Out-of-Distribution Generalization in Kernel Regression

Anonymous Author(s)
Affiliation
Address
email

Abstract

In real word applications, data generating process for training a machine learning model often differs from what the model encounters in the test stage. Understanding how and whether machine learning models generalize under such distributional shifts have been a theoretical challenge. Here, we study generalization in kernel regression when the training and test distributions are different using methods from statistical physics. Using the replica method, we derive an analytical formula for the out-of-distribution generalization error applicable to any kernel and real datasets. We identify an overlap matrix that quantifies the mismatch between distributions for a given kernel as a key determinant of generalization performance under distribution shift. Using our analytical expressions we elucidate various generalization phenomena including possible improvement in generalization when there is a mismatch. We develop procedures for optimizing training and test distributions for a given data budget to find best and worst case generalizations under the shift. We present applications of our theory to real and synthetic datasets and for many kernels. We compare results of our theory applied to Neural Tangent Kernel with simulations of wide networks and show agreement. We analyze linear regression in further depth.

1 Introduction

Machine learning models are trained to perform predictions on previously unseen samples. A central goal of machine learning theory has been to understand this generalization performance [1, 2]. While most of the theory in this domain focused on generalizing in-distribution, i.e. when the training examples are sampled from the same distribution as that of the test examples, in real world applications there is often a mismatch between the training and the test distributions. Such difference, even when small, may lead to large effects in generalization performance [3, 4, 5, 6, 7]. Hence, novel theoretical accounts of how a model generalizes out-of-distribution (OOD) are needed.

In this paper, we provide a theory of OOD generalization for kernel regression [8, 9, 10] applicable to any kernel and real datasets. Besides being a popular machine learning method itself, kernel regression is recovered from an infinite width limit of neural networks [11], making our results relevant to understanding OOD generalization in deep neural networks. Indeed, it has been argued that understanding generalization in kernel methods is necessary for understanding generalization in deep learning [12].

Using methods from statistical physics [13] and building on techniques developed in a recent line of work [14, 15], we obtain an analytical expression for generalization in kernel regression when test data is drawn from a different distribution than the training distribution. Our results apply to average or typical case learning curves and describe numerical experiments well including those on real data.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

In contrast, previous works mostly focus on worst-case OOD generalization bounds in the spirit of statistical learning theory [16, 17, 1].

We show examples of how mismatched training and test distributions affect generalization and in particular that the mismatch may improve test performance. We use our analytical expression to develop a procedure for minimizing generalization error with respect to the training distribution and present applications of it. We study OOD generalization with a linear kernel in depth and present examples of how dimensional reduction of the training distribution can be helpful in generalization, including shifting of a double-descent peak [18], when the target does not depend on certain features. We present further analyses of various models in Supplementary Information (SI).

2 OOD Generalization Error for Kernel Regression from the Replica Method

We consider kernel regression in the setting where training examples are sampled i.i.d. from a distribution $p(\mathbf{x})$ but the test examples are drawn from a different distribution $\tilde{p}(\mathbf{x})$. We derive our main analytical formula for the generalization error here, and illustrate and discuss its implications in the following sections.

2.1 Problem setup

We consider a training set $\mathcal{D} = \{(\mathbf{x}^\mu, y^\mu)\}_{\mu=1}^P$ of size P where the D-dimensional inputs $\mathbf{x} \in \mathbb{R}^D$ are drawn independently from a training distribution $p(\mathbf{x})$ and the noisy labels are generated from a target function $y^\mu = \bar{f}(\mathbf{x}^\mu) + \epsilon^\mu$ where the noise covariance is $\langle \epsilon^\mu \epsilon^\nu \rangle = \varepsilon^2 \delta_{\mu\nu}$. Kernel regression model is trained through minimizing the training error:

$$f^* = \underset{f \in \mathcal{H}}{\arg\min} \frac{1}{2} \sum_{\mu=1}^{P} \left(f(\mathbf{x}^\mu) - y^\mu \right)^2 + \lambda \langle f, f \rangle_{\mathcal{H}} \,, \tag{1}$$

where $\mathcal{H}$ is a Reproducing Kernel Hilbert Space (RKHS) associated with a positive semi-definite kernel $K(\mathbf{x}, \mathbf{x}') : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$, and $\langle ., . \rangle_{\mathcal{H}}$ is the Hilbert inner product.

The generalization error on the test distribution $\tilde{p}(\mathbf{x})$ is given by $E_g(\mathcal{D}) = \left\langle \left(f^*(\mathbf{x}) - \bar{f}(\mathbf{x}) \right)^2 \right\rangle_{\tilde{p}(\mathbf{x})}$. We note that this quantity is a random variable whose value depends on the sampled training dataset. We calculate its average over the distribution of all datasets with sample size P:

$$E_g = \left\langle \left(f^*(\mathbf{x}) - \bar{f}(\mathbf{x}) \right)^2 \right\rangle_{\tilde{p}(\mathbf{x}), \mathcal{D}} \,. \tag{2}$$

As P grows, we expect fluctuations around this average to fall and $E_g(\mathcal{D})$ to concentrate around E_g. We will demonstrate this concentration in simulations.

2.2 Overview of the calculation

We calculate E_g using the replica method from statistical physics of disordered systems [13, 19]. Details of calculations are presented in the SI Section.1. Here we give a short overview. Readers may choose to skip this part to proceed to the main result in Section 2.3.

Our goal is to calculate the dataset averaged estimator $\langle f^*(\mathbf{x}) \rangle_{\mathcal{D}}$ and its covariance $\left\langle \left(f^*(\mathbf{x}) - \langle f^*(\mathbf{x}) \rangle_{\mathcal{D}} \right) \left(f^*(\mathbf{x}') - \langle f^*(\mathbf{x}') \rangle_{\mathcal{D}} \right) \right\rangle_{\mathcal{D}}$ which depend only on the eigenvalues and features of the kernel, target function and the size of the training set P. From these quantities, we can reconstruct E_g using the bias-variance decomposition of $E_g = B + V$, where $B = \int d\mathbf{x}\, \tilde{p}(\mathbf{x}) \left(\left\langle f^*(\mathbf{x}) \right\rangle_{\mathcal{D}} - \bar{f}(\mathbf{x}) \right)^2$ and $V = \left\langle \int d\mathbf{x}\, \tilde{p}(\mathbf{x}) \left(f^*(\mathbf{x}) - \left\langle f^*(\mathbf{x}) \right\rangle_{\mathcal{D}} \right)^2 \right\rangle_{\mathcal{D}}$.

For this purpose, it will be convenient to work with a basis in the RKHS defined by Mercer's theorem [20]. One can find a (possibly infinite dimensional) complete orthonormal basis $\{\phi_\rho\}_{\rho=1}^M$ for $L^2(\mathbb{R}^D)$ with respect to the training probability distribution $p(\mathbf{x})$ due to Mercer's theorem [20] such that

$$K(\mathbf{x}, \mathbf{x}') = \mathbf{\Phi}(\mathbf{x})^\top \mathbf{\Lambda} \mathbf{\Phi}(\mathbf{x}'), \quad \int d\mathbf{x}\, p(\mathbf{x}) \mathbf{\Phi}(\mathbf{x}) \mathbf{\Phi}(\mathbf{x})^\top = \mathbf{I}, \tag{3}$$

where we defined $M \times M$ diagonal eigenvalue matrix $\mathbf{\Lambda}_{\rho\gamma} = \eta_\rho \delta_{\rho\gamma}$ and the column vector $\mathbf{\Phi}(\mathbf{x}) = \big(\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_M(\mathbf{x})\big)$. Eigenvalues and eigenfunctions satisfy the integral eigenvalue equation

$$\int d\mathbf{x}' p(\mathbf{x}') K(\mathbf{x}, \mathbf{x}') \phi_\rho(\mathbf{x}') = \eta_\rho \phi_\rho(\mathbf{x}). \tag{4}$$

We note that the kernel might not express all eigenfunctions if the corresponding eigenvalues vanish. Here we assume that all kernel eigenvalues are non-zero for presentation purposes, however in SI Section.1 we consider the full case. We also define a feature map via the column vector $\mathbf{\Psi}_\rho(\mathbf{x}) \equiv (\mathbf{\Lambda}^{1/2}\mathbf{\Phi}(\mathbf{x}))_\rho = \sqrt{\eta_\rho}\phi_\rho(\mathbf{x})$ so that $\langle \mathbf{\Psi}(\mathbf{x}), \mathbf{\Psi}_\gamma(\mathbf{x})\rangle_{\mathcal{H}} = \mathbf{I}$. The complete basis $\mathbf{\Phi}(\mathbf{x})$ can be used to decompose the target function:

$$\bar{f}(\mathbf{x}) = \bar{\mathbf{a}}^\top \mathbf{\Phi}(\mathbf{x}) = \bar{\mathbf{w}}^\top \mathbf{\Psi}_\rho(\mathbf{x}), \tag{5}$$

where $\bar{w} = (\mathbf{\Lambda}^{-1/2}\bar{\mathbf{a}})$, and $\bar{\mathbf{a}}$ and $\bar{\mathbf{w}}$ are vectors of coefficients. A function belongs to the RKHS $\mathcal{H}$ if it has finite Hilbert norm $\langle f, f\rangle^2_{\mathcal{H}} = \mathbf{a}^\top \mathbf{\Lambda}^{-1}\mathbf{a} < \infty$.

With this setup, denoting the estimator as $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{\Psi}(\mathbf{x})$, estimation of a target function $\bar{f}(\mathbf{x})$ as given in Eq.(5), kernel regression problem reduces to minimization of the energy function $H(\mathbf{w}; \mathcal{D}) \equiv \frac{1}{2\lambda}\sum_{\mu=1}^{P}\left(\sum_{\rho=1}^{N}(\bar{w}_\rho - w_\rho)\psi_\rho(\mathbf{x}^\mu) + \epsilon^\mu\right)^2 + \frac{1}{2}\|\mathbf{w}\|_2^2$. with optimal estimator weights $\mathbf{w}^*(\mathcal{D}) = \arg\min_{\mathbf{w}} H(\mathbf{w}; \mathcal{D})$.We emphasize the dependence of the estimator to the particular choice of training data $\mathcal{D}$.

We map this problem to statistical mechanics by defining a Gibbs distribution $\propto e^{-\beta H(\mathbf{w};\mathcal{D})}$ over estimator weights $\mathbf{w}$ which concentrates around the kernel regression solution $\mathbf{w}^*$ as $\beta \to \infty$. This can be used to calculate any function $O(\mathbf{w}^*; \mathcal{D})$ by the following trick:

$$O(\mathbf{w}^*; \mathcal{D}) = \lim_{\beta\to\infty} \frac{\partial}{\partial J} \log Z[J; \beta, \mathcal{D}]\big|_{J=0}, \quad Z[J; \beta, \mathcal{D}] = \int d\mathbf{w} e^{-\beta H(\mathbf{w};\mathcal{D}) + JO(\mathbf{w})}. \tag{6}$$

Therefore, the dataset average $\langle O(\mathbf{w}^*; \mathcal{D})\rangle_{\mathcal{D}}$ requires averaging the logarithm of the partition function $\langle \log Z\rangle_{\mathcal{D}}$ which we perform using replica method from statistical physics of disordered systems [13]. A crucial step in our computation is approximating $\mathbf{w}^\top \mathbf{\Psi}(\mathbf{x})$ as a Gaussian random variable via its first and second moments when averaged over the training distribution $p(\mathbf{x})$. It has been shown that this approximation yields perfect agreement with experiments [21, 14, 15].

2.3 Main Result

The outcome of our statistical mechanics calculation (SI Section.1) which constitutes our main theoretical result is presented in the following proposition.

Proposition 1. *Consider the kernel regression problem outlined in Section 2.1, where the model is trained on $p(\mathbf{x})$ and tested on $\tilde{p}(\mathbf{x})$. Consider the Mercer decomposition of the RKHS kernel $K(\mathbf{x}, \mathbf{x}') = \mathbf{\Phi}(\mathbf{x})^\top \mathbf{\Lambda}\mathbf{\Phi}(\mathbf{x}')$, where we defined $M \times M$ (M possibly infinite) diagonal eigenvalue matrix $\mathbf{\Lambda}_{\rho\gamma} = \eta_\rho\delta_{\rho\gamma}$ and the column vector $\mathbf{\Phi}(\mathbf{x}) = \big(\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_M(\mathbf{x})\big)$, with $\int d\mathbf{x}\, p(\mathbf{x})\mathbf{\Phi}(\mathbf{x})\mathbf{\Phi}(\mathbf{x})^\top = \mathbf{I}$. Also consider an expansion of the target function in the Mercer basis $\bar{f}(\mathbf{x}) = \bar{\mathbf{a}}^\top \mathbf{\Phi}(\mathbf{x})$.*

The dataset averaged out-of-distribution generalization error is given by:

$$E_g = E_g^{0,p(\mathbf{x})} + \frac{\gamma' - \gamma}{1-\gamma}\varepsilon^2 + \kappa^2 \bar{\mathbf{a}}^\top (P\mathbf{\Lambda} + \kappa\mathbf{I})^{-1}\mathscr{O}'(P\mathbf{\Lambda} + \kappa\mathbf{I})^{-1}\bar{\mathbf{a}},$$
$$\kappa = \lambda + \kappa\,\mathrm{Tr}\,(P + \kappa\mathbf{\Lambda}^{-1})^{-1}, \quad \gamma = P\,\mathrm{Tr}\,(P + \kappa\mathbf{\Lambda}^{-1})^{-2}, \quad \gamma' = P\,\mathrm{Tr}\,\mathscr{O}(P + \kappa\mathbf{\Lambda}^{-1})^{-2}, \tag{7}$$

where κ must be solved self-consistently, and we defined the $M \times M$ overlap matrix

$$\mathscr{O}_{\rho\gamma} = \int d\mathbf{x}\, \tilde{p}(\mathbf{x})\phi_\rho(\mathbf{x})\phi_\gamma(\mathbf{x}), \quad \mathscr{O}' = \mathscr{O} - \frac{1-\gamma'}{1-\gamma}\mathbf{I}. \tag{8}$$

Here $E_g^{0,p(\mathbf{x})}$ denotes the generalization error when both training and test distributions are matched to $p(\mathbf{x})$ and is given by:

$$E_g^{0,p(\mathbf{x})} = \frac{\gamma}{1-\gamma}\varepsilon^2 + \frac{\kappa^2}{1-\gamma}\bar{\mathbf{a}}^\top (P\mathbf{\Lambda} + \kappa\mathbf{I})^{-2}\bar{\mathbf{a}}, \tag{9}$$

which coincides with the findings of [14, 15]. Further, the expected estimator is:

$$\langle f^*(\mathbf{x}; P)\rangle_{\mathcal{D}} = \sum_{\rho} \frac{P\eta_\rho}{P\eta_\rho + \kappa} \bar{a}_\rho \phi_\rho(\mathbf{x}). \tag{10}$$

Several remarks are in order.

Remark 1. Our result is general in that it applies to any kernel, data distribution and target function. When applied to kernels arising from the infinite width limit of neural networks [11], the information about the architecture of the network will be in the kernel spectrum $\mathbf{\Lambda}$.

Remark 2. Formally, the replica computation requires a thermodynamic limit in which $P \to \infty$, where variations in E_g due to the sampling of the training set become negligible. The precise nature of the limit depends on the kernel and the data distribution, and includes scaling of other variables such as D and M with P. We will give examples of such limits. However, we observe in simulations that our formula predicts average learning curves accurately for even as low as a few samples.

Remark 3. We recover the result obtained in [14, 15] when the training and test distributions are the same ($\mathscr{O} = \mathbf{I}$ which implies $E_g = E_g^{0,p(\mathbf{x})}$).

Remark 4. *Mismatched training and test distributions may improve test error.* Central to our analysis is the shifted overlap matrix $\mathscr{O}'$ which may have negative eigenvalues and hence cause better generalization performance when compared to in-distribution testing/training.

Remark 5. The estimator f^* only depends on the training distribution and our theory predicts that the estimator eventually approaches to the target $f^* \to \bar{f}$ for large enough P which performs perfectly on the training distribution unless there are out-of-RKHS components in target function. The latter case is studied in depth in SI Section.1.

Remark 6. While stated for a scalar output, our result can be trivially generalized to vector output by simply adding the independent error due to each component of the output vector, as described in [14].

Next we analyze this result by studying various examples.

3 Applications to Real Datasets

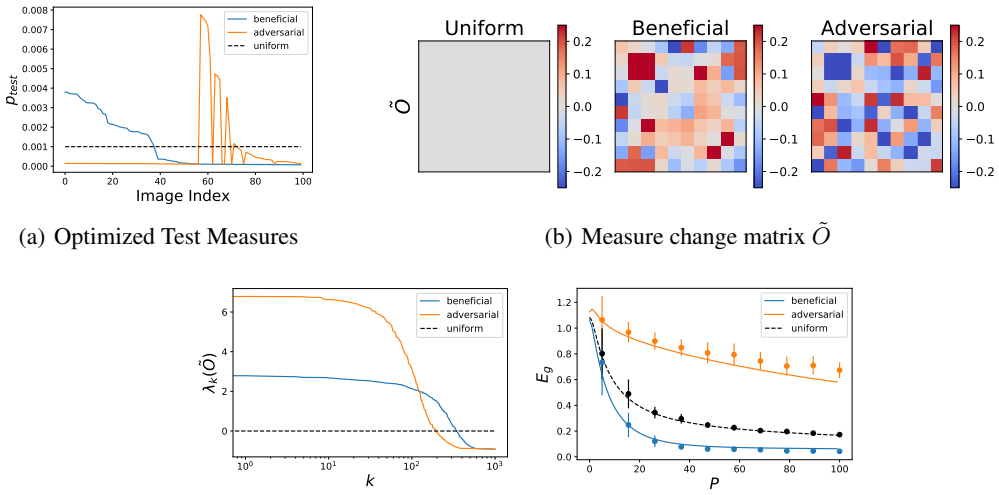
First, we test our theory on real datasets and demonstrate its broad applicability. To do that, we define Dirac training and test probability measures with some point mass on each of the data points $p(\mathbf{x}) = \sum_{\mu=1}^{P} p^\mu \delta(\mathbf{x} - \mathbf{x}^\mu)$ and $p(\mathbf{x}) = \sum_{\mu=1}^{P} \tilde{p}^\mu \delta(\mathbf{x} - \mathbf{x}^\mu)$. With this measure, the kernel eigenvalue problem Eq.(4) becomes $\boldsymbol{K}\mathrm{diag}(\boldsymbol{p})\mathbf{\Phi} = \mathbf{\Phi}\mathbf{\Lambda}$. Once the eigenvalues $\mathbf{\Lambda}$ and eigenvectors $\mathbf{\Phi}^\top$ have been identified, we compute the target function coefficients by projecting the target data $\mathbf{y}_c$ onto these principal components $\overline{\mathbf{w}}_\mathbf{c} = \mathbf{\Phi}\mathbf{y}_c$ for each target $c = 1, \ldots, C$. Once all of these ingredients are obtained, theoretical learning curves can be computed using Proposition 1.

3.1 Shift in test distribution may help or hurt generalization

To analyze the influence of distribution shift on generalization error, we first study the problem of a fixed training distribution and a shifting test distribution. While one may naively expect that a shift in the test distribution would always result in worse performance, we demonstrate that mismatch can be either beneficial or detrimental to generalization. A similar point was made in [3].

Since our generalization error formula is a differentiable function with respect to the training measure, we develop a routine to numerically optimize the generalization error with respect to a test measure using gradient based optimization. We numerically optimize E_g with respect to $\{\tilde{p}^\mu\}$ using gradient descent to find beneficial test measures where the error is smaller than for the case where training and test measures are matched. Similarly, we perform gradient ascent to find adversarial test measures, which give higher test error than the matched case.

The result of this procedure is shown in Figure 1. We perform gradient descent (labeled beneficial) and gradient ascent (labeled adversarial) on $M = 1000$ MNIST images, where target outputs are one-hot vectors of digit labels, with a depth 3 ReLU fully-connected Neural Tangent Kernel. We compute the kernels using the Neural Tangents API [22]. The probability mass assigned to the first 50 points are provided in (a). We see that points given high mass for the beneficial test distribution are given low probability mass for the adversarial test distribution and vice versa. We plot the measure



(a) Optimized Test Measures

(b) Measure change matrix $\tilde{O}$

(c) Spectrum of $\tilde{O}$

(d) Learning Curves

Figure 1: Shifts in the test distribution can be understood through the matrix $\tilde{O}$. (a) We ran gradient descent (beneficial) and gradient ascent (adversarial) on the theoretical generalization error E_g with respect to the test measure on 1000 MNIST images. (a) The final probability of the first 100 images sorted by probability mass for the beneficial measure. (b) For these three distributions (uniform, beneficial, adversarial), the measure change matrices $\tilde{O}$ are plotted for the top 10 modes. The beneficial and adversarial matrices are roughly opposite. (c) The spectrum of $\tilde{O}$ reveals both negative and positive eigenvalues which demonstrates that increase or a decrease in the generalization error is possible. (d) The learning curves show the predicted generalization error (lines) along with some experiments (dots). Error bars indicate standard deviation over 35 random trials.

change matrix $\tilde{\mathbf{O}}$ which quantifies the change in the generalization error due to distribution shift. Recall that in the absence of noise,E_g is of the form $E_g = E_g^{\text{matched}} + \boldsymbol{v}^\top \tilde{\mathbf{O}} \boldsymbol{v}$ for a vector $\boldsymbol{v}$ which depends on the task, kernel, training measure and sample size P. Depending on the eigenvalues and eigenvectors of $\tilde{\mathbf{O}}$, and the vector $\boldsymbol{v}$, the quadratic form $\boldsymbol{v}^\top \tilde{\mathbf{O}} \boldsymbol{v}$ may be positive or negative, resulting in improvement or detriment in the generalization error. In (b), we see that the $\tilde{\boldsymbol{O}}$ matrix for the beneficial test measure appears roughly opposite that of the adversarial measure. This is intuitive since it would imply the change in the generalization error to be opposite in sign for beneficial and adversarial measures $\boldsymbol{v}^\top \left(-\tilde{\boldsymbol{O}}\right) \boldsymbol{v} = -\boldsymbol{v}^\top \tilde{\boldsymbol{O}} \boldsymbol{v}$. The spectrum of the matrix, shown in (c), reveals that it possesses both positive and negative eigenvalues. We plot the theoretical and experimental leanring curves in (d). As promised, the beneficial (blue) measure has lower generalization error while the adversarial measure (orange) has higher generalization error than the matched case (black). Experiments show excellent match to our theory.

3.2 Optimized Training Measure for Digit Classification with Wide Neural Networks

Often in real life tasks test distribution is fixed while one can alter the training distribution for more efficient learning. Therefore, we next study how different training distributions affect the generalization performance of kernel regression when the test distribution is fixed. We provide pseudocode in Algoritm 1 which describes a procedure to optimize the expected generalization error with respect to a training distribution of P data points. All of the operations in the computation of E_g support automatic differentiation with respect to logits $\boldsymbol{z} \in \mathbb{R}^M$ defining the training distribution, allowing us to perform gradient descent on the training measure [23].

As an example, we fix the test distribution to be uniform MNIST digits for 8's and 9's and optimize over the probability mass of each unique digit to minimize E_g, again using a depth 3 ReLU neural tangent kernel and one-hot target vectors. Our results indicate that it is possible to reduce generalization when the training distribution is chosen differently than the uniform test distribution. In Figure 2,

Algorithm 1: Optimizing Training Measure at sample size P

Result: GET_LOSS($\boldsymbol{z} \in \mathbb{R}^M$, $\boldsymbol{K} \in \mathbb{R}^{M\times M}$, $\boldsymbol{y} \in \mathbb{R}^M$, $\lambda \in \mathbb{R}_+$)
  Compute normalized distribution $\boldsymbol{p} = \text{softmax}(\boldsymbol{z})$;
  Diagonalize on Train Measure $\boldsymbol{K}\text{diag}(\boldsymbol{p})\boldsymbol{\Phi} = \boldsymbol{\Phi}\boldsymbol{\Lambda}$ with $\boldsymbol{\Phi}^\top \text{diag}(\boldsymbol{p})\boldsymbol{\Phi} = \boldsymbol{I}$;
  Get Target Function Weights $\bar{\mathbf{a}} = \boldsymbol{\Phi}^\top \text{diag}(\boldsymbol{p})\boldsymbol{y}$;
  Get Overlap Matrix $\mathcal{O} = \frac{1}{M}\boldsymbol{\Phi}^\top\boldsymbol{\Phi}$;
  Solve Implicit Equation $\kappa = \text{ODE-INT}\left[\dot{\kappa} = \lambda + \kappa \sum_k \frac{\Lambda_{kk}}{\Lambda_{kk}P+\kappa} - \kappa\right]$;
  return $E_g(\kappa, P, \boldsymbol{\Lambda}, \mathcal{O})$ (see Equation 1);
;
Result: OPT_MEASURE($\boldsymbol{K} \in \mathbb{R}^{M\times M}$, $\boldsymbol{y} \in \mathbb{R}^M$, P, T ,η, λ)
  Initialize $\boldsymbol{z} = \mathbf{0} \in \mathbb{R}^M$, $t = 0$;
  Diagonalize Kernel on Uniform Measure $\frac{1}{M}\boldsymbol{K} = \tilde{\boldsymbol{\Phi}}\tilde{\boldsymbol{\Lambda}}\tilde{\boldsymbol{\Phi}}^\top$;
  while $t < T$ **do**
    $\boldsymbol{z} = \boldsymbol{z} - \eta\,\text{GRAD}[\text{GET_LOSS}(\boldsymbol{z}, \boldsymbol{K}, \boldsymbol{y}, \lambda)]$;
    $t = t + 1$;
  end
  return $\text{softmax}(\boldsymbol{z})$;

we optimize E_g for $P = 30$ training samples to extract the optimal training distribution on NTK. We observe that the high mass training digits are closer to the center of their class, Figure 2 (e). However we also find that optimizing the distribution for different values of P may cause worse generalization beyond the P used to optimize the distribution Figure 2 (g) and (h).

We also test our results on neural networks, exploiting a correspondence between kernel regression with the Neural Tangent Kernel, and training infinitely wide neural networks [11]. Figure 2 (i) shows that our theory matches experiments with ReLU networks of modest width (2000).

4 Linear Regression: An Analytically Solvable Model

Next we study linear regression to demonstrate various interesting OOD generalization phenomena. Consider a linear target function $\bar{f}(\mathbf{x}) = \beta^\top \mathbf{x}$ where $\mathbf{x}, \beta \in \mathbb{R}^D$ and a linear kernel $K(\mathbf{x}, \mathbf{x}') = \frac{1}{D}\mathbf{x}^\top\mathbf{x}'$ (in this model $M = D$). Data are sampled from zero-mean Gaussians with arbitrary covariance matrices $\mathbf{C}$ and $\tilde{\mathbf{C}}$ for training and test distributions, respectively.

In this case, the kernel eigenvalue equation can be solved exactly. Denoting the eigenvalues and eigenvectors of the covariance matrix of the training distribution as $\mathbf{C}\mathbf{U} = \mathbf{U}\boldsymbol{\Sigma}$ where $\boldsymbol{\Sigma}_{\rho\gamma} = \sigma_\rho^2\delta_{\rho\gamma}$ is the diagonal eigenvalue matrix and $\mathbf{U}$ is an orthogonal matrix with columns being eigenvectors of $\mathbf{C}$, the integral eigenvalue problem becomes $\eta_\rho\phi_\rho(\mathbf{x}) = \langle K(\mathbf{x}, .), \phi_\rho(.)\rangle_{p(\mathbf{x})} = \frac{\sigma_\rho}{D}\mathbf{u}_\rho^\top\mathbf{x}$. Therefore, the normalized eigenfunctions are $\phi_\rho(\mathbf{x}) = \mathbf{u}_\rho^\top\mathbf{x}/\sigma_\rho$ and the eigenvalues are $\eta_\rho = \sigma_\rho^2/D$. The overlap matrix is

$$\mathscr{O}_{\rho\gamma} = \langle \phi_\rho(.), \phi_\gamma(.)\rangle_{\tilde{p}(\mathbf{x})} = \boldsymbol{\Sigma}^{-1/2}\mathbf{U}^\top\tilde{\mathbf{C}}\mathbf{U}\boldsymbol{\Sigma}^{-1/2}. \tag{11}$$

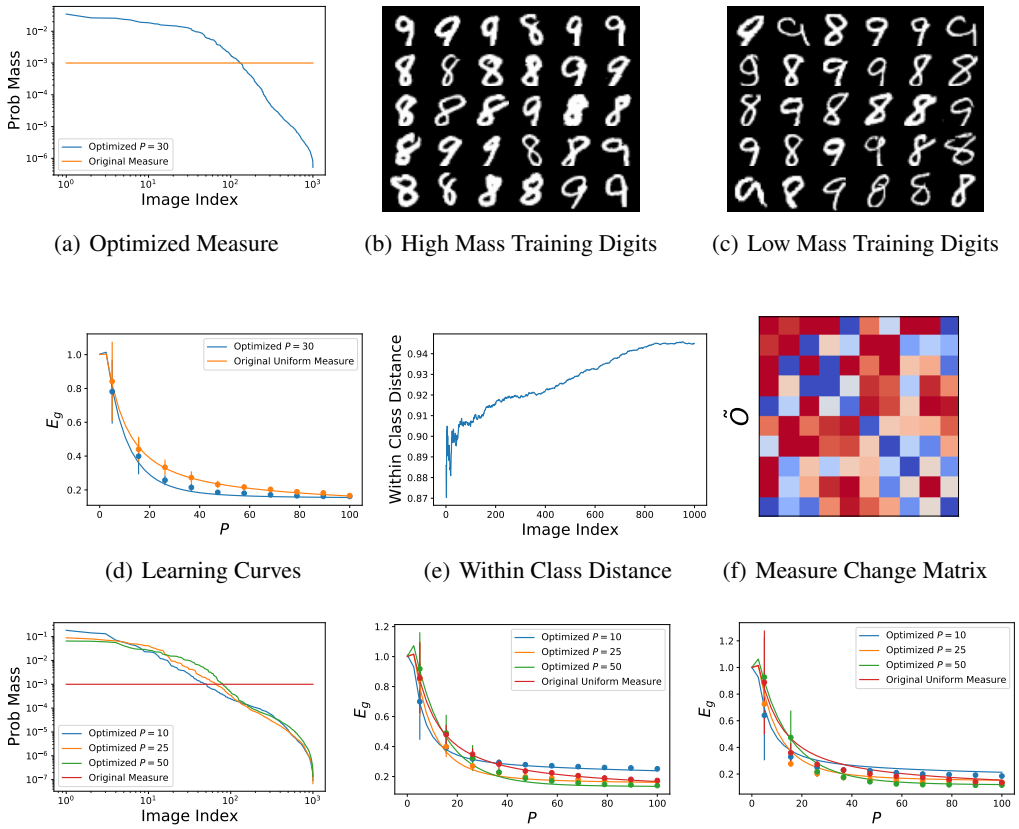
Finally computing the target weights as $\mathbf{a} = \boldsymbol{\Sigma}^{1/2}\mathbf{U}^\top\beta$, we obtain the generalization error Eq.(1):

$$E_g = E_g^{0,p(\mathbf{x})} + \frac{\gamma' - \gamma}{1-\gamma}\varepsilon^2 + (\kappa D)^2\beta^\top(P\mathbf{C} + \kappa D\mathbf{I})^{-1}\left(\tilde{\mathbf{C}} - \frac{1-\gamma'}{1-\gamma}\mathbf{C}\right)(P\mathbf{C} + \kappa D\mathbf{I})^{-1}\beta,$$
$$\kappa = \lambda + \kappa\,\text{Tr}\,\mathbf{C}(P\mathbf{C} + \kappa D\mathbf{I})^{-1}, \quad \gamma = P\,\text{Tr}\,\mathbf{C}^2(P\mathbf{C} + \kappa D\mathbf{I})^{-2}, \quad \gamma' = P\,\text{Tr}\,\tilde{\mathbf{C}}\mathbf{C}(P\mathbf{C} + \kappa D\mathbf{I})^{-2}, \tag{12}$$

and

$$E_g^{0,p(\mathbf{x})} = \frac{\gamma}{1-\gamma}\varepsilon^2 + \frac{(\kappa D)^2}{1-\gamma}\beta^\top\mathbf{C}(P\mathbf{C} + \kappa D\mathbf{I})^{-2}\beta. \tag{13}$$

As a consistency check, we note that the generalization error is minimized ($E_g = 0$) when $\tilde{\mathbf{C}} = 0$ corresponding to a Dirac measure at the origin. This makes sense since target function at origin is 0 and the estimator on the test distribution is also 0.



(a) Optimized Measure (b) High Mass Training Digits (c) Low Mass Training Digits

(d) Learning Curves (e) Within Class Distance (f) Measure Change Matrix

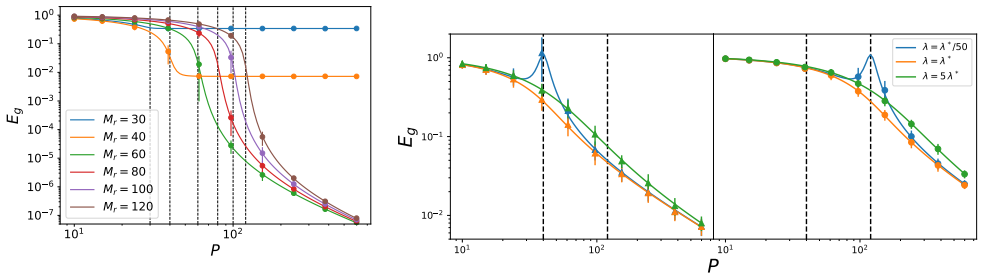
(g) Optimized Measures (h) Kernel Regression (i) Width 2000 Neural Network

Figure 2: We study regression with a uniform test distribution on MNIST 8's and 9's. For this fixed task, we optimize our theoretical generalization error expression over training distributions at $P = 30$. This gives a probability distribution over MNIST digits shown in (a). The 30 images with highest probability mass (b) appear qualitatively more representative of handwritten 8's and 9's than those given the lowest probability mass (c). We plot the theoretical (solid) and experimental (dots) learning curves for the optimized and original uniform training measure, showing that changing the sampling strategy can improve generalization. Error bars display standard deviation over 30 repeats. (e) After ordering each point by their probability mass on the optimized measure, we calculate the average feature space (in the sense of the kernel) distance to all other points from the same class. This measure rises with image index, indicating that images with higher probability are approximately centroids for each class. (f) The optimized measure induces a non-zero measure change matrix $\tilde{\boldsymbol{O}}$. (g) The optimized probability distributions have different shapes for different training budget sizes P, with flatter distributions at larger P. (h) Kernel regression experiments agree with theory for each of these measures. A measure which performs best at low sample sizes may give sub-optimal generalization at larger sample sizes. (i) The theory also approximates learning curves for finite width 2000 fully connected neural networks with depth 3, initilialized with NTK initialization [11].

Next, we consider a diagonal covariance matrix $\tilde{\mathbf{C}} = \tilde{\sigma}^2 \mathbf{I}$ for test distribution and

$$\mathbf{C} = \text{diag}\,(\underbrace{\sigma^2, \dots \sigma^2}_{M_r}, \underbrace{0, \dots 0}_{D - M_r}) \tag{14}$$

for training distribution to demonstrate how training distribution affects generalization in a simplified setting. Furthermore we take the target function to have power only in the first N features ($\beta_{\rho > N} = 0$) where we adopt to the normalization $\sum_{\rho=1}^{N} \beta_\rho^2 = 1$. The target does not depend on directions $x_{\rho > N}$ and we study how compressing some directions in training distribution helps generalization. In this



(a) Varying Training Measure Dimension (b) Effect of Training Measure

Figure 3: (a) Learning curves for linear regression on a target $\bar{f} = \beta^\top \mathbf{x}$ where $\beta_{\rho \leq 40} \sim \mathcal{N}(0,1)$, $\beta_{60>\rho>40} \sim \mathcal{N}(0, 0.01)$ and $\beta_{\rho>60} = 0$, hence $N = 60$. Input dimension, label noise and ridge parameter are $D = 120$, $\varepsilon^2 = 0$ and $\lambda = 10^{-3}$. Altering the training measure dimensionality below or above $M_r = 60$ hurts generalization. For all curves $E_g \to 0$ as $P \to \infty$ except the blue and orange lines at $M_r = 30, 40$ where there is irreducible error since some portion of the target coefficients are not learned. (b) Same experiment with $\varepsilon^2 = 0.1$ label noise and with varying ridge parameters λ. Target coefficients are $\beta_{\rho \leq 40} \sim \mathcal{N}(0,1)$ and $\beta_{\rho>40} = 0$ ($N = 40$). The variances of training and test distributions are $\sigma^2 = \tilde{\sigma}^2 = 1$. Left panel and right panels are the learning curves for $M_r = 40, 120$, respectively. Dashed vertical lines indicate the number $P = M_r = 40, 120$. Dots are experiment, lines are theory. Errorbars represent standard deviation over 30 trials.

case, the self-consistent equation for κ becomes exactly solvable. The generalization error reduces to:

$$E_g = \tilde{\sigma}^2 \left(\frac{\varepsilon^2}{\sigma^2} \frac{\alpha}{(\kappa' + \alpha)^2 - \alpha} + \frac{\kappa'^2}{(\kappa' + \alpha)^2 - \alpha} \sum_{\rho=1}^{M_r} \beta_\rho^2 + \sum_{\rho=M_r+1}^{N} \beta_\rho^2 \right), \tag{15}$$

where we defined $\kappa' = \frac{\kappa}{\sigma^2 M_r / D} = \frac{1}{2}\left[(1 + \tilde{\lambda} - \alpha) + \sqrt{(1 + \tilde{\lambda} + \alpha)^2 - 4\alpha}\right]$, $\tilde{\lambda} = \frac{\lambda}{\sigma^2 M_r / D}$ and $\alpha = P/M_r$. This result matches that of [15] analyzing in-distribution generalization error when $M_r = N = D$ and $\sigma^2 = \tilde{\sigma}^2$. We identify $\tilde{\lambda}$ as an *effective regularization* parameter as it assumes the role λ plays for in-distribution generalization (compare to Eq. 7 of [15]).

First, we note that the generalization linearly scales with the variance of the test distribution. This is due to the linearity of the target and the estimator; any mismatch between them is amplified when the test points are further away from the origin.

Next, when $M_r < N$ there is an irreducible error due to the fact that kernel machine cannot fit the directions which are not expressed in the training data (Figure 3a). However in certain cases choosing $M_r < N$ may help generalization if the data budget is limited to $P \sim M_r$ since the learning happens faster compared to larger M_r. As illustrated in Figure 3a, if the target power is distributed such that $\sum_{\rho=N'+1}^{N} \beta_\rho^2 \ll \sum_{\rho=1}^{N'} \beta_\rho^2$, choosing $M_r = N'$ can be a better strategy despite the irreducible error due to unexplored target power. In Figure 3a, we picked $N = 60$ and $N' = 40$ which is the number of directions target places most of its power on. If the data budget was limited to $P < N$, choosing $M_r = N'$ (orange curve in Figure 3a) performs better than $M_r = N$ (green curve) although for $P \geq N$ there is no irreducible error. Reducing M_r below N' (blue curve in Figure 3a) on the other hand does not bring much difference.

Next, we observe that κ' is a monotonically decreasing function of α and shows a sharp decrease near $\alpha \approx 1 + \tilde{\lambda}$ which also implies a sharp change in generalization error. In fact when $\tilde{\lambda} = 0$, κ' becomes zero at $\alpha = 1$ and its first derivative diverges implying either a divergence in generalization error due to label noise (called sample-wise double-descent [18, 24, 25, 26, 27, 28, 29, 30, 31, 32, 15]) or vanishing of E_g in the absence of noise (SI Section.2). Hence reducing M_r means that this transition occurs earlier in the learning curve ($P \sim (1 + \tilde{\lambda}) M_r$) which implies that less training samples are necessary to estimate the target function well. Intuitively, sampling from an effectively higher dimensional space increases the necessary amount of training data to fit the modes in all directions. In Figure 3(b), we demonstrate this prediction for a linear target and obtain perfect agreement with experiment.

Finally, we show that increasing effective regularization $\tilde{\lambda}$, which is controlled by the training distribution (σ^2, M_r), leads to the suppression of double-descent peak, hence avoiding possible non-monotonicities. However large $\tilde{\lambda}$ leads slower learning creating a trade-off. Results from previous studies [29, 15] have shown the existence of an optimal ridge parameter in linear regression minimizing the generalization error for all P. In our setting, we find that $\tilde{\lambda} = \varepsilon^2/\sigma^2$ is the optimal ridge parameter (SI Section.2) and is given by:

$$\lambda^* = \frac{M_r}{D}\varepsilon^2. \tag{16}$$

In Figure 3(b), we show that choosing optimal ridge parameters indeed mitigates the double descent and results in the best generalization performance.

Other analytically solvable models: In SI, we present other analytically solvable models.

- (SI Section.2) We study a more general linear regression model with diagonal overlap matrix where train/test distributions, target function and number of directions kernel expresses vary. When the target has out-of-RKHS components, we show how distribution shifts may help in generalization.
- (SI Section.3) We apply our theory to rotation invariant kernels such as the Gaussian RBF kernel and NTK [11] acting on spherical data in high-dimensions. We examine how a mismatched sphere radius affects the generalization error in the limit $P, D \to \infty$ similar to the case solved in [14, 15].
- (SI Section.4) We study interpolation versus extrapolation in linear regression and regression in Fourier space by studying rectangular distributions with different ranges for one-dimensional inputs.

5 Discussion

The interest in kernel methods has recently surged due to their profound connections to deep neural networks [11, 12, 33, 34] and replica method of statistical physics proved useful in studying its generalization properties, inductive biases and non-monotonic learning curves [14, 15, 21, 35, 36, 37, 30, 38]. Along the lines of these research, we analyzed generalization performance of kernel regression under distribution mismatch and obtained analytical expressions which match experiments perfectly including those with wide neural networks. We considered several analytically solvable models to show how learning rate with training set size changes with training distribution and cases where choosing it different than test distribution helps. We demonstrated that our formula can be used to optimize over training distribution on MNIST which revealed that certain digits are better to sample more often than others. We applied our results to linear regression and demonstrated how dimensional reduction of the training distribution can be helpful in generalization, including shifting of a double-descent peak. Our theory brings many insights about how kernel regression generalizes to unseen distributions.

Previous theoretical studies on OOD generalization mostly consider minimizing a more general empirical risk which takes into account possible perturbations of the training distribution [39] or requires the solution to be the minima across all possible environments [17, 16]. Unlike these line of work, we study only how a specific empirical risk minimization algorithm, kernel regression, performs under OOD generalization. The advantage of kernel regression was the analytical handle it provides and the ability to obtain precise predictions instead of worst case bounds. We note that ideas such as Invariant Risk Minimization (IRM) [17] have been also shown to be possible in random features and kernel models [40].

Limitations: While our theory accurately describes experiments, it has limitations. First, using it to optimize for a training distribution requires the knowledge of exact test distribution which could not be available at the time of training. Second, the theory requires an eigendecomposition of the kernel which is computationally costly for large datasets. Third, our theory uses the replica theory [13], which is not fully rigorous. Fourth, if to be used to describe neural networks, our theory's applicability is limited to their kernel regime.

References

[1] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[2] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT Press, 2018.

[3] Carlos R González and Yaser S Abu-Mostafa. Mismatched training and test distributions can outperform matched ones. *Neural computation*, 27(2):365–387, 2015.

[4] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

[5] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. Exploring the landscape of spatial robustness. In *International Conference on Machine Learning*, pages 1802–1811. PMLR, 2019.

[6] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019.

[7] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[8] Grace Wahba. *Spline models for observational data*. SIAM, 1990.

[9] Theodoros Evgeniou, Massimiliano Pontil, and Tomaso Poggio. Regularization networks and support vector machines. *Advances in computational mathematics*, 13(1):1, 2000.

[10] John Shawe-Taylor, Nello Cristianini, et al. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[11] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

[12] Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. In *International Conference on Machine Learning*, pages 541–549, 2018.

[13] Marc Mézard, Giorgio Parisi, and Miguel Virasoro. *Spin glass theory and beyond: An Introduction to the Replica Method and Its Applications*, volume 9. World Scientific Publishing Company, 1987.

[14] Blake Bordelon, Abdulkadir Canatar, and Cengiz Pehlevan. Spectrum dependent learning curves in kernel regression and wide neural networks. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

[15] Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks. *Nature Communications*, 12(1):1–12, 2021.

[16] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex), 2020.

[17] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization, 2019.

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

[19] Andreas Engel and Christian Van den Broeck. *Statistical mechanics of learning*. Cambridge University Press, 2001.

[20] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[21] Rainer Dietrich, Manfred Opper, and Haim Sompolinsky. Statistical mechanics of support vector networks. *Physical review letters*, 82(14):2975, 1999.

[22] Roman Novak, Lechao Xiao, Jiri Hron, Jaehoon Lee, Alexander A. Alemi, Jascha Sohl-Dickstein, and Samuel S. Schoenholz. Neural tangents: Fast and easy infinite neural networks in python. In *International Conference on Learning Representations*, 2020.

[23] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[24] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. In *International Conference on Learning Representations*, 2019.

[25] A Krogh and J. Hertz. Generalization in a linear perceptron in the presence of noise. *Journal of Physics A: Mathematical and General*, 25:1135, 01 1999.

[26] J A Hertz, A Krogh, and G I Thorbergsson. Phase transitions in simple learning. *Journal of Physics A: Mathematical and General*, 22(12):2133–2150, Jun 1989.

[27] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J. Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *arXiv preprint arXiv:1903.08560*, 2019.

[28] Preetum Nakkiran. More data can hurt for linear regression: Sample-wise double descent. *arXiv preprint arXiv:1912.07242*, 2019.

[29] Preetum Nakkiran, Prayaag Venkat, Sham Kakade, and Tengyu Ma. Optimal regularization can mitigate double descent. *arXiv preprint arXiv:2003.01897*, 2020.

[30] Stéphane d'Ascoli, Levent Sagun, and Giulio Biroli. Triple descent and the two kinds of overfitting: Where and why do they appear? *arXiv preprint arXiv:2006.03509*, 2020.

[31] Tengyuan Liang, Alexander Rakhlin, and Xiyu Zhai. On the multiple descent of minimum-norm interpolants and restricted lower isometry of kernels. volume 125 of *Proceedings of Machine Learning Research*, pages 2683–2711. PMLR, 09–12 Jul 2020.

[32] Lin Chen, Yifei Min, Mikhail Belkin, and Amin Karbasi. Multiple descent: Design your own generalization curve, 2020.

[33] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

[34] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in neural information processing systems*, pages 8570–8581, 2019.

[35] Bruno Loureiro, Cédric Gerbelot, Hugo Cui, Sebastian Goldt, Florent Krzakala, Marc Mézard, and Lenka Zdeborová. Capturing the learning curves of generic features maps for realistic data sets with a teacher-student model. *arXiv preprint arXiv:2102.08127*, 2021.

[36] H. S. Seung, H. Sompolinsky, and N. Tishby. Statistical mechanics of learning from examples. *Phys. Rev. A*, 45:6056–6091, Apr 1992.

[37] M Opper, W Kinzel, J Kleinz, and R Nehl. On the ability of the optimal perceptron to generalise. *Journal of Physics A: Mathematical and General*, 23(11):L581–L586, jun 1990.

[38] Stéphane d'Ascoli, Maria Refinetti, Giulio Biroli, and Florent Krzakala. Double trouble in double descent : Bias and variance(s) in the lazy regime. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

[39] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Princeton university press, 2009.

[40] Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Learning with invariances in random features and kernel models, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] We do not anticipate that our work will have negative societal impacts as outlined in the ethics guidelines, as it is purely theoretical.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] The main computation has been performed in detail in supplemental material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] All code and data will be provided with supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] All details about the computational resources are detailed in supplemental material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]