Classifying high-dimensional Gaussian mixtures: Where kernel methods fail and neural networks succeed

Maria Refinetti [1 2] **Sebastian Goldt** [3] **Florent Krzakala** [2] **Lenka Zdeborová** [4]

Abstract

A recent series of theoretical works showed that the dynamics of neural networks with a certain initialisation are well-captured by kernel methods. Concurrent empirical work demonstrated that kernel methods can come close to the performance of neural networks on some image classification tasks.These results raise the question of whether neural networks only learn successfully if kernels also learn successfully, despite neural nets being more expressive. Here, we show theoretically that two-layer neural networks (2LNN) with only a few neurons can beat the performance of kernel learning on a simple Gaussian mixture classification task. We study the high-dimensional limit, i.e. when the number of samples is linearly proportional to the dimension, and show that while small 2LNN achieve near-optimal performance on this task, lazy training approaches such as random features and kernel methods do not.Our analysis is based on the derivation of a closed set of equations that track the learning dynamics of the 2LNN and thus allow to extract the asymptotic performance of the network as a function of signal-to-noise ratio and other hyperparameters. We finally illustrate how over-parametrising the neural network leads to faster convergence, but does not improve its final performance.

1. Introduction

Explaining the success of deep neural networks in many areas of machine learning remains a key challenge for learning theory. A series of recent theoretical works made progress towards this goal by proving trainability of two-layer neural networks (2LNN) with gradient-based methods (Jacot

[1]Laboratoire de Physique de l'École Normale Supérieure, Université PSL, CNRS, Sorbonne Université, Université Paris-Diderot, Sorbonne Paris Cité [2]IdePHICS Lab, EPFL [3]International School of Advanced Studies (SISSA), Trieste, Italy [4]SPOC Lab, EPFL. Correspondence to: <maria.refinetti@ens.fr>.

et al., 2018; Allen-Zhu et al., 2018; Li & Liang, 2018; Allen-Zhu et al., 2019; Cao & Gu, 2019; Du et al., 2019). These results are based on the observation that strongly over-parameterised 2LNN can achieve good performance even if their first-layer weights remain almost constant throughout training. This is the case if the initial weights are chosen with a particular scaling, which was dubbed the "lazy regime" by Chizat et al. (2019). Going a step further, simply fixing the first-layer weights of a 2LNN at their initial values yields the well-known random features model of Rahimi & Recht (2008; 2009), and can be seen as an approximation of kernel learning (Scholkopf & Smola, 2018). This behaviour is to be contrasted with the "feature learning regime", where the weights of the first layer move significantly during training. Recent empirical studies showed that on some benchmark data sets in computer vision, kernels derived from neural networks achieve comparable performance to neural networks (Matthews et al., 2018; Lee et al., 2018; Garriga-Alonso et al., 2019; Arora et al., 2019; Li et al., 2019; Shankar et al., 2020).

These results raise the question of whether neural networks only learn successfully if random features can also learn successfully, and have led to a renewed interest in the exact conditions under which neural networks trained with gradient descent achieve a better performance than random features (Bach, 2017; Yehudai & Shamir, 2019; Wei et al., 2019; Li et al., 2020; Daniely & Malach, 2020; Geiger et al., 2020; Paccolat et al., 2021; Suzuki & Akiyama, 2020). Chizat & Bach (2020) studied the implicit bias of wide two-layer networks trained on data with a low-dimensional structure. They derived strong generalisation bounds, which, when both layers of the network are trained, are independent of ambient dimensions, indicating that the network is able to adapt to the low dimensional structure. In contrast, when only the output layer of the network is trained, the network does not possess such an adaptivity, leading to worse performance. Ghorbani et al. (2019; 2020) analysed in detail how data structure breaks the curse of dimensionality in wide two-layer neural networks, but not in learning with random features, leading to better performance of the former.
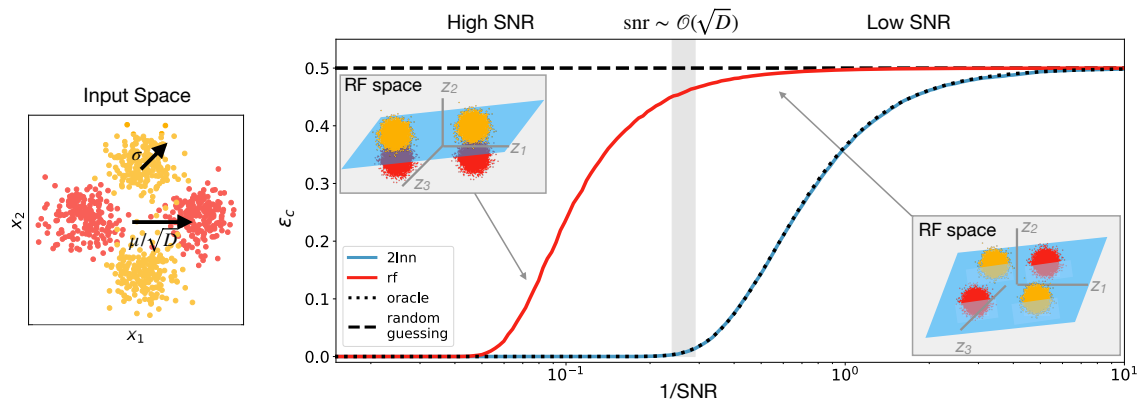
Figure 1. **Random Features and 2LNN on high-dimensional Gaussian mixture classification** *(Left)* Consider a data distribution that is a mixture of four Gaussians in D dimensions. The first two components of the centroids organized in a XOR-like manner as shown, while the other $D - 2$ directions of the centroids are set to zero. The signal-to-noise is $\mathsf{snr} = |\mu|/\sqrt{D}\sigma$. We used $\mu = \sqrt{D}$ so the snr is effectively given by the inverse of the width of the Gaussian: $\mathsf{snr} = 1/\sigma$. *(Right)* A two-layer neural network with $K = 4$ hidden neurons and ReLU non-linearity trained using stochastic gradient descent, achieves a long-time test error close to the optimal (oracle) error (3) for the whole range of snr. The test error is obtained analytically using techniques described in Sec. (2.3). In sharp contrast, random features (RF), whose performance is given by Eq. (16), require a high snr to perform as well as the oracle. They performs better than chance when $\mathsf{snr} \gg \sqrt{D}/\min(P, N)^{1/4}$, and thus requires a diverging snr in the high-dimensional limit. The insets show the mixture *after* applying random features; only at high snr does the mixture become linearly separable. *Parameters:* RF error is computed with $D = 10000$ and $P = 2D$ and the 2LNN's with $D = 1000$. For both methods $\eta = 0.1$, $|\mu|/\sqrt{D} = 1$.

1.1. Main contributions

We show that even a two-layer neural network with *only a few hidden neurons* outperforms kernel methods on the classical problem of Gaussian mixture classification. We give a sharp asymptotic analysis of 2LNN and random features on Gaussian mixture classification in the high-dimensional regime where the number of samples N is linearly proportional to the input dimension $D \to \infty$. More precisely:

(i) We analyse 2LNN by deriving a closed set of ordinary differential equations (ODEs), which track the test error of 2LNN with a few hidden neurons $K \sim O(1)$ trained using one-pass (or online) SGD on Gaussian mixture classification. We thereby extend the classical ODE analysis of Riegler & Biehl (1995); Saad & Solla (1995a) where the label $y(x)$ is a function of the input x, to a setup where the input is conditional on the label. Solving these equations for their asymptotic fixed point, i.e. taking $t \to \infty$ *after* $D \to \infty$, yields the final classification error of the 2LNN.

(ii) Keeping in mind the high-dimensional limit where the number of samples N is proportional to D, We analyse how Gaussian mixtures are transformed under P random features in the regime $P, D \to \infty$ with $\gamma \equiv P/D$ fixed. In the high-dimensional limit the performance at large γ converges to the one of the corresponding kernel (Rahimi & Recht, 2008; 2009; El Karoui, 2010; Pennington & Worah, 2017; Louart et al., 2018; Liao & Couillet, 2018), and we can thus recover

the performance of kernel learning by taking γ large enough.

(iii) We compute the asymptotic generalisation of random features on mixtures of Gaussians, which allows us to compare their performance to the performance of 2LNN for various signal-to-noise ratios.

While we do not attempt formal rigorous derivations, to keep the paper readable, our theoretical claims are however amenable to rigorous theorems. In particular the ODEs analysis could be formalised rigorously using the technique of (Wang et al., 2019; Goldt et al., 2019). Our results are valid for generic Gaussian Mixtures with $O(1)$ clusters and we focus on the particular example of the XOR-like mixture in order to make the problematic clear.

1.2. A paradigmatic example

Our results can be illustrated with a data distribution where inputs are distributed in a mixture of four Gaussians, whose centres form a XOR-function, and thus cannot be linearly separated in direct space. We find that neural networks with a few neurons have no problems learning a good partitioning of the space in this situation, reaching oracle-like performance in the process. Kernel methods, however, manage to do so only if the centres are extremely well separated, and completely fail when they are too close. This is illustrated in Fig. 1: Inputs $x = (x_r) \in \mathbb{R}^D$ are drawn from the Gaussian mixture shown on the left, where inputs in red, yellow have

labels $y = 1, -1$, respectively. The first two components of the means are organised as in the diagram, while the remaining $D - 2$ components are zero, yielding a XOR-like pattern. Each Gaussian cloud has standard deviation $\sigma \mathbb{I}_D$, as Gaussian noise is added to all components of the input, resulting in an snr of $|\mu|/\sqrt{D\sigma^2}$.

We compare the performance of a **two-layer neural network** ϕ_θ with parameters $\theta = (K, v, W, g)$,

$$\phi_\theta(x) = \sum_{k=1}^{K} v^k g\left(\lambda^k\right), \quad \lambda^k \equiv \frac{1}{\sqrt{D}} \sum_{r=1}^{D} w_r^k x_r, \quad (1)$$

where $v = (v^k) \in \mathbb{R}^K$ and $W = (w_r^k) \in \mathbb{R}^{K \times D}$ are the weights of the network and $g : \mathbb{R} \to \mathbb{R}$ is a non-linear function shown above only has $K = 4$ neurons, and we keep K of order 1 compared to the input dimension $D \to \infty$ throughout this paper. We train the 2LNN using online stochastic gradient descent, where at each step of the algorithm, we draw a new sample from the mixture. We study the high-dimensional limit $t \equiv N/D = O(1)$, and obtain the final performance of the 2LNN in the limit $t \to \infty$ (after $D \to \infty$) of the ODEs we derive in Sec. 2.2. In blue, we plot the final classification error

$$\epsilon_c(\theta) = \mathbb{E} \, \Theta\left[-y\phi_\theta(x)\right], \quad (2)$$

where the expectation $\mathbb{E}$ is computed over the Gaussian mixture for a network with fixed parameters θ and Θ is the Heaviside step function. The classification error of the 2LNN is very close to that of an *oracle* with knowledge of the means of the mixture that assigns to each input the label of the nearest mean, achieving a classification error of

$$\epsilon_c^{\text{oracle}} = 1/2 \left(1 - \text{erf}\left(|\mu|/2\sigma\sqrt{D}\right)^2\right). \quad (3)$$

We compare the performance of the 2LNN to the performance of **random features** (Rahimi & Recht, 2008; 2009), where we first project the inputs x to a higher-dimensional feature space, where features $z = (z_i) \in \mathbb{R}^P$ are given by

$$z_i = \psi(u_i), \qquad u_i \equiv \sum_{r=1}^{D} \frac{1}{\sqrt{D}} F_{ir} x_r, \quad (4)$$

where $F \in \mathbb{R}^{P \times D}$ is a random, but fixed projection matrix and $\psi : \mathbb{R} \to \mathbb{R}$ is an element-wise non-linearity. The features are then fit by training a linear model,

$$\phi_\theta(z) = \frac{1}{\sqrt{P}} \sum_{i=1}^{P} w_i z_i, \quad (5)$$

with weights $w \in \mathbb{R}^P$, using SGD, where we again draw a fresh sample from the mixture to evaluate the gradients at each step. The performance of RF is shown in red in

Fig. 1. While RF achieve low classification error at high snr, there is a wide range of snr where random features do significantly worse than the 2LNN. The insets give the intuition behind this result: at high snr, RF map the inputs into linearly separable mixture in random feature space (left) while at lower snr, the transformed mixture is not linearly separable in RF space (right), leading to poor performance.

We emphasise that we study random features in the *high-dimensional limit* where we let $N, D \to \infty$ with their ratio $t = N/D \sim O(1)$ as before, while also letting the number of random features $P \to \infty$ with their ratio $\gamma \equiv P/D \sim O(1)$ fixed. This regime has been studied in a series of recent works (Lelarge & Miolane, 2019; Couillet, 2019; Liao & Couillet, 2019; Mai & Liao, 2019; Deng et al., 2019; Kini & Thrampoulidis, 2020; Mignacco et al., 2020a). While we concentrate on random features, we note that we can recover the performance of kernel methods (Rahimi & Recht, 2008; 2009) by sending $\gamma \to \infty$. Indeed, as $\gamma = P/D$ grows, the gram matrix converges to the limiting kernel gram matrix in the high-dimensional regime; detailed studies of the convergence in this regime can be found in (El Karoui, 2010; Pennington & Worah, 2017; Louart et al., 2018; Liao & Couillet, 2018). We can thus recover the performance for any general distance or angle based kernel method, e.g. the NTK of Jacot et al. (2018), by considering γ large enough in our computations with random features. Note, however, that this must be done with some care. Our results for random projections are given for $N > P$. As discussed by Ghorbani et al. (2019); Mei et al. (2021), the relevant dimension for random features performances is, rather than P, the minimum between N and P. Since we focus here in the regime where increasing P beyond $O(D)$, and therefore γ beyond $O(1)$, is not allowed. Indeed, we shall see that Lazy training methods such as kernels or random projections require asymptotically $N = O(D^2)$ samples to beat a random guess, while neural-networks achieves oracle-like performances with only $N = O(D)$ samples.

Reproducibility We provide code to reproduce our plots and solve the equations of Sec. 2.2 at github.com/mariaref/rfvs2lnn_GMM_online.

1.3. Further related work

Separation between kernels & 2LNN Barron (1993) already discussed the limitations of approximating functions with a bounded number of random features within a worst-case analysis. Yehudai & Shamir (2019) construct a data distribution that can be efficiently learnt by a single ReLU neuron, but not by random features. Wei et al. (2019) studied the separation between 2LNN & RF and show the *existence* of a small ($K \sim O(1)$) network that beats kernels on this data distribution, and study the *dynamics* of learning in the same mean-field limit as Chizat & Bach (2020)

and Ghorbani et al. (2019; 2020). Likewise, Li et al. (2020) show separation between kernels & neural networks in the mean-field limit on the phase retrieval problem. Geiger et al. (2020) investigated numerically the role of architecture and data in determining whether lazy or feature learning perform better. Paccolat et al. (2021) studied how neural networks can compress inputs of effectively low-dimensional data.

Gaussian mixture classification is a well-studied problem in statistical learning theory, and its supervised version was recently considered in a series of works from the perspective of Bayes-optimal inference (Lelarge & Miolane, 2019; Mai & Liao, 2019; Deng et al., 2019). Mignacco et al. (2020a;b) studied the dynamics of stochastic gradient descent on a finite training set using dynamical mean-field theory for the perceptron, which corresponds to the case $K = 1, v^1 = 1$ in Eq. (1). Liao & Couillet (2019) and Couillet (2019) studied mixture classification with kernel in an unsupervised setting using random matrix theory.

Dynamics of 2LNN A classic series of papers by Biehl & Schwarze (1995) and Saad & Solla (1995a) studied the dynamics of 2LNN as in Eq. (1) trained using online SGD in the classic teacher-student setup (Gardner & Derrida, 1989), where inputs x are element-wise i.i.d. Gaussian variables and labels are obtained from a "teacher" network with random weights. They derived a set of closed ODEs that track the test error of the student (see also Saad & Solla (1995b); Biehl et al. (1996); Saad (2009) for further results and Goldt et al. (2019) for a recent proof of these equations). There have been several extensions of this approach to different data distributions (Yoshida & Okada, 2019; Goldt et al., 2020b;a). All of these works, though, consider the label y as a function of the input x, or as a function of a latent variable from which x is generated. Here, we extend this type of analysis to a case where the input is conditional on the label, a point of view taken implicitly by Cohen et al. (2020).

The reduction of the dynamics to a set of low-dimensional ODEs should be contrasted with the "mean-field" approach, where the number of hidden neurons K is sent to infinity while the input dimension D is kept finite. In this limit, the neural networks are still a more expressive function class than the corresponding reproducing kernel Hilbert space (Chizat & Bach, 2018; Sirignano & Spiliopoulos, 2019; Rotskoff & Vanden-Eijnden, 2018; Mei et al., 2018). The evolution of the network parameters in this limit can be described by a high-dimensional partial differential equation. This analysis was used in the aforementioned works by Ghorbani et al. (2019; 2020).

2. Neural networks for GM classification

2.1. Setup

We draw inputs $x = (x_i) \in \mathbb{R}^D$ from a high-dimensional Gaussian mixture, where all samples from one Gaussian are assigned to one of two possible labels $y = \pm 1$, which are equiprobable. The data distribution is thus

$$q(x, y) = q(y)q(x|y), \quad q(x|y) = \sum_{\alpha \in \mathcal{S}(y)} \mathcal{P}_\alpha \mathcal{N}_\alpha(x), \quad (6)$$

where $\mathcal{N}_\alpha(x)$ is a multivariate normal distribution with mean $\mu^\alpha/\sqrt{D}$ and covariance Ω^α. The index set $\mathcal{S}(y)$ contains all the Gaussians that are associated with the label y. We choose the constants $\mathcal{P}_\alpha$ such that $q(x, y)$ is correctly normalised. To simplify notation, we focus on binary classification, which can be learnt using a student with a single output unit. Extending our results to C-class classification, where the student has C output heads, is straightforward.

Training The network is trained using stochastic gradient descent on the *quadratic error* for technical reasons related to the analysis. The update equations for the weights at the μth step of the algorithm, $dw_i^k \equiv (w_i^k)_{\mu+1} - (w_i^k)_\mu$, read

$$dw_i^k = -\frac{\eta}{\sqrt{D}} v^k \Delta g'(\lambda^k) x_i - \frac{\eta}{\sqrt{D}} \kappa w_i^k, \quad (7a)$$

$$dv^k = -\frac{\eta}{D} g(\lambda^k) \Delta - \frac{\eta}{D} \kappa v^k, \quad (7b)$$

where $\Delta = \sum_{j=1}^K v^j g(\lambda^j) - y$ and $\kappa \in \mathbb{R}$ is a L_2-regularisation constant. Initial weights are taken i.i.d. from the normal distribution with standard deviation σ_0. The different scaling of the learning rates η for first and second-layer weights guarantees the existence of a well-defined limit of the SGD dynamics as $D \to \infty$. We make the crucial assumption that at each step of the algorithm, we use a previously unseen sample (x, y) to compute the updates in Eq. (7). This limit of infinite training data is variously known as online learning or one-shot/single-pass SGD.

2.2. Theory for the learning dynamics of 2LNN

Statics Since we are training on the quadratic error, the first step of our analysis is to rewrite the prediction mean-squared error pmse as a sum over the error made on inputs from each Gaussian α in the mixture,

$$\mathsf{pmse}(\theta) = \mathbb{E}_{q(x,y)} (y - \phi_\theta(x))^2$$

$$= \sum_y \sum_{\alpha \in \mathcal{S}(y_i)} q(y_i) \mathcal{P}_\alpha \mathbb{E}_\alpha \left[\sum_k v^k g(\lambda^k) - y \right]^2 \quad (8)$$

where the average $\mathbb{E}_\alpha$ is taken over the αth normal distribution $\mathcal{N}_\alpha$ for fixed parameters θ. To evaluate the average,

notice that the input x only enters the expression via products with the student weights $\lambda = (\lambda^k)$; we can hence replace the high-dimensional averages over x with an average over the K "local fields" λ^k. An important simplification occurs since the λ^k are jointly Gaussian when averages are evaluated over just a single distribution in the mixture. We write the first two moments of the local fields as $M = (M_\alpha^k)$ and $Q = (Q_\alpha^{k\ell})$, with

$$M_\alpha^k \equiv \mathbb{E}_\alpha \, \lambda^k = \frac{1}{D} \sum_r w_r^k \mu_r^\alpha, \qquad (9a)$$

$$Q_\alpha^{k\ell} \equiv \mathsf{Cov}_\alpha \left(\lambda^k, \lambda^\ell \right) = \frac{1}{D} \sum_{r,s} w_r^k \Omega_{rs}^\alpha w_s^\ell. \qquad (9b)$$

Any average over a Gaussian distribution is a function of only the first two moments of that distribution, so the pmse can be written as a function of the "order parameters" M and Q and of the $K \sim O(1)$ second-layer weights $v = (v^k)$:

$$\lim_{D \to \infty} \mathsf{pmse}(\theta) \to \mathsf{pmse}(Q, M, v). \qquad (10)$$

Likewise, the classification error ϵ_c (2) can also be written as a function of the order parameters only: $\lim_{D \to \infty} \epsilon_c(\theta) \to \epsilon_c(Q, M, v)$. The order parameters have a clear interpretation: M_α^k encodes the overlap between the kth student node and the mean of the α cluster, and plays a similar role to the teacher-student overlap in the vanilla teacher-student scenario. Q_α^{kl} instead tracks the overlap between the various student weight vectors, with the input-input covariance Ω^α intervening. The strategy for our analysis is thus to derive equations that describe how the order parameters (Q, M, v) evolve during training, which will in turn allow us to compute the pmse of the network at all times.

Dynamics   We derived a closed set of ordinary differential equations that describe the evolution of the order parameters in the case where each Gaussian in the mixture has the same covariance matrix Ω. We proceed here with a brief statement of the equations and deffer the detailed derivation to Sec. B.2. The approach is most easily illustrated with the second-layer weights v^k. The key idea to compute the average change in the weight v^k upon an SGD update (7b), dv^k, which can be decomposed into a contribution from every Gaussian in the mixture,

$$\mathbb{E} \, dv^k = \sum_{\alpha \in \mathcal{S}(+)} \mathcal{P}_\alpha \, dv_{\alpha^+}^k + \sum_{\alpha \in \mathcal{S}(-)} \mathcal{P}_\alpha \, dv_{\alpha^-}^k, \qquad (11)$$

where the change $dv_{\alpha^+}^k$ is obtained directly from Eq. (7b),

$$dv_\alpha^k = \frac{\eta}{D} \, \mathbb{E}_\alpha \, y_\alpha g(\lambda^k) - \frac{\eta}{D} \sum_j v^j \, \mathbb{E}_\alpha \, g(\lambda^k) g(\lambda^j) - \frac{\eta}{D} \kappa v^k. \qquad (12)$$

The averages that remain to be computed only involve the true label and the local fields λ. The former is a constant

within each Gaussian while the latter are jointly Gaussian. It follows, that also these averages can be expressed in terms of only the order parameters and the equation closes. As we discuss in the appendix, in the high-dimensional limit $D \to \infty$ the normalised number of samples $t \equiv N/D$ can be interpreted as a continuous time, which allows the dynamics of v^k to be captured by the ODE (B.26).

The order parameters Q require an additional step which consists in diagonalising the sum $Q^{k\ell} \sim \sum_{r,s}^D w_r^k \Omega_{rs} w_s^\ell$ by introducing the integral representation

$$Q^{kl} = \int d\rho \, p_\Omega(\rho) \, \rho q^{kl}(\rho), \qquad (13)$$

where $p_\Omega(\rho)$ is the spectral density of Ω, and $q^{kl}(\rho)$ is a density whose time evolution can be characterised in the thermodynamic limit. We relegate the full expression of the equation of motion for $q^{kl}(\rho)$ to Eq. (B.25) of the appendix. Crucially, it involves only averages that can be expressed in terms of the order parameters (9), and hence the equation closes. Likewise, the order parameter M can be rewritten in terms of a density as $M^{\alpha k} = \int d\rho \, p_\Omega(\rho) m^{\alpha k}(\rho)$. The dynamics of $m^{\alpha k}$ is described by Eq. (B.20).

Solving the equations of motion   The equations are valid for any mean and covariance matrix Ω. Solving them requires evaluating multidimensional integrals of dimension up to 4, e.g. $\mathbb{E}_\alpha \, g'(\lambda^k) \lambda^\ell y$, which can be effeciently estimated using Monte-Carlo (MC) methods. We provide a ready-to-use numerical implementation on the GitHub.

Comparing theory and simulation   On the left of Fig. 2, we plot the evolution of the pmse (8) and the classification error (2) of a 2LNN with $K = 8$ neurons trained on the XOR-like mixture of Fig. 1. We plot the test errors obtained from integration of the order parameters with solid lines, and the same quantities computed using a test set during the simulation with crosses. The agreement between ODE predictions and a single run of SGD is good, even at intermediate system size ($D = 1000$). In the App. B.2, we give additional plots for the simulated dynamics of the individual order parameters and find very good agreement with predictions obtained from the ODEs (cf. Fig. 7). Note that although we initialise the weights of the student randomly and independently of the means, there is an initial overlap between student weights and the means of order $1/\sqrt{D}$ due to finite-size fluctuations. To capture this with the ODEs, we initialise them in a regime of weak recovery, where $M_\alpha^k \neq 0$. For a detailed discussion of the early period of learning up to weak recovery, see Arous et al. (2020).

How 2LNNs learn the XOR-like mixture   A closer look at the learning dynamics on the right of Fig. 2 reveals several phases of learning. There we show the first-layer weight
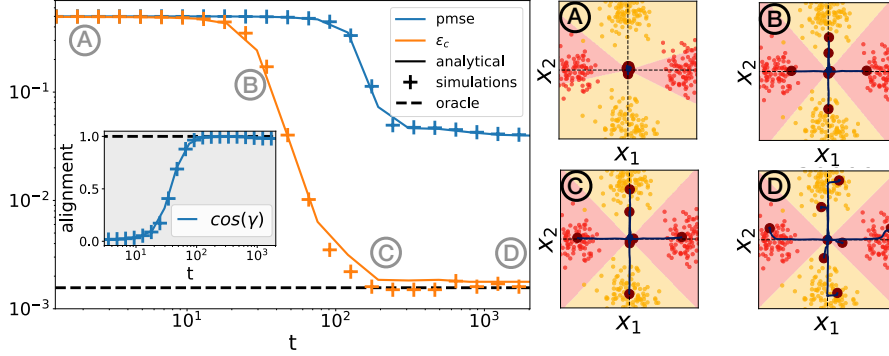
Figure 2. **How the 2LNN learns a XOR-like GM (Left)** Evolution of the prediction mean-squared error pmse (8) and the classification error (2) of a 2LNN with $K = 8$ neurons trained on the XOR-like mixture of Fig. 1. We plot the test errors as obtained from a single simulation with $D = 1000$ (crosses) and from integration of the ODEs of Sec. 2.2. The dashed black line is the classification error of an oracle with knowledge of the means μ^{α}, Eq. (3). The inset shows the mean angle of the network weights to the means of the mixture. **(Right)** Projections of the first layer weights (dots) onto the plane spanned by the means of the XOR-like mixture at different times during training. Shaded areas indicate the decision boundaries of the network, where its output $\phi_\theta(x)$ changes sign. *Parameters:* $K = 8, D = 1000, \sigma = 0.05, \eta = 0.1$ weights initialised with s.t.d. $\sigma_0 = 1, \kappa = 10^{-2}$.

vectors of the 2LNN, projected into the plane spanned by the four means of the mixture, at four different times during training. The regions shaded in red and yellow indicate the decision boundaries of the network, which correspond to the line where the network's output $\phi_\theta(x)$ changes its sign. A 2LNN with $K \geq 4$ neurons can approach the classification error of the oracle (3) if its weight vectors approach the four means, with corresponding second-layer weights. Panel (C) shows that network reaches this configuration. However, this configuration does not minimise the mean-squared error used during training (7), so eventually the weights depart slightly from the means to converge to a solution with lower mean squared error (D). This is confirmed by the inset on the left of Fig. 2, where we see that the average angle of the network weights to the means has a maximum around $t = 300$, before decaying slightly at the end of training.

2.3. Predicting the long-time performance of 2LNN

Direct integration of the ODEs is numerically expensive. A more straight forward way to extract information from the ODEs is to find their asymptotic fixed point, which fully characterises the $t \to \infty$ performance of the network. However, the number of equations is already 26 for a 2LNN with 4 neurons trained on the XOR mixture, and scales like K^2. The key to finding fixed points efficiently is thus to make an *ansatz* with fewer degrees of freedom for the matrices Q and M which solve the equations. For example, one could impose $Q^{kk} = Q$ and $Q^{k\ell} = C, k \neq \ell$. By exploiting the symmetries of the XOR-like mixture, we find that the fix points of the equations can be described by only $K = 4$ parameters: $K/2$ angles between weight vectors and means, and $K/2$ norms, as described in Appendix B.3. Finding the fixed-point of this reduced four-dimensional system allows to compute, for example, the dependence of the generali-
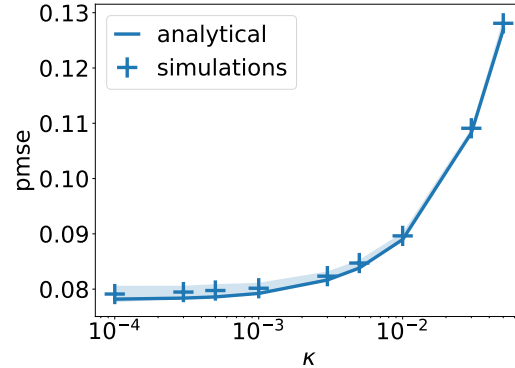

Figure 3. Prediction mean-squared error (8) on the XOR-like mixture versus weight decay. Results obtained from a fixed point analysis of the ODEs with 4 degrees of freedom discussed in Sec. 2.3 ($\sigma^2 = 0.1, K = 4, \eta = 0.1, 10^4$) Monte-Carlo samples. The shaded area indicates standard deviation over 10 runs.

sation error as a function the regularisation in Fig. 3. The agreement between simulation and analytical predictions is again good, and we find that increasing the regularisation only increases the test error of the student. This is in agreement with previous work on two-layer networks in the same limit in the teacher-student setup, where L_2-regularisation was also found to hurt performance (Saad & Solla, 1997).

2.4. The impact of over-parametrisation

We also studied the effect of over-parametrisation, which we define as the number of additional neurons a student has on top of the $K = 4$ neurons that it needs to reach the oracle's performance on the XOR mixture. We show in the inset of Fig. 4 that over-parametrisation does *not* improve final performance, since the remaining error of the student is dominated by "spill-over" of points from one mixture into
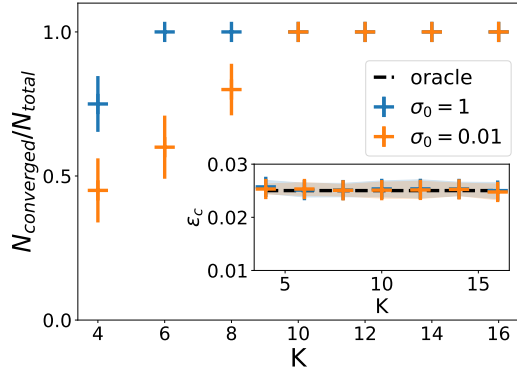
Figure 4. Fraction of simulations that converged to the optimal solution for the 2LNN out of 20 simulations for increasing values of K. Overparametrisation increases the probability of finding the optimal solution but does not affect classification performances. Simulations started with initial weights of std.dev. σ_0. The inset shows the classification error of the networks that converged. *Parameters:* $D = 800, \eta = 0.1, \kappa = 0, \sigma^2 = 0.1$, run time 10^5.

adjacent quadrants. However, over-parametrisation leads to an "implicit acceleration" effect: over-parametrised networks are much more likely to converge to a solution that approaches the oracle's performance, as we show in the main of Fig. 4. The term "implicit acceleration" was coined by Arora et al. (2018) for similar effects in deep neural networks, and analysed for two-layer networks in the teacher-student setup by Livni et al. (2014); Safran & Shamir (2018). A complete understanding of the phenomenon remains an open problem, which we leave for future work.

3. Random features on GM classification

To understand the performance of random features on Gaussian Mixtures classification, we analyse the performances of the linear model (5) trained with online SGD with the squared loss on the random features z (4) (Steinwart et al., 2009; Caponnetto & De Vito, 2007).

First, we assume that we have enough samples, so that $N \gg P$, and discuss the situation when $N \ll P$ later. For any finite D, P, running the algorithm up to convergence then corresponds to taking the limit $t \to \infty$. The random features' weights converge to an estimate $\hat{W}$ which can be computed analytically, see Eq. D.6, and allows to precisely characterise the test error:

$$\text{pmse}_{t \to \infty} = \frac{1}{2}\Big(1 - \sum_\tau \frac{\tilde{\Phi}_\tau^2}{\rho_\tau}\Big), \qquad (14)$$

where ρ_τ are the eigenvalues of the feature's covariance matrix $\Omega_{ij} = \mathbb{E}\, z_i z_j$, with associated eigenvector Γ_τ. $\tilde{\Phi}_\tau \equiv \sum_{i=1}^P \Gamma_{\tau i}\Phi_i/\sqrt{P}$ is the input-label covariance after rotation into the eigenbasis of Ω (see Appendix D). Crucially, the test error and $\hat{W}$ *only* depend on the first two moments of

the features. The formula for these moments, as well as the one for the classification error can be obtained when P, D are large using the Gaussian equivalence of (Goldt et al., 2020a). Indeed, the distribution of the features z remains a mixture of distributions (see App. C). We then define

$$M_\alpha = \sum_{i=1}^P \frac{\hat{w}_i\, \mathbb{E}_\alpha[z_i]}{\sqrt{P}}, \;\; Q_\alpha = \sum_{i=1}^P \frac{\hat{w}_i \hat{w}_j}{P}\, \mathsf{Cov}_\alpha(z, z), \quad (15)$$

and we find for D, P large enough, that

$$\epsilon_{ct \to \infty} = \frac{1}{2}\left(1 - \sum_\alpha \mathcal{P}_\alpha\, y\, \text{erf}\left(\frac{M_\alpha}{\sqrt{2Q_\alpha}}\right)\right). \qquad (16)$$

As discussed in App. C.3, in the case of ReLU activation function, the feature distribution $p(z_i)$ is a truncated Gaussian. Hence, at large D, P, both the mean of z_i and the population covariance $\mathsf{Cov}(z_i, z_j)$ can be obtained analytically in terms of the matrix F and means μ, see Eq. (C.16) and (C.19) for the full result.

We used this formula to obtain precisely the error (14), and the results are shown in Fig. 5. We see that the RF error is a function of $\sigma D^{1/2}/P^{1/4} = \sigma(D/\gamma)^{1/4}$ leading to the conclusion that – as discussed in Fig. 1 – the "transition" from the high to low snr regime happens when $\sigma^{-1} \approx D^{1/2}/P^{1/4}$. This scaling further reveals that $P \approx D^2$ features are required in order to obtain good performance. The validity of Eq. 16 is verified in Fig. 9. Reaching this performance, however, requires the number of samples N to be larger than P, so $N > O(D^2)$. In the so-called *high-dimensional* regime analysed in this paper, where $N \approx D$, such performances remain out of reach. The scaling analysis can be easily generalised; as discussed by Ghorbani et al. (2019); Mei et al. (2021), the relevant dimension for RF performances is, rather than P, the minimum between N and P.

The classification error is thus a function of $\sigma D^{1/2}/\min(N,P)^{1/4}$. If N is $O(D)$, then even in the kernel limit when $P \to \infty$, the performance degrades to no more than a random guess as soon as

$$\sigma \gg N^{1/4}/D^{1/2}, \qquad (17)$$

and therefore for any value of σ when $D, N \to \infty$ with fixed N/D. In a nutshell, for any fixed σ, lazy training methods such as random features or kernels will fail to beat a random guess in the high-dimensional limit. This, and the requirement of at least $N = O(D^2)$ samples to learn, are to be contrasted with the the oracle-like performance achieved by a simple neural net with only $N = O(D)$ samples.

Why do random features fail? The linear regime of features maps This raises the question of *why* the random feature and kernel methods fail in the high-dimensional setting. As we shall see – and this has been already discussed
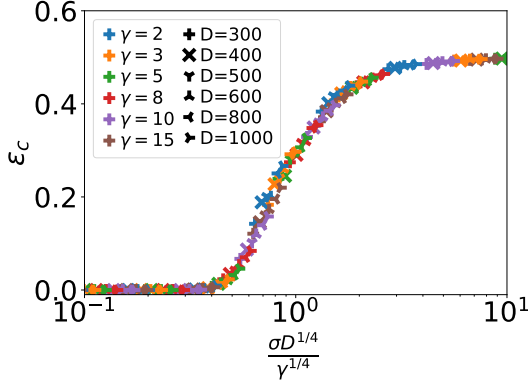
Figure 5. **Evolution of the classification error of random features (RF) for various values of** σ, $\gamma = P/D$ **and** D **on the XOR mixture of Fig. 1, in the limit of large number of samples** $N \gg P$. All these different cases can be collapsed into a single master curve by plotting the classification error versus $\sigma D^{1/2}/P^{1/4} = \sigma D^{1/4}/\gamma^{1/4}$, showing that for large D, P, it should be a function $\epsilon_c = f(\sigma D^{1/2}/P^{1/4})$. For a finite input dimension D, increasing the number of features allows RF to perform increasingly better as they approach the Kernel limit. Analytical predictions are obtained by the linear regression analysis of Eq. (16) for RF. *Parameters:* $\eta = 0.1$, $|\mu|/\sqrt{D} = 1$. Note however that this analysis requires $N \gg P$. Given random features are sensitive to the minimum of P and N (Ghorbani et al., 2019; Mei et al., 2021) the effective scaling variable is rather $\sigma D^{1/2}/\min(P, N)^{1/4}$ (see text).

in different contexts by El Karoui (2010); Mei & Montanari (2019) – this can be understood analytically from the fact that the feature map is effectively linear when when $P = O(D)$. This section is now dedicated to computing the moments $\mathbb{E}_\alpha[z_i]$ and $\mathsf{Cov}_\alpha(z, z)$ analytically in this region, revealing that this effective linearity is indeed the underlying reason for the failure of RF in this regime.

Since the mixture remains a mixture after the application of random features, our main task is to compute the new means and variances of the distribution in the transformed space. We thus focus on transformation of a random variable drawn from a single Gaussian $x_r = \mu_r/\sqrt{D} + \sigma w_r$, where w_r is a standard Gaussian, in the kernel, or random feature, space.

For generic activation function the first two moments of the features can be obtained in the well studied low signal-to-noise regime snr $\sim O(1)$. Key to do so, is the observation that $F_{ir}\mu_r/D \sim O(1/\sqrt{D})$. The activation function can thus be expanded in orders of $1/D$ and its action is essentially linear. We define the constants

$$a \equiv \mathbb{E}\,\psi\,(\sigma\zeta), \quad b \equiv \mathbb{E}\,\zeta\psi\,(\sigma\zeta), \quad c^2 \equiv \mathbb{E}\,\psi\,(\sigma\zeta)^2 \quad (18)$$

with the expectation taken over the standard Gaussian random variable ζ. To leading order, the mean and covariance of the features are given by (cf. Sec. C):

$$\mathbb{E}\,z_i = a + b\sum_{r=1}^{D}\frac{F_{ir}\mu_r}{\sigma D} \quad (19)$$

$$\mathrm{cov}(z_i, z_j) = \begin{cases} c^2 - a^2, & i = j, \\ b^2 \sum_r \frac{F_{ir}F_{jr}}{D} & i \neq j. \end{cases} \quad (20)$$

This computation immediately reveals the reason random features cannot hope to learn in the low snr regime: the transformation of the means is only linear; hence a Gaussian mixture that is not linearly separable in input space will remain so even after random features. In other words, if the centres of the Gaussian are too close, the kernel fails to map the data non-linearly to a large dimensional space. In contrast, in the high-snr regime, where the centres are separated enough, the non-linearity kicks-in and the data becomes separable in feature space.

Relation to kernel methods The same argument explains the failure of kernel methods: if two centres $\mathbf{x}$ and $\mathbf{y}$ are close, the kernel function $K(\mathbf{x}, \mathbf{y})$ can be expanded to low order and the kernel is essentially linear, leading to bad performances. The connection can be made explicit using the convergence of random features to a kernel (Rahimi & Recht, 2008; 2009):

$$K(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{P}\sum_{i=1}^{P}\mathop{\mathbb{E}}_{F}\left[\psi\left(\sum_{r=1}^{D}\frac{x_r F_{ir}}{\sqrt{D}}\right)\psi\left(\sum_{s=1}^{D}\frac{y_s F_{is}}{\sqrt{D}}\right)\right],$$
(21)

At low SNR, the constants a, b, c can be obtained from the kernel via

$$c^2 = \mathbb{E}\,K(\sigma\boldsymbol{\omega}_1, \sigma\boldsymbol{\omega}_1), \quad a^2 = \mathbb{E}\,K(\sigma\boldsymbol{\omega}_1, \sigma\boldsymbol{\omega}_2),$$
$$b^2 = D\sigma^2\left[-a^2 + \mathbb{E}\,K\left(\frac{\boldsymbol{\mu}}{\sqrt{D}} + \sigma\boldsymbol{\omega}_1, \frac{\boldsymbol{\mu}}{\sqrt{D}} + \sigma\boldsymbol{\omega}_2\right)\right]$$

where the average is taken over two standard Gaussian random vectors $\omega_1, \omega_2 \in \mathbb{R}^D$. This relation, similar in nature to one of El Karoui (2010), allows to express the statistical properties of the features directly from the kernel function.

4. Neural networks vs random features

We now collect our results for a comparison of the performance of 2LNN and RF on the XOR-like mixture from Fig. 1. We look at three different regimes for the snr, illustrated in the first column of Fig. 6. The second column visualises the mixture after the Gaussian random features transformation with $\psi(x) = \max(0, x)$ (4). The third and fourth columns show the evolution of the pmse of 2LNN and RF, respectively, during training with online SGD. Since overparametrisation does not impact the 2LNN's performance in these tasks, Sec. 2.3, we train a $K = 10$ network to increase the number of runs that converge.

At **low** snr (a) the distance of each Gaussian to the origin is $O(1)$ and the standard deviation $\sigma \sim O(1)$ as well. The
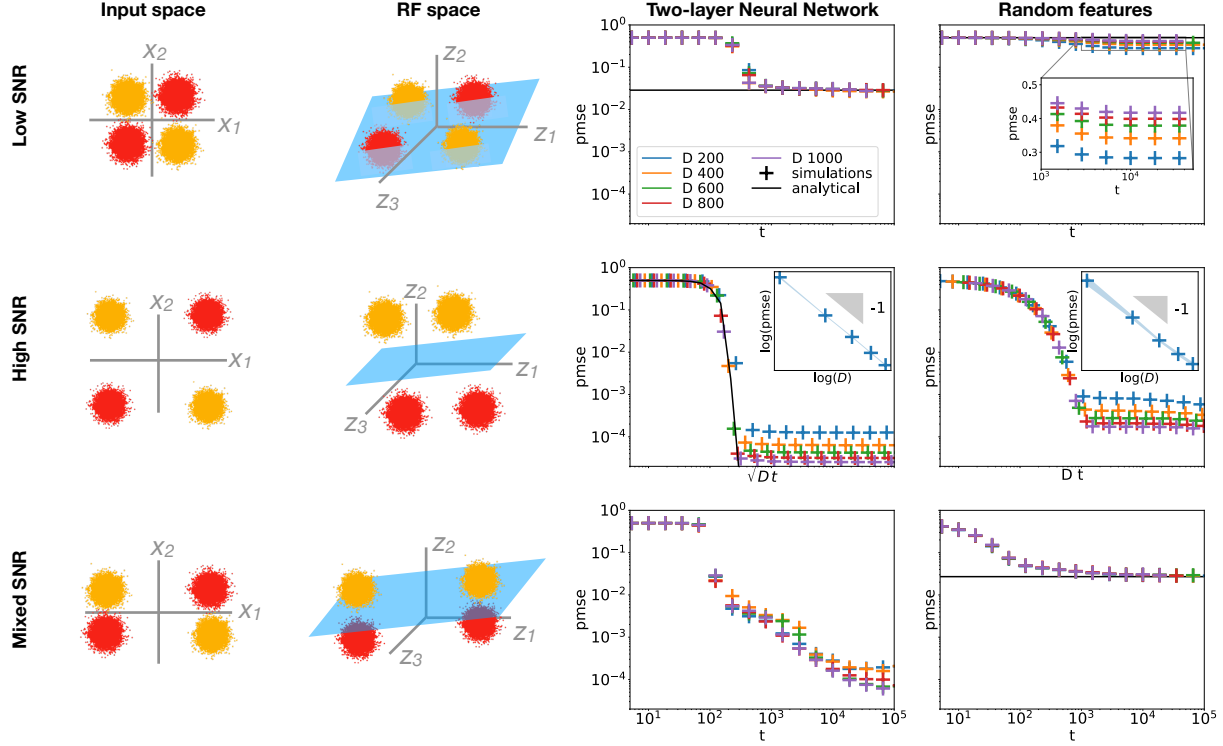
Figure 6. **The performance of 2LNN and RF on the XOR-like mixture with different signal-to-noise ratios.** The *first and second columns* show the XOR-like mixture in input space and random feature space, resp. The *third and fourth columns* show the pmse (8) of 2LNN and RF during training, resp. In the low snr regime (top row), while the 2LNN learns a non trivial function of the inputs, RF cannot perform better than random chance since the XOR-like mixture remains one in feature space. Both networks learn to classify the XOR-mixture in the high snr regime (middle row) as the clusters become well separated in feature space. For mixed snr (bottom row), even though both networks do better than random guessing, 2LNN outperform RF as the distance between opposite sign clusters remains of order one in feature space. In all plots, crosses are obtained from simulations with input dimension $D = 1000$. Solid black lines in the 2LNN plots are obtained using tools from Sec. 2.2 (long-time performance of Sec. 2.3 for the first row and integration of the odes for the second). Solid black lines in the RF plots indicate the test error obtained from the analysis of Sec. 3 with $D = 10000$. $\sigma^2 = 0.05, P = 2D, \eta = 0.1, K = 10, \sigma_0 = 10^{-2}$.

two-layer neural network learns to predict the correct labels almost as well as the oracle (3). Its performance does not depend on D, and using the long-time solution of Sec. 2.3, we can predict its asymptotic error (black line) which agrees well with simulations (crosses). In contrast, random features display an asymptotic error that approaches random guessing as the input dimensions increases (inset). This is clear from Eq. 20: in the large D limit, random features only produce a *linear* transformation of their input. The XOR therefore remains a XOR in RF space, leading linear regression's failure to do better than chance.

At **high** snr (b), the distance between the clusters scales as $\sqrt{D}$ while the σ remains fixed. The asymptotic error of the 2LNN thus decreases with D and the network is able to learn perfectly in the $D \to \infty$ limit (black line). The error of random features also approaches 0 as $D \to \infty$, since the mixture is now well separated in random feature space, too.

We finally consider a regime of **mixed** snr (c) where the mixture is well-separated in one dimension, but very close in the other dimension. We achieve this by setting $\mu_1^0 \sim \sqrt{D}, \mu_2^0 \sim D$ for the mean of the first mixture, etc. Random features then achieve a non-trivial generalisation error, which can be understood by considering the means of the features z_i. The large component μ_2, induces the activation function to perform a non-linear transformation of the centres and allows for opposite sign centroids to be separated by a hyper-plane in feature space. The small component μ_1, causes the distance between opposite sign centroids, which is of order $O(1)$ in input space, to remain of order one in feature space, for all D. This leads to a finite generalisation error of RF which remains invariant with increasing input dimension. In this regime, the 2LNN still achieve better performance than the random features, thereby completing the picture we developed in Fig. 1.

Acknowledgements

References

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. 2018.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pp. 242–252. PMLR, 2019.

Arora, S., Cohen, N., and Hazan, E. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International Conference on Machine Learning*, pp. 244–253. PMLR, 2018.

Arora, S., Du, S., Hu, W., Li, Z., Salakhutdinov, R., and Wang, R. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, pp. 8141–8150, 2019.

Arous, G. B., Gheissari, R., and Jagannath, A. A classification for the performance of online sgd for high-dimensional inference. *arXiv:2003.10409*, 2020.

Bach, F. Breaking the curse of dimensionality with convex neural networks. *The Journal of Machine Learning Research*, 18(1):629–681, 2017.

Barron, A. R. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory*, 39(3):930–945, 1993.

Biehl, M. and Schwarze, H. Learning by on-line gradient descent. *J. Phys. A. Math. Gen.*, 28(3):643–656, 1995.

Biehl, M., Riegler, P., and Wöhler, C. Transient dynamics of on-line learning in two-layered neural networks. *Journal of Physics A: Mathematical and General*, 29(16), 1996.

Cao, Y. and Gu, Q. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 10836–10846, 2019.

Caponnetto, A. and De Vito, E. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.

Chizat, L. and Bach, F. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in Neural Information Processing Systems 31*, pp. 3040–3050, 2018.

Chizat, L. and Bach, F. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on Learning Theory*, pp. 1305–1338. PMLR, 2020.

Chizat, L., Oyallon, E., and Bach, F. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems*, pp. 2937–2947, 2019.

Cohen, U., Chung, S., Lee, D., and Sompolinsky, H. Separability and geometry of object manifolds in deep neural networks. *Nature communications*, 11(1):1–13, 2020.

Couillet, R. High dimensional robust classification: A random matrix analysis. In *2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pp. 420–424, 2019.

Daniely, A. and Malach, E. Learning parities with neural networks. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Deng, Z., Kammoun, A., and Thrampoulidis, C. A model of double descent for high-dimensional binary linear classification. *arXiv:1911.05822*, 2019.

Du, S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019.

El Karoui, N. The spectrum of kernel random matrices. *Ann. Statist.*, 38(1):1–50, 02 2010.

Gardner, E. and Derrida, B. Three unfinished works on the optimal storage capacity of networks. *Journal of Physics A: Mathematical and General*, 22(12):1983–1994, 1989.

Garriga-Alonso, A., Rasmussen, C., and Aitchison, L. Deep convolutional networks as shallow gaussian processes. In *International Conference on Learning Representations*, 2019.

Geiger, M., Spigler, S., Jacot, A., and Wyart, M. Disentangling feature and lazy training in deep neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(11):113301, 2020.

Ghorbani, B., Mei, S., Misiakiewicz, T., and Montanari, A. Limitations of lazy training of two-layers neural network. In *Advances in Neural Information Processing Systems*, volume 32, pp. 9111–9121, 2019.

Ghorbani, B., Mei, S., Misiakiewicz, T., and Montanari, A. When do neural networks outperform kernel methods? In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Goldt, S., Advani, M., Saxe, A., Krzakala, F., and Zdeborová, L. Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. In *Advances in Neural Information Processing Systems 32*, 2019.

Goldt, S., Loureiro, B., Reeves, G., Mézard, M., Krzakala, F., and Zdeborová, L. The gaussian equivalence of generative models for learning with two-layer neural networks. *arXiv:2006.14709*, 2020a.

Goldt, S., Mézard, M., Krzakala, F., and Zdeborová, L. Modeling the influence of data structure on learning in neural networks: The hidden manifold model. *Phys. Rev. X*, 10(4):041044, 2020b.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems 32*, pp. 8571–8580, 2018.

Kini, G. R. and Thrampoulidis, C. Analytic study of double descent in binary classification: The impact of loss. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 2527–2532. IEEE, 2020.

Lee, J., Sohl-Dickstein, J., Pennington, J., Novak, R., Schoenholz, S., and Bahri, Y. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

Lelarge, M. and Miolane, L. Asymptotic bayes risk for gaussian mixture in a semi-supervised setting. In *2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pp. 639–643. IEEE, 2019.

Li, Y. and Liang, Y. Learning Overparameterized Neural Networks via Stochastic Gradient Descent on Structured Data. In *Advances in Neural Information Processing Systems 31*, 2018.

Li, Y., Ma, T., and Zhang, H. R. Learning over-parametrized two-layer neural networks beyond ntk. In Abernethy, J. and Agarwal, S. (eds.), *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pp. 2613–2682. PMLR, 2020.

Li, Z., Wang, R., Yu, D., Du, S. S., Hu, W., Salakhutdinov, R., and Arora, S. Enhanced convolutional neural tangent kernels. *arXiv:1911.00809*, 2019.

Liao, Z. and Couillet, R. On the spectrum of random features maps of high dimensional data. In *International Conference on Machine Learning*, pp. 3063–3071. PMLR, 2018.

Liao, Z. and Couillet, R. On inner-product kernels of high dimensional data. In *2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pp. 579–583, 2019.

Livni, R., Shalev-Shwartz, S., and Shamir, O. On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*, volume 27, pp. 855–863, 2014.

Louart, C., Liao, Z., and Couillet, R. A random matrix approach to neural networks. *The Annals of Applied Probability*, 28(2):1190–1248, 2018.

Mai, X. and Liao, Z. High dimensional classification via empirical risk minimization: Improvements and optimality. *arXiv preprint arXiv:1905.13742*, 2019.

Matthews, A. G. d. G., Hron, J., Rowland, M., Turner, R., and Ghahramani, Z. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations*, 2018.

Mei, S. and Montanari, A. The generalization error of random features regression: Precise asymptotics and double descent curve. arXiv:1908.05355, 2019.

Mei, S., Montanari, A., and Nguyen, P. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

Mei, S., Misiakiewicz, T., and Montanari, A. Generalization error of random features and kernel methods: hypercontractivity and kernel matrix concentration. *arXiv preprint arXiv:2101.10588*, 2021.

Mignacco, F., Krzakala, F., Lu, Y. M., and Zdeborová, L. The role of regularization in classification of high-dimensional noisy gaussian mixture. In *37th International Conference on Machine Learning*, 2020a.

Mignacco, F., Krzakala, F., Urbani, P., and Zdeborová, L. Dynamical mean-field theory for stochastic gradient descent in gaussian mixture classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020b.

Paccolat, J., Petrini, L., Geiger, M., Tyloo, K., and Wyart, M. Geometric compression of invariant manifolds in neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(4):044001, 2021. doi: 10.1088/1742-5468/abf1f3. URL https://doi.org/10.1088/1742-5468/abf1f3.

Pennington, J. and Worah, P. Nonlinear random matrix theory for deep learning. In *Advances in Neural Information Processing Systems*, pp. 2637–2646, 2017.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pp. 1177–1184, 2008.

Rahimi, A. and Recht, B. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in neural information processing systems*, pp. 1313–1320, 2009.

Riegler, P. and Biehl, M. On-line backpropagation in two-layered neural networks. *Journal of Physics A: Mathematical and General*, 28(20), 1995.

Rotskoff, G. and Vanden-Eijnden, E. Parameters as interacting particles: long time convergence and asymptotic error scaling of neural networks. In *Advances in Neural Information Processing Systems 31*, pp. 7146–7155, 2018.

Saad, D. *On-line learning in neural networks*, volume 17. Cambridge University Press, 2009.

Saad, D. and Solla, S. Exact Solution for On-Line Learning in Multilayer Neural Networks. *Phys. Rev. Lett.*, 74(21): 4337–4340, 1995a.

Saad, D. and Solla, S. On-line learning in soft committee machines. *Phys. Rev. E*, 52(4):4225–4243, 1995b.

Saad, D. and Solla, S. Learning with Noise and Regularizers Multilayer Neural Networks. In *Advances in Neural Information Processing Systems 9*, pp. 260–266, 1997.

Safran, I. and Shamir, O. Spurious local minima are common in two-layer relu neural networks. In *International Conference on Machine Learning*, pp. 4433–4441. PMLR, 2018.

Scholkopf, B. and Smola, A. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. Adaptive Computation and Machine Learning series, 2018.

Shankar, V., Fang, A., Guo, W., Fridovich-Keil, S., Ragan-Kelley, J., Schmidt, L., and Recht, B. Neural kernels without tangents. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8614–8623. PMLR, 2020.

Sirignano, J. and Spiliopoulos, K. Mean field analysis of neural networks: A central limit theorem. *Stochastic Processes and their Applications*, 2019.

Steinwart, I., Hush, D. R., Scovel, C., et al. Optimal rates for regularized least squares regression. In *COLT*, pp. 79–93, 2009.

Suzuki, T. and Akiyama, S. Benefit of deep learning with non-convex noisy gradient descent: Provable excess risk bound and superiority to kernel methods. *arXiv preprint arXiv:2012.03224*, 2020.

Wang, C., Hu, H., and Lu, Y. A solvable high-dimensional model of gan. In *Advances in Neural Information Processing Systems*, pp. 13759–13768, 2019.

Wei, C., Lee, J. D., Liu, Q., and Ma, T. Regularization matters: Generalization and optimization of neural nets v.s. their induced kernel. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Yehudai, G. and Shamir, O. On the power and limitations of random features for understanding neural networks. In *Advances in Neural Information Processing Systems*, volume 32, pp. 6598–6608, 2019.

Yoshida, Y. and Okada, M. Data-dependence of plateau phenomenon in learning with neural network — statistical mechanical analysis. In *Advances in Neural Information Processing Systems 32*, pp. 1720–1728, 2019.

Yoshida, Y., Karakida, R., Okada, M., and Amari, S.-I. Statistical mechanical analysis of learning dynamics of two-layer perceptron with multiple output units. *Journal of Physics A: Mathematical and Theoretical*, 52(18): 184002, 2019.