrum can thus reduce the Rademacher complexity by bounding attainable network weights in massively overcomplete networks.

8. Discussion

Our findings show that the dynamics of gradient descent learning in the high-dimensional regime conspire to yield acceptable generalization performance in spite of large model sizes. Making networks very large, even when they have more free parameters than the number of samples in a dataset, can reduce overtraining because many of the directions of the network have zero gradient and thus are never learned. This frozen subspace protects against overtraining regardless of whether learning is stopped early. The worst setting for overtraining is when the network width matches the number of samples in shallow networks and when the number of parameters matches the number of samples in nonlinear random networks. Thus, our analysis of the learning dynamics helps to explain why overtraining is not a severe problem in very large networks.

Additionally, we have shown that making a non-linear two layer network very large can continuously improve generalization performance both when the first layer is random and even when it is trained, despite the high Rademacher complexity of a deep network (Zhang et al., 2017). We demonstrate this effect both on learning from a random teacher and on an MNIST classification task.

Our findings result from random matrix theory predictions of a greater abundance of small eigenvalues when the number of samples matches the number of parameters. In the under- or over-complete regimes, learning is well-behaved due to the gap between the origin and the smallest non-zero eigenvalue of the input correlation matrix. In our analysis, we have employed a simplified setting with Gaussian assumptions on the inputs, true parameters, and noise. However, the Gaussian assumption on parameters and noise is not essential. All that is required is that they are sampled *i.i.d.* with a finite mean and variance. In terms of other input distributions, we have shown empirically that our results apply for the MNIST dataset, but more broadly, the Marchenko–Pastur distribution is a universal limit for the eigenvalues of random matrices containing *i.i.d.* elements drawn from non-Gaussian distributions with sub-exponential tails (see e.g. Pillai, Yin, et al., 2014). Thus, the predictions are theoretically justified when the noise, parameters, and input are selected *i.i.d.* from non-heavy tailed distributions. We note that the exact extent of this universality is a current topic of research, and in particular, input distributions with strong correlations can deviate substantially.

Finally, our analysis points to several factors under the control of practitioners that impact generalization performance. To make recommendations for practice, it is necessary to understand the regime in which high-performing deep networks typically operate. We suggest that the relevant regime is the high-dimensional, high-SNR setting ($\alpha < 1, \text{SNR} \gg 1$). In this regime, very large networks have dramatic advantages: they generalize better than their smaller counterparts, even without any regularization or early stopping (c.f. Fig. 14). Consider, for instance, our results on the MNIST dataset presented in Fig. 13. Here, while early stopping could improve performance, even at long training times the best model was the largest tested. For this setting, the practical message emerging from our theory is that larger models have no downside from a generalization perspective, provided they are initialized with small initial weights. Our results point to a strong impact of initialization on generalization error in the limited-data and large network regime: starting with small weights is essential to good generalization performance, regardless of whether a network is trained using early stopping.

Except for the deep linear network reduction, our results have focused on minimally deep networks with at most one hidden layer. It remains to be seen how these findings might generalize

to deeper nonlinear networks (Kadmon & Sompolinsky, 2016), and if the requirement for good generalization (small weights) conflicts with the requirement for fast training speeds (large weights, Saxe et al., 2014) in very deep networks.

Since the original pre-print version of this paper was released, recent works have considered similar ideas (see Bahri et al., 2020 for a review). In particular, Spigler et al. (2018) trained deeper networks with a hinge loss and observed a similar empirical finding to the one observed in this paper that over-training is most problematic when the number of examples is near the number of effective parameters in a neural network. There has also been further investigation of the ‘double-descent’ phenomena (Belkin, Hsu, Ma et al., 2019; Belkin, Hsu and Xu, 2019), including in modern state of the art deep learning models (Nakkiran et al., 2020), and a growing number of papers have provided settings where double descent can be analyzed precisely (Ba, Erdogdu, Suzuki, Wu, & Zhang, 2019; Bartlett, Long, Lugosi, & Tsigler, 2020; Belkin, Hsu and Xu, 2019; Hastie, Montanari, Rosset, & Tibshirani, 2019). For instance, Hastie et al. (2019) statistically analyze ridgeless regression (corresponding to infinitely late time in our analysis) in nonlinear settings. Going beyond learning readout weights from fixed nonlinear transforms, Ba et al. (2019) investigate asymptotic performance in a student–teacher setting where only the first layer weight matrices are trained, for different initialization regimes. Other studies have used tools from random matrix theory to investigate more complex input statistics than the simple Gaussian assumption considered here, including the impact of clustered inputs (Louart et al., 2017), and features arising from random nonlinear transformations (Pennington & Worah, 2017).

The importance of initialization in controlling inductive biases in deep networks has been highlighted by results showing that infinitely wide networks can be trained in an ‘‘active’’ or ‘‘lazy’’ regime (Chizat, Oyallon, & Bach, 2018). In the lazy regime corresponding to relatively large initializations, the dynamics of deep nonlinear networks evolve as a linear system with features determined by the Neural Tangent Kernel (Arora, Du, Hu, Li, & Wang, 2019; Jacot, Gabriel, & Hongler, 2018; Lee et al., 2019), showing that the frozen subspace idea extends straightforwardly to this setting. However in the active regime, the dynamics remain nonlinear and can differ dramatically from this linearized behavior, often attaining better performance (Chizat et al., 2018; Mei, Montanari, & Nguyen, 2018), and presenting a continuing challenge for theoretically understanding generalization properties in deep neural networks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We would like to thank Kenyon Tsai, James Fitzgerald, and Pankaj Metha for many useful discussions. M.A. and A.M.S. warmly thank the Swartz Program in Theoretical Neuroscience at Harvard University for support. A.M.S. is supported by a Sir Henry Dale Fellowship from the Wellcome Trust, United Kingdom and Royal Society, United Kingdom (Grant Number 216386/Z/19/Z). H.S. is partially supported by the Gatsby Charitable Foundation, United Kingdom.

Appendix A. Deep linear neural network dynamics

Here we derive the reduction describing the dynamics of learning in deep linear neural networks. To begin, we make the following change of variables: $W_l(t) = r_l z(t) V^T$, where $z(t) \in R^{1 \times N_l}$ is a vector encoding the time-varying overlap with each principle axis in the input (recall $\Sigma^{xx} = V \Lambda V^T$); and $W_l(t) = d(t) r_{l+1} r_l^T$, $l = 2, \dots, D$ where the vectors $r_i \in R^{N_i \times 1}$ are arbitrary unit norm vectors ($r_i^T r_i = 1$) specifying freedom in the internal representation of the network, r_{D+1} is defined to be one, and $d(t)$ is a scalar encoding the change in representation over time. With these definitions, $W^{\text{tot}} = \left(\prod_{l=2}^D d(t) r_{l+1} r_l^T \right) r_2 z(t) V^T = d(t)^{D-1} z(t) V^T = u(t) z(t) V^T$ where we have defined the scalar $u(t) = d(t)^{D-1}$.

We thus have, for $l > 1$,

$$\tau \frac{d}{dt} W_l = \left(\prod_{i=l+1}^D W_i \right)^T \left[\Sigma^{yx} - \left(\prod_{i=1}^D W_i \right) \Sigma^{xx} \right] \times \left(\prod_{i=1}^{l-1} W_i \right)^T, \quad (51)$$

$$\tau \frac{d}{dt} d(t) r_{l+1} r_l^T = r_{l+1} d(t)^{D-l} \left[\Sigma^{yx} - d(t)^{D-1} z(t) V^T V \Lambda V^T \right] \times V d(t)^{l-2} z(t)^T r_l^T \quad (52)$$

$$\tau \frac{d}{dt} d(t) r_{l+1} r_l^T = d(t)^{D-2} \left[\tilde{s} z^T - d(t)^{D-1} z(t) \Lambda z(t)^T \right] r_{l+1} r_l^T \quad (53)$$

and equating coefficients yields the dynamics

$$\tau \frac{d}{dt} d(t) = d(t)^{D-2} \left[\tilde{s} z^T - d(t)^{D-1} z(t) \Lambda z(t)^T \right] \quad (54)$$

as desired. We then change variables to $u(t) = d(t)^{D-1}$ to derive the first differential equation:

$$\tau \dot{u} = (D-1) u^{\frac{2D-4}{D-1}} (\tilde{s} z^T - u z \Lambda z^T). \quad (55)$$

Similarly for $l = 1$, we have

$$\tau \frac{d}{dt} W_1 = \left(\prod_{i=2}^D W_i \right)^T \left[\Sigma^{yx} - \left(\prod_{i=1}^D W_i \right) \Sigma^{xx} \right], \quad (56)$$

$$\tau \frac{d}{dt} r_2 z(t) V^T = r_2 d(t)^{D-1} \left[\Sigma^{yx} - d(t)^{D-1} z(t) V^T V \Lambda V^T \right], \quad (57)$$

$$\tau \frac{d}{dt} r_2 z(t) V^T = d(t)^{D-1} \left[\tilde{s} - d(t)^{D-1} z(t) \Lambda \right] V^T. \quad (58)$$

Again equating coefficients and making the substitution $u(t) = d(t)^{D-1}$ yields

$$\tau \dot{z} = u (\tilde{s} - u z \Lambda). \quad (59)$$

Appendix B. Optimal weight norm growth with SNR

Here we derive the optimal weight norm as a function of the signal to noise ratio ((19) in the main text). We analyze ridge regularized regression with strength $\gamma = \frac{1}{\text{SNR}}$,

$$\hat{w} = y X^T \left(X X^T + \frac{1}{\text{SNR}} I \right)^\dagger. \quad (60)$$

Substituting from the model that

$$y = \bar{w} X + \epsilon, \quad (61)$$

yields:

$$\hat{w} = (\bar{w} X X^T + \epsilon X^T) \left(X X^T + \frac{1}{\text{SNR}} I \right)^\dagger. \quad (62)$$

Utilizing the decompositions

$$XX^T = V\Lambda V^T, \quad (63)$$

$$X = V\Lambda^{1/2}U^T, \quad (64)$$

the previous equation can be written as

$$\hat{w}V = (\hat{w}V\Lambda + \epsilon U\sqrt{\Lambda})(\Lambda + \gamma I)^{-1}. \quad (65)$$

Because the noise and parameter values are drawn iid, the rotations U and V will not impact their mean or moments, we can square the above equations and compute the average parameter strength:

$$\langle \hat{w}^2 \rangle = \sigma_w^2 \frac{1}{N} \sum_{i=1}^N \frac{\lambda_i}{\lambda_i + \frac{1}{\text{SNR}}} \rightarrow \sigma_w^2 \int \rho^{\text{MP}}(\lambda) \left(\frac{\lambda}{\lambda + \frac{1}{\text{SNR}}} \right). \quad (66)$$

Thus, we see that the average estimated parameter norm monotonically increases with SNR which makes sense intuitively because there is less regularization required as the data quality improves.

Appendix C. Generalization error decomposition

We begin by defining w_{\parallel}^* to be the weights which minimize generalization error for a specific architecture given that the weights must be zero in the null space of the hidden layer covariance. By optimizing generalization error under this constraint it follows that:

$$w_{\parallel}^* = RC^{\dagger}P_{\parallel}. \quad (67)$$

We can then define approximation noise in the parallel subspace as:

$$\epsilon_{\parallel}(x) = y(x) - w_{\parallel}^* \phi(x). \quad (68)$$

It follows from the definition of w_{\parallel}^* that

$$\langle \epsilon_{\parallel}^a(x)(u \cdot \phi(x)) \rangle \quad \forall a, \quad (69)$$

where u is in not in the null space of \hat{C} . This fact will allow us to simplify cross terms in the expression for generalization error. We begin by decomposing the training data into a realizable component along with approximation noise:

$$Y = w_{\parallel}^* \Phi + \mathcal{E}, \quad (70)$$

where $Y_{a\mu} = y_a(x^{\mu})$, $\Phi_{j\mu} = \phi_j(x^{\mu})$, and $\mathcal{E}_{a\mu} = \epsilon_a(x^{\mu})$. From Eq. (70) and the expression for \hat{w} , we find:

$$\hat{w} = w_{\parallel}^* + E\Phi^T(\Phi\Phi^T)^{\dagger}. \quad (71)$$

The generalization error has the form:

$$E_g = \left\langle \sum_a \left(\sum_j \hat{w}_{aj} \phi_j(x) - y_a(x) \right)^2 \right\rangle_x \quad (72)$$

which we can write as:

$$E_g = \left\langle \sum_a \left(\sum_j (\hat{w}_{aj} - (w_{\parallel}^*)_{aj}) \phi_j(x) - \epsilon_{\parallel}^a(x) \right)^2 \right\rangle_x. \quad (73)$$

We can compose the average into 3 terms with the cross term going to zero because $\hat{w} - w_{\parallel}^*$ must lie in the parallel subspace for each a . It follows that:

$$E_g = (\hat{w} - w_{\parallel}^*) C (\hat{w} - w_{\parallel}^*)^T + \sum_a \langle \epsilon_{\parallel}^a(x)^2 \rangle. \quad (74)$$

Substituting the form of \hat{W} from Eq. (71) into the above equation yields:

$$E_g = \mathcal{E}\Phi^T(\Phi\Phi^T)^{\dagger}C(\Phi\Phi^T)^{\dagger}\Phi\mathcal{E}^T + \sum_a \langle \epsilon_{\parallel}^a(x)^2 \rangle. \quad (75)$$

Dropping the index a , from the form of ϵ and ϵ_{\parallel} for simplicity, we have that:

$$\langle (\epsilon_{\parallel}(x))^2 \rangle_x = \langle (y - w_{\parallel}^* \phi(x))^2 \rangle = \langle (y - RC^{\dagger}P_{\parallel}\phi(x))^2 \rangle \quad (76)$$

It follows that:

$$\langle (\epsilon_{\parallel}(x))^2 \rangle_x = \langle (y - RC^{\dagger}\phi(x) + RC^{\dagger}(I - P_{\parallel})\phi(x))^2 \rangle, \quad (77)$$

which (from the orthogonality of phi and epsilon) may be expanded as:

$$\langle (\epsilon_{\parallel}(x))^2 \rangle_x = \langle \epsilon(x)^2 \rangle_x + \langle \phi(x)^T(I - P_{\parallel})^T C^{\dagger} R^T RC^{\dagger}(I - P_{\parallel})\phi(x) \rangle. \quad (78)$$

It follows that:

$$\langle (\epsilon_{\parallel}(x))^2 \rangle_x = \langle \epsilon(x)^2 \rangle_x + \text{Tr} [RC^{\dagger}P_{\perp}CP_{\perp}^T C^{\dagger} R^T]. \quad (79)$$

The second term simply corresponds to the generalization error due to the optimal weights projected onto the orthogonal subspace. Note that the trace is not needed for scalar output settings. More generally if there are some initialization weights they will be added in with RC^{\dagger} to the second term above, which is the error from the frozen subspace approximation.

References

- Advani, M., & Ganguli, S. (2016a). An equivalence between high dimensional bayes optimal inference and m-estimation. *Advances in Neural Information Processing Systems*.
- Advani, M., & Ganguli, S. (2016b). Statistical mechanics of high dimensional inference supplementary material. In: See <https://ganguli-gang.stanford.edu/pdf/HighDimInf.Supp.pdf>.
- Advani, M., & Ganguli, S. (2016c). Statistical mechanics of optimal convex inference in high dimensions. *Physical Review X*, 6(3), Article 031034.
- Advani, M., Lahiri, S., & Ganguli, S. (2013). Statistical mechanics of complex neural systems and high dimensional data. *Journal of Statistical Mechanics: Theory and Experiment*, (03), P03014.
- Amari, S., Murata, N., & Müller, K. R. (1996). Statistical theory of overtraining - Is cross-validation asymptotically effective?. *Advances in Neural Information Processing Systems*.
- Amari, S., Murata, N., Müller, K. R., Finke, M., & Yang, H. (1995). Asymptotic statistical theory of overtraining and Cross Validation. *IEEE Transactions on Neural Networks*, 8(5), 985–996.
- Arora, S., Cohen, N., & Hazan, E. (2018). On the optimization of deep networks: Implicit acceleration by overparameterization. In *35th international conference on machine learning, ICML 2018, Vol. 1* (pp. 372–389).
- Arora, S., Du, S. S., Hu, W., Li, Z., & Wang, R. (2019). Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *36th international conference on machine learning* (pp. 477–502).
- Arpit, D., Jastrz, S., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., et al. (2017). A closer look at memorization in deep Networks. In *ICML*.
- Ba, J., Erdogdu, M. A., Suzuki, T., Wu, D., & Zhang, T. (2019). Generalization of two-layer Neural networks: an asymptotic Viewpoint. In *International conference on learning representations*. 2016.
- Bahri, Y., Kadmon, J., Pennington, J., Schoenholz, S. S., Sohl-Dickstein, J., & Ganguli, S. (2020). Statistical mechanics of deep learning. *Annual Review of Condensed Matter Physics*, 11(1), 501–528.
- Bai, Z., & Silverstein, J. W. (2010). *Spectral analysis of large dimensional random matrices*. Springer.
- Baldi, P., & Chauvin, Y. (1991). Temporal evolution of Generalization during Learning in Linear networks. *Neural Computation*, 3, 589–603.
- Baldi, P., Chauvin, Y., & Hornik, K. (1990). Supervised and unsupervised learning in Linear Networks. In *International neural network conference* (pp. 825–828).
- Baldi, P., & Hornik, K. (1989). Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1), 53–58.
- Baldi, P. F., & Hornik, K. (1995). Learning in linear neural networks: a survey. *IEEE Transactions on Neural Networks*, 6(4), 837–858.

- Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3), 930–945.
- Bartlett, P. L., Long, P. M., Lugosi, A., & Tsigler, G. (2020). Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*.
- Bartlett, P. L., & Mendelson, S. (2002). Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(3), 463–482.
- Belkin, M., Hsu, D., Ma, S., & Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32), 15849–15854.
- Belkin, M., Hsu, D., & Xu, J. (2019). Two models of double descent for weak features. arXiv preprint arXiv:1903.07571.
- Benaych-Georges, F., & Rao, R. (2011). The eigenvalues and eigenvectors of finite, low rank perturbations of large random matrices. *Advances in Mathematics*, 227(1), 494–521.
- Benaych-Georges, F., & Rao, R. (2012). The singular values and vectors of low rank perturbations of large rectangular random matrices. *Journal of Multivariate Analysis*, 111, 120–135.
- Biehl, M., Caticha, N., & Riegler, P. (2009). Statistical mechanics of on-line learning. In T. Villmann, M. Biehl, B. Hammer, & M. Verleysen (Eds.), *Similarity-based clustering: recent developments and biomedical applications*. Berlin Heidelberg: Springer.
- Canziani, A., Paszke, A., & Culurciello, E. (2017). An analysis of deep neural network models for practical applications. (pp. 1–7). In: arXiv.
- Caruana, R., Lawrence, S., & Giles, C. L. (2001). Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Advances in neural information processing systems* (pp. 402–408).
- Chauvin, Y. (1990). Generalization dynamics in LMS Trained linear networks. In *NIPS* (pp. 890–896).
- Chizat, L., Oyallon, E., & Bach, F. (2018). On lazy training in differentiable programming. In *NeurIPS*.
- Dodier, R. (1995). Geometry of early stopping in Linear Networks. In *Advances in neural information processing systems 8*.
- Du, S. S., Hu, W., & Lee, J. D. (2018). Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. *Advances in Neural Information Processing Systems*, (NeurIPS), 384–395.
- Dunmur, A. P., & Wallace, D. J. (1999). Learning and generalization in a linear perceptron stochastically trained with noisy data. *Journal of Physics A: Mathematical and General*, 26(21), 5767–5779.
- Engel, A., & Van Den Broeck, C. (2001). *Statistical mechanics of learning*. Cambridge: Cambridge University Press.
- Fukumizu, K. (1998). Effect of batch learning in multilayer neural networks. In *Proceedings of the 5th international conference on neural information processing* (pp. 67–70).
- Goodfellow, I. J., Vinyals, O., & Saxe, A. M. (2015). Qualitatively characterizing Neural Network Optimization problems. In *The international conference on learning representations*. San Diego, CA: Oral presentation.
- Hastie, T., Montanari, A., Rosset, S., & Tibshirani, R. J. (2019). Surprises in high-dimensional ridgeless least squares interpolation. arXiv preprint arXiv:1903.08560.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV* (pp. 1026–1034).
- Hoyle, D. C., & Rattay, M. (2007). Statistical mechanics of learning multiple orthogonal signals: Asymptotic theory and fluctuation effects. *Physical Review E*, 75.
- Jacot, A., Gabriel, F., & Hongler, C. (2018). Neural tangent kernel: Convergence and generalization in neural networks. *Advances in Neural Information Processing Systems*, (4), 8571–8580.
- Ji, Z., & Telgarsky, M. (2019). Gradient descent Aligns the Layers of deep Linear Networks. In *International conference on learning representations*.
- Kadmon, J., & Sompolinsky, H. (2016). Optimal architectures in a solvable model of deep networks. In *NIPS*.
- Kawaguchi, Kenji (2016). Deep learning without Poor Local Minima. In *NIPS*.
- Kinouchi, O., & Caticha, N. (1995). On-line versus off-line learning in the linear perceptron: A comparative study. *Physical Review E*, 52(3).
- Krogh, A., & Hertz, J. A. (1992). Generalization in a linear perceptron in the presence of noise. *Journal of Physics A: Mathematical and General*, 25, 1135–1147.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521, 436–444.
- LeCun, Y., Cortes, C., & Burges, C. J. C. (1998). The MNIST database of handwritten digits.
- LeCun, Y., Kanter, I., & Solla, S. A. (1991). Eigenvalues of covariance matrices: Application to neural-network learning. *Physical Review Letters*, 66(18), 2396.
- Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Novak, R., Sohl-Dickstein, J., et al. (2019). Wide Neural Networks of any Depth evolve as linear models under gradient descent. In *NeurIPS*.
- Louart, C., Liao, Z., & Couillet, R. (2017). A random matrix approach to neural networks. arXiv preprint arXiv:1702.05419.
- Marchenko, V. A., & Pastur, L. A. (1967). Distribution of eigenvalues for some sets of random matrices. *Matematicheskii Sbornik*, 114, 507–536.
- Mei, S., Montanari, A., & Nguyen, P. M. (2018). A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33), E7665–E7671.
- Mishkin, D., & Matas, J. (2016). All you need is a good init. In *ICLR* (pp. 1–13).
- Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012). *Foundations of machine learning*. MIT press.
- Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., & Sutskever, I. (2020). Deep double descent: Where Bigger Models and more data Hurt. In *International conference on learning representations*.
- Pennington, J., & Bahri, Y. (2017). Geometry of neural network loss surfaces via random matrix theory. In *International conference on machine learning* (pp. 2798–2806).
- Pennington, J., & Worah, P. (2017). Nonlinear random matrix theory for deep learning. In *Advances in neural information processing systems* (pp. 2637–2646).
- Pillai, Natesh S., Yin, Jun, et al. (2014). Universality of covariance matrices. *Annals of Applied Probability*, 24(3), 935–1001.
- Saad, D., & Solla, S. A. (1995). Exact solution for on-line learning in multilayer neural networks. *Physical Review Letters*, 74, 4337–4340.
- Saxe, A. M., McClelland, J. L., & Ganguli, S. (2014). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In Y. Bengio and Y. LeCun (Eds.), *The international conference on learning representations*. Banff, Canada.
- Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61, 85–117.
- Seung, H. S., Sompolinsky, H., & Tishby, N. (1992). Statistical mechanics of learning from examples. *Physical Review A*, 45(8), 6056–6091.
- Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *ICLR* (pp. 1–14).
- Soudry, D., & Hoffer, E. (2017). Exponentially vanishing sub-optimal local minima in multilayer neural networks. In: arXiv.
- Spigler, S., Geiger, M., d’Ascoli, S., Sagun, L., Biroli, G., & Wyart, M. (2018). A jamming transition from under- to over-parametrization affects loss landscape and generalization. arXiv preprint arXiv:1810.09665.
- Watkin, T. L. H., Rau, A., & Biehl, M. (1993). The statistical mechanics of learning a rule. *Reviews of Modern Physics*, 65(2), 499–556.
- Wei, H., & Amari, S. (2008). Dynamics of learning near singularities in radial basis function networks. *Neural Networks*, 21, 989–1005.
- Wei, H., Zhang, J., Cousseau, F., Ozeki, T., & Amari, S. (2008). Dynamics of learning near singularities in layered networks. *Neural Computation*, 20, 813–843.
- Yao, Y., Rosasco, L., & Caponnetto, A. (2007). On early stopping in Gradient Descent Learning. *Constructive Approximation*, 26, 289–315.
- Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *ICLR*.