A Kernel Analysis of Feature Learning in Deep Neural Networks

Abdulkadir Canatar
Center for Computational Neuroscience
Flatiron Institute
New York City, NY, USA
acanatar@flatironinstitute.org

Cengiz Pehlevan
John A. Paulson School of Engineering and Applied Sciences,
and Center for Brain Science
Harvard University
Cambridge, MA, USA
cpehlevan@g.harvard.edu

***Abstract*—Deep neural networks learn useful representations of data, yet the nature of these representations has not been fully understood. Here, we empirically study the kernels induced by the layer representations during training by analyzing their kernel alignment to the network's target function. We show that representations from earlier to deeper layers increasingly align with the target task for both training and test sets, implying better generalization. We analyze these representations across different architectures, optimization methods and batch sizes. Furthermore, we compare the Neural Tangent Kernel (NTK) of deep neural networks and its alignment with the target during training and find that NTK-target alignment also increases during training.**

***Index Terms*—deep learning, kernel methods**

I. Introduction

Often complex tasks such as image classification or natural language processing require transforming the inputs to a different space where their classification is much simpler than their respective input space. While a large number of studies are dedicated to engineering such representations [1], deep neural networks has proven many times to learn better representations to solve such problems [2]. Yet, the complicated dynamics of deep neural networks make the analysis of the features they learn very hard.

A promising approach have gained attention in the last few years. The Neural Tangent Kernel (NTK) introduced in [3] initiated a large body of work on how neural networks learn in the limit where the number of units in the hidden layer are taken to infinity, or the infinite-width limit [4], [5]. Specifically, training an infinitely wide neural network with mean-squared-loss and gradient-flow to interpolation recovers a kernel regression predictor with the NTK specified by the network architecture. Because of the tractability of kernel regression, many insights were gained about generalization properties of deep neural networks in the infinite-width limit, i.e. [3], [6], [7], [8], [9], [10], [11], [12], [13] and more.

For our purposes, a problem with the infinite-width limit of deep neural networks defined in [3] is that in this limit representation learning does not occur: the network features are static and do not adapt to data. Multiple approaches were taken to address this problem. In one line of work, alternative infinite-width limits were defined in which network learn representations from data [14], [15]. Other work used perturbation theory around the infinite-width limit to gain insight into networks at finite-width where representation learning occurs [16], [17], [18], [19], [20], [21], [22], [23]. It still remains to be shown that these approaches explain realistic feature learning scenarios. They are still limited by large-width approximations and may fall short of explaining possible non-perturbative effects at finite width [24], [25].

Several relevant findings that potentially challenge these theories could be noted here. Predictions from perturbative studies imply that the feature kernels of the internal layers of DNNs receive perturbatively small corrections related to the outer product of the outputs on the training set [20], [16], [23], [17]. This is in direct contrast to the recently discovered phenomena of neural collapse [26], where the feature kernel of the penultimate layer becomes the outer product of the outputs. Furthermore, recent work found that the final output of a trained neural network is equivalent to the output of a kernel regression predictor with the empirical NTK obtained at the end, not at the beginning, of the training under certain conditions [27]. Other work showed that the final output can be approximated well by the network's linearization at the later stages of training [28] which is also equivalent to kernel regression with the empirical NTK [3]. These observations motivate our empirical study of feature learning in deep neural networks.

In this work, using several alignment metrics such as kernel-target alignment [29] and task-model alignment [9], we empirically study the feature kernels of the internal layers of a DNN and report our results on how the target function aligns with the internal representations throughout training. We present an empirical survey on the feature learning properties of different architectures trained with different optimizers.

A motivation for the comparison to the target function comes from the kernel regression literature, which suggests that the optimal kernel for a given target function in terms of sample-complexity is proportional to the outer product of the target function with itself [29]. We also explore whether the final NTK approaches to this optimal kernel.

II. Problem Setup

We consider linear probes to the hidden layer features learned by deep neural networks [30] to assess how the emerging representations help solve the underlying task. We study a supervised learning setting with a dataset $\{\mathbf{x}^\mu, \mathbf{y}^\mu\}_{\mu=1}^P$ where $\mathbf{x}^\mu \in \mathcal{X} \subset \mathbb{R}^D$ are $D-$dimensional inputs assumed to be drawn from a probability distribution $d\mu(\mathbf{x})$, and $\mathbf{y}^\mu \in \mathcal{Y} \subset \mathbb{R}^C$ are $C-$dimensional labels assumed to be generated by an underlying target function $\bar{\mathbf{f}}(\mathbf{x})$. We consider L hidden layer deep neural networks of the form $\mathbf{f}(\mathbf{x}) = \mathbf{W}\boldsymbol{\psi}^{(L)}(\mathbf{x})$, where $\boldsymbol{\psi}^{(L)}(\mathbf{x})$ is an N_L-dimensional vector of activations of the final hidden layer features and $\mathbf{W}$ is a $C \times N_L$ matrix of readout weights. Similarly, we denote the activations of other hidden layers with $\boldsymbol{\psi}^{(\ell)}(\mathbf{x})$ with dimension N_ℓ. We only consider neural networks trained with mean-squared-error (MSE) loss.

At any time during training, we assess whether the feature maps for each layer are good representations for the original task by studying the linear regression performance in the feature space. We define a model for each layer $\mathbf{f}^{(\ell)}(\mathbf{x}) = \mathbf{W}^{(\ell)}\boldsymbol{\psi}^{(\ell)}(\mathbf{x})$ where the parameters of the features $\boldsymbol{\psi}^{(\ell)}(\mathbf{x})$ are frozen. Minimizing MSE loss for this model on the training set corresponds to kernel regression [31] with a *layer feature kernel* $K^{(\ell)}(\mathbf{x}, \mathbf{x}') = \boldsymbol{\psi}^{(\ell)}(\mathbf{x})^\top \boldsymbol{\psi}^{(\ell)}(\mathbf{x}')$.

III. Metrics for Representational Alignment and Feature Learning

In machine learning and deep learning literature, several representational measures are considered based on statistical techniques such as Canonical Correlation Analysis (CCA) [32], [33] (see [34] for a review), Partial Least Squares (PLS) regression [35], Kernel Independent Component Analysis (Kernel ICA) [36], Hilbert Schmidt Independence Criterion (HSIC) [37], Kernel-Target Alignment (KTA) [29] and Centered Kernel Alignment [38], [29].

While all these methods are based on *linear probes* on the features, they are qualitatively different. CCA, PLS and Kernel ICA measure cross-correlations between two feature representations and are related to each other [39], [36]. On the other hand, HSIC, KTA and CKA measure population level similarities based on the kernels induced by the inner product of individual features. It has been shown in [40] that CCA ignores the variance scale (eigenvalue) associated to each principal component $\phi_\rho^{(\ell)}$ in analyzing the similarity and treats each of them in equal footing. On the other hand, CKA has been shown to be insensitive to the removal of principal components which have small eigenvalues which are nevertheless important in quantifying representations [41]. While several works used these measures to quantify the representational similarity between learned features in different neural networks [40], [41], [42], [43], [44], the notion of a correct representational similarity metric for comparing the representations of two feature maps remains unclear.

In this work we are only interested in comparing the internal layer representations to the either input (image) or output (label) representations which remain fixed for any architecture. To refine the notion of representational alignment, we define the scalar kernels induced by the input and output representations by $K_{\mathbf{x}}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}'$ and $K_{\bar{\mathbf{f}}}(\mathbf{x}, \mathbf{x}') = \bar{\mathbf{f}}(\mathbf{x})^\top \bar{\mathbf{f}}(\mathbf{x}')$, respectively. Furthermore, we define the *layer feature* kernels by $K^{(\ell)}(\mathbf{x}, \mathbf{x}') = \boldsymbol{\psi}^{(\ell)}(\mathbf{x})^\top \boldsymbol{\psi}^{(\ell)}(\mathbf{x}')$ for each layer. We consider various metrics for measuring representational similarity between two features via their corresponding kernel.

1) **Cumulative Power for Task-Model Alignment (KTA)** Recent works show that the generalization performance of kernel regression can be analytically obtained from the spectral properties of the kernel [8], [9], [10], [13]. The spectral decomposition for layer kernels can be obtained by finding an orthonormal eigenbasis of L^2_μ [45], [31] such that

$$K^{(\ell)}(\mathbf{x}, \mathbf{x}') = \boldsymbol{\psi}^{(\ell)}(\mathbf{x})^\top \boldsymbol{\psi}^{(\ell)}(\mathbf{x}') = \sum_{\rho=1}^{\infty} \eta_\rho^{(\ell)} \phi_\rho^{(\ell)}(\mathbf{x}) \phi_\rho^{(\ell)}(\mathbf{x}'), \tag{1}$$

where $\{\eta_\rho^{(\ell)}\}$ are the eigenvalues, and the eigenbasis $\{\phi_\rho^{(\ell)}\}$ satisfies $\int d\mu(\mathbf{x})\phi_\rho^{(\ell)}(\mathbf{x})\phi_\gamma^{(\ell)}(\mathbf{x}) = \delta_{\rho\gamma}$. Moreover, any L^2_μ target function can be decomposed as

$$\bar{\mathbf{f}}(\mathbf{x}) = \sum_\rho \bar{\mathbf{a}}_\rho^{(\ell)} \phi_\rho^{(\ell)}(\mathbf{x}), \tag{2}$$

where we assumed that each target class $\bar{\mathbf{f}}_i(\mathbf{x})$ is an L^2_μ function for $i = 1, \ldots, C$.
Previous works [8], [9] showed that generalization in kernel regression arises from two factors: 1) *spectral bias* which states hierarchical learning of a target function's projections to kernel eigenfunctions starting with the ones with the largest eigenvalues, and 2) *task-model alignment* which measures if the task can be learned sample-efficiently given the spectral bias. The task-model alignment can be characterized by our first metric, the *cumulative power* [8], [9], defined as

$$C(\rho) = \frac{\sum_{\gamma<\rho} \mathbf{a}_\gamma^{(\ell)} \cdot \mathbf{a}_\gamma^{(\ell)}}{\sum_\gamma \mathbf{a}_\gamma^{(\ell)} \cdot \mathbf{a}_\gamma^{(\ell)}}. \tag{3}$$

Cumulative power measures the amount of power placed in the first ρ eigenmodes. Earlier the mode ρ for which the $C(\rho)$ reaches near unity, the better the task aligns with the model [8], [9]. This is later demonstrated in Figure 4. The definition of cumulative power is not arbitrary; it can be shown that it enters directly in the expression for generalization error for kernel regression obtained in [9]. To demonstrate how, consider a particular kernel whose spectrum for a given mode ρ^* satisfies $\eta_{\rho^*+1} \ll \eta_{\rho^*} \ll \eta_{\rho^*-1}$. Then, for a training set of size $P \approx \eta_{\rho^*}^{-1}$, the analysis of [9] suggests that the target modes up to η_{ρ^*} are completely learned, and target modes after η_{ρ^*} remain to be learned. It can be shown that the contributions from these modes to the generalization error can be written as $E_g \approx 1 - C(\rho^*)$ [46].
Finally, we note on a subtle point about the rank of the feature layer kernels. The rank of these kernels can be at most N_ℓ since it is composed of N_ℓ rank-1 components. This, in turn, restricts the class of functions expressible through this kernel; components of target functions along

the zero-eigenmodes can not be learned [9]. For perfect generalization, the target function should lie in the low-dimensional space spanned by the eigenfunctions with non-zero eigenvalues. From this point of view, Eq. (2) quantifies the representational power of the features for the particular task; for fewer modes the expansion coefficients $\bar{\mathbf{a}}_{\rho}^{(\ell)}$ are non-zero for top eigenfunctions, the more aligned representation is.

2) **Kernel-Target Alignment (KTA)**: Our metric is defined by [29]:

$$A(K_1, K_2) = \frac{\|K_1 K_2\|_{\mathrm{HS}}^2}{\|K_1 K_1\|_{\mathrm{HS}} \|K_2 K_2\|_{\mathrm{HS}}}, \tag{4}$$

where $\|K_1 K_2\|_{\mathrm{HS}}^2$ denotes the Hilbert-Schmidt norm of the integral operators associated to two kernels and given by

$$\int d\mu(\mathbf{x}) d\mu(\mathbf{x}') K_1(\mathbf{x}, \mathbf{x}') K_2(\mathbf{x}, \mathbf{x}')$$

for the continuous case, and

$$\frac{\mathrm{Tr}(\mathbf{K}_1 \mathbf{K}_2)}{\sqrt{\mathrm{Tr}(\mathbf{K}_1 \mathbf{K}_1)\, \mathrm{Tr}(\mathbf{K}_2 \mathbf{K}_2)}}$$

for the finite case where K_1, K_2 are $P \times P$ Gram matrices on a set of P inputs.

It has been shown that the optimal kernel for sample-efficient learning in kernel regression [29] for a *scalar* target function is rank-1 and given by $\bar{f}(\mathbf{x})$ is $K_{\bar{f}}^{\mathrm{opt}}(\mathbf{x}, \mathbf{x}') = \bar{f}(\mathbf{x}) \bar{f}(\mathbf{x}')$. This makes sense from the task-model alignment point of view; there is a single non-zero eigenvalue and eigenfunction which can be learned, and the target is proportional to this eigenfunction. Similarly, for multi-class targets, the optimal kernel is again rank-1 and given by $K_{\bar{\mathbf{f}}}^{\mathrm{opt}}(\mathbf{x}, \mathbf{x}') = \bar{\mathbf{f}}(\mathbf{x}) \bar{\mathbf{f}}(\mathbf{x})^{\top}$. In this case, the optimal kernel is *operator valued* as it lies in the space of bounded operators in $\mathcal{L}(\mathcal{Y})$ [47]. Note that for the multi-class targets, no layer kernel can approach the optimal kernel $K_{\bar{\mathbf{f}}}^{\mathrm{opt}}$ since they are all scalar valued. Instead, we consider $K_{\bar{\mathbf{f}}}(\mathbf{x}, \mathbf{x}') = \mathrm{Tr}\, K_{\bar{\mathbf{f}}}^{\mathrm{opt}}(\mathbf{x}, \mathbf{x}')$ as our target kernel, which is rank-C since it is built out of C rank-1 components.

One caveat with KTA is that when one or both kernels have constant components in their eigendecomposition (zero-mode), it may obscure the alignment between finite variance components and yield inaccurate results (see [38]). Therefore, here we consider CKA, the centered version of KTA by transforming both kernels as $\mathbf{K}_1 \to \mathbf{H}\mathbf{K}_1\mathbf{H}$ and $\mathbf{K}_2 \to \mathbf{H}\mathbf{K}_2\mathbf{H}$, where $\mathbf{H} = \mathbf{I} - \frac{1}{P}\mathbf{1}\mathbf{1}^{\top}$ is the centering matrix.

IV. Experimental Setting

Given the connection between internal layer kernels and generalizability, we perform a series of experiments on ResNet architectures [48] with the CIFAR-10 dataset [49]. We study a custom ResNet architecture with 11 convolutional layers with max pooling after each residual block based on the methods and the architecture introduced in [50]. We vary the architectural hyperparameters (with/without residual connections), dataset properties (augmented/not-augmented), optimizer choices (SGD/SGD with momentum) and the batch size (512, 2048, 4096). We train each model with all combinations of different hyperparameters for 5000 epochs on mean squared error (MSE) loss with a linearly decaying learning schedule and batch normalization. Out of 48 experiments obtained this way, we consider only 36 experiments whose training accuracy reached above $\%95$. We probe the feature outputs of 10-layers and their induced kernels. We track the spectral properties of these kernels throughout training and study how the representations change based on the metrics described in Section III.

V. Results

In Figure 1, we present the CKA between individual layer kernels and input/target kernels across 36 models which are trained under MSE loss. The models are ordered from left to right according to increasing test accuracy (Figure 6a). To quantify the trends we saw in data, we calculated Pearson correlations between the hyperparameters and training loss, test accuracy, CKA on test data and CKA on training data. These results are presented in Table I.

In Figure 1a,b, we plot the CKA between $K^{(\ell)}$ and the target kernel $K_{\bar{\mathbf{f}}}$ when evaluated both on the training and test sets. We immediately observe that the CKA on the training set is well correlated with CKA on the test set (Figure 2a), and that alignment increases as one goes deeper in the network.

We also observe that higher CKA between the layer kernels and the target, whether evaluated on the training or the test sets, does not necessarily imply better test accuracy, however CKA on the test set is better correlated with generalization (Table I). This may seem counterintuitive, since performing regression with a kernel that aligns well with the target should imply better generalization [29]. However, we note that these networks do not end up effectively performing regression with last layer features.

Some other trends we observe are the following. We find that SGD with momentum significantly improves the CKA between deeper layer kernels and the target compared to vanilla SGD. We see that data augmentation strongly correlates with test accuracy, while its correlation with CKA is not as strong. Finally, we find that the effect of batch size and the presence of residual connections are less significant for generalization.

Furthermore, we study the layer alignments with input features in Figure 1c. We observe that better performing models generally have less alignment with the input across all layers. In Figure 2b, we plot the CKA between layer feature kernels, and targets and inputs. We find that earlier/later layers align better with input/output features, respectively. This implies that the representations hierarchically evolve based on the output correlations from input to output layers (see Figure 1). This evolution is consistent with the theoretical predictions obtained with perturbative corrections to infinite width layer kernels [17], [20], [22], [23].

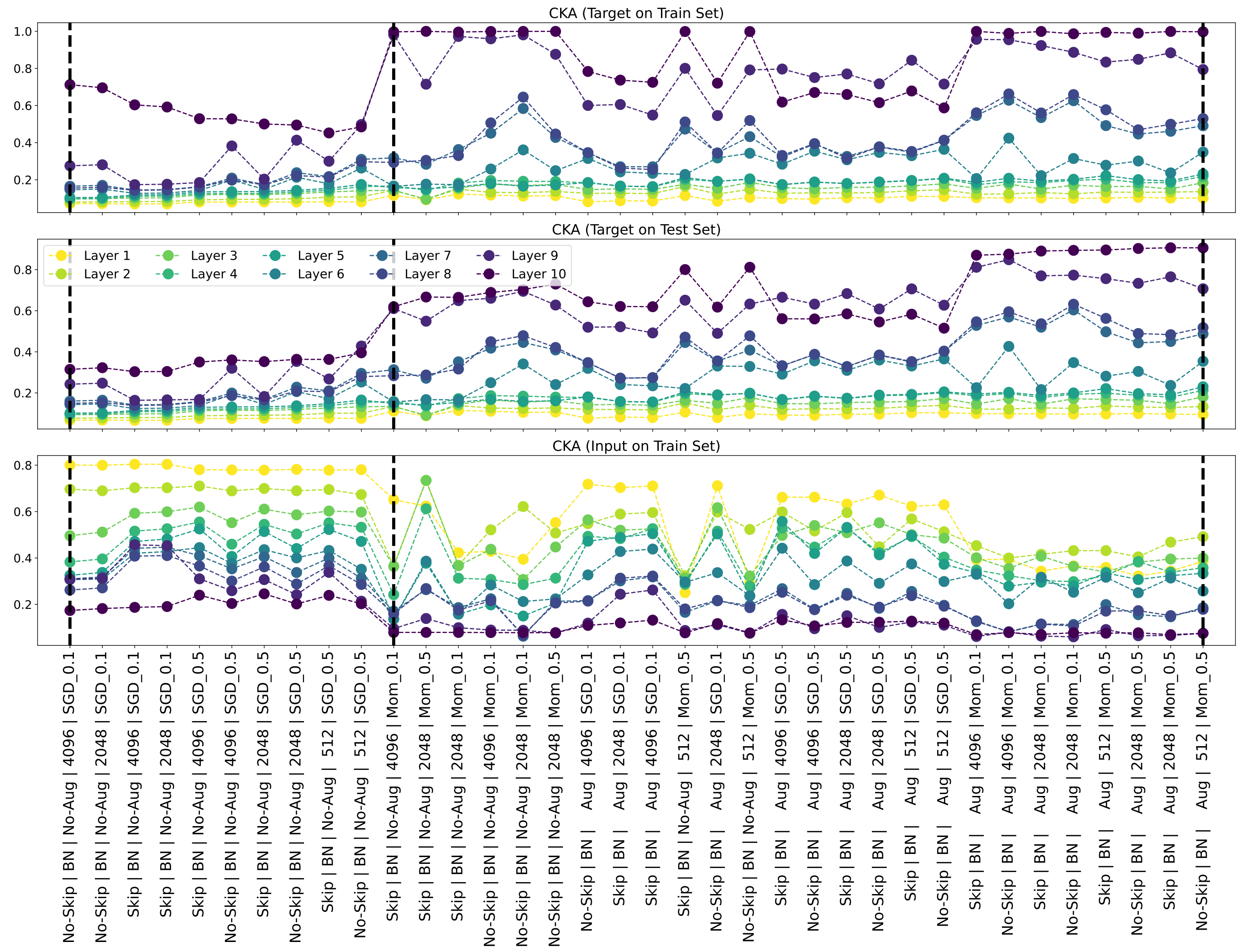

Fig. 1. CKA for the internal layer kernels and labels and inputs at the end of the training. Models are ordered from smallest ($\%58.2$) to highest ($\%94.7$) test accuracy. Labels indicate in order whether there was skip connection and augmentation, the batch size and the optimization method with the learning rate. The target alignment is highest for the last layer and lowest for the first layer, while the opposite holds for the input alignment. The hierarchy of layer alignments from earlier to deeper layers is robust across different experiments. Finally, the target alignment is largest for the last layer when momentum used as the optimizer and the input alignment is largest for the first layer when SGD is used as the optimizer. The vertical dashed lines correspond to three models studied in Figure 3, Figure 5 and Figure 4 from left to right, respectively.

TABLE I
PEARSON CORRELATION COEFFICIENTS BETWEEN DIFFERENT HYPERPARAMETERS AND MODEL PERFORMANCE METRICS

	Loss	Test Acc.	CKA (Test)	CKA (Train)
Skip	0.097	-0.024	-0.026	-0.024
Aug.	0.335	0.806	0.529	0.162
Batch Size	-0.039	-0.247	-0.181	-0.045
LR	-0.078	0.163	-0.035	-0.263
Momentum	0.099	0.448	0.823	0.936
Loss	1.000	0.384	0.366	0.253
Acc.	0.384	1.000	0.850	0.528
CKA (Test)	0.366	0.850	1.000	0.870
CKA (Train)	0.253	0.528	0.870	1.000
Avg. CKA (Test)	0.279	0.920	0.944	0.747
Avg. CKA (Train)	0.221	0.828	0.930	0.837

Next, we investigate the spectral properties of the internal layers of the following three trained models on the training set; the worst ($\%58.2$) and the best performing model ($\%94.7$), and a model with very high alignment in the last two layers (nearly 1) but low test accuracy ($\%75.2$) (tenth point from left in Figure 1a denoted with a vertical dashed line). For each model, we plot the eigenvalues and the cumulative power for Layer-10 (penultimate layer) and Layer-9 at the beginning, middle and end of the training (epochs $0, 25, 5000$, respectively). We chose these models because they all perform perfectly on the training set but one has both low CKA and test accuracy, another has excellent CKA but moderate test accuracy and the last one has both good CKA and test accuracy. We wondered what is intrinsically different between the representations of these models from a spectral point of view.

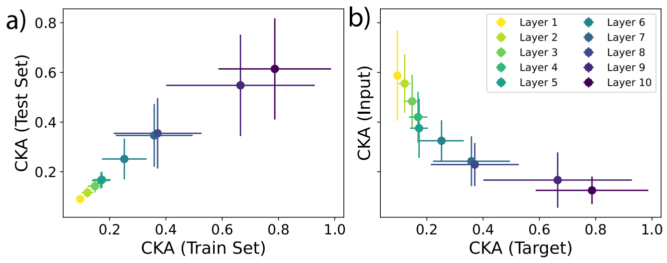

Fig. 2. **a)** CKA between internal layer kernels and target on both training and test set at the end of training. For each layer kernel, errorbars show the standard deviation across 36 experiments. CKA correlates well on training and test sets, and monotonically improves for deeper layers. **b)** CKA between internal layer kernels and target/input on training set. Earlier layers are aligned more with inputs than outputs. Similarly, errorbars show the standard deviation over 36 experiments with different hyperparameters.

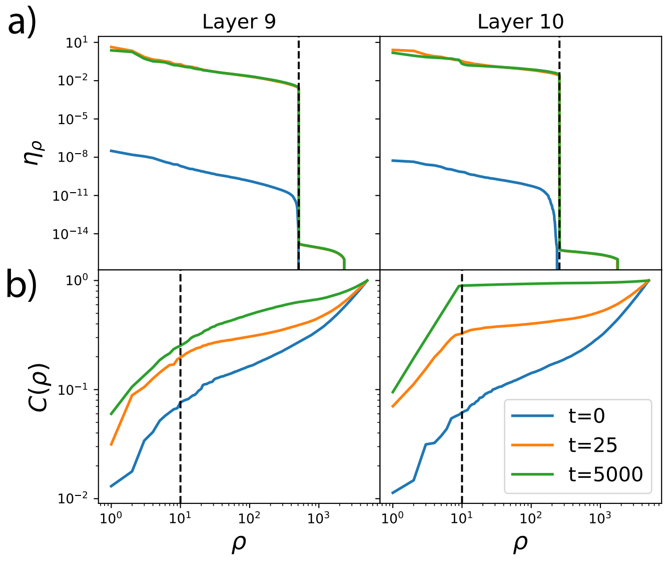

Fig. 3. Eigenspectrum and cumulative power for the last two layers of the worst performing model at three different time points during training. In the top figure, the orange curve mostly coincides with the green curve.

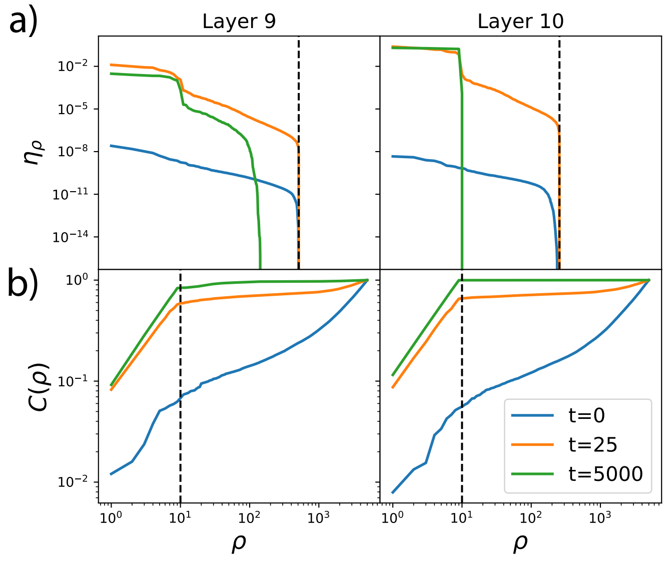

Fig. 4. Eigenspectrum and cumulative power for the last two layers of the best performing model at three different time points during training.

In Figure 3a, we present the spectral properties of the worst performing model (left-most dashed line in Figure 1). As we discussed before, initially the layer kernels have the ranks given by N_ℓ which are $N_9 = 512$ and $N_{10} = 256$ for our architecture (black dashed lines in the eigenvalue panels). Throughout training, the magnitude of the eigenvalues increase significantly but their high-rank structure stays intact, implying there was barely an improvement over their initial spectral bias. At the same time, we track the cumulative power of the target on the eigenbasis of each layer kernel in Figure 3b. The initial total power is distributed over the entire feature space, implying poor task-model alignment. Training, on the other hand, improves the alignment of the target with the penultimate layer kernel significantly on the training dataset, causing a target uniformly distributed over the first $C = 10$ modes (dashed lines in Figure 3b), which is precisely the rank of the target kernel. On the other hand, we do not observe the same with Layer 9, where task-model alignment barely improves.

In Figure 4a for the best performing model (right most dashed line in Figure 1), we see that both layer kernels acquire a low rank structure compared to their initializations. Again, the layer kernels have rank N_ℓ with are $N_9 = 512$ and $N_{10} = 256$ initially. However, throughout training, they adapt to a lower rank structure where the new ranks for both kernels are close to the number of classes $C = 10$, that is, the rank of the target kernel. Hence, the layer kernels get closer to the target kernel, explaining large CKA. At the same time, we track the cumulative power of the target on the eigenbasis of each layer kernel in Figure 4b. Initially, the target alignment is poor, and the total power is distributed over the entire feature space. However, training also improves the alignment of the target on the low-dimensional space spanned by the layer kernel; there are only $C = 10$ (dashed lines in Figure 4b) approximately equally important components of the target when projected on the layer kernels. Hence, training improves both spectral bias and the task-model alignment on the training set.

Having reviewed the spectral outlook of the worst and best models throughout training, we now look at a model which has large CKA between the target and the last layers on the training set, but nevertheless have poor generalization performance (middle dashed line in Figure 1). In Figure 5a, we see a mixture of the best and worst models; the eigenspectrum of the penultimate layer kernel looks very much like it was for the former, the spectrum for the previous layer (Layer 9) looks similar to the one for the latter model. On the other hand, the task-model alignments for both layer look very similar to the case with best model. So what is different? Upon close inspection, we see that Layer 9 kernel still picks up a low-rank structure which is not as pronounced as before, but it is enough to obtain a large CKA. That CKA can be insensitive to smaller but yet important eigenvalues has been pointed out in [41].

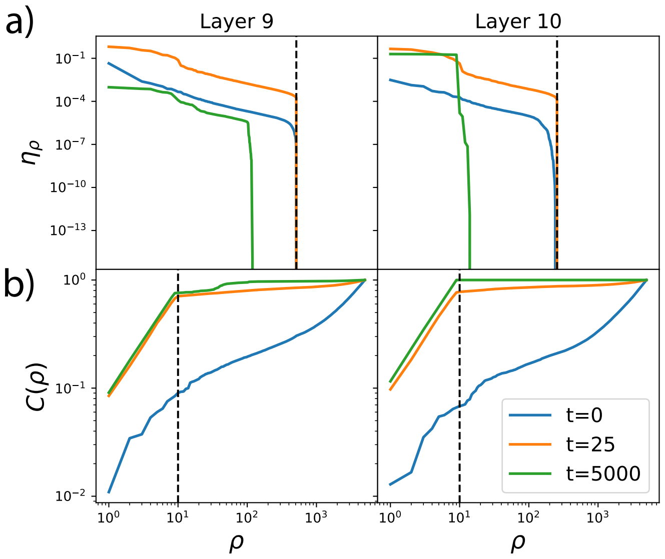

Fig. 5. Eigenspectrum and cumulative power for the last two layers of a poorly generalizing model with high CKA at three different time points during training.

We make the following observations from these figures. First, target-model alignment increases during training. Second, motivated by the observation that the target-model alignment in layer 9 in the worst model is worse than the other models while the target-model alignment in layer 10 is similar, we conjectured that the average CKA across all layers may better correlate with the overall model performance than CKA of the last layer. We test this idea in Figure 6, where we plot the ordered training and test accuracy across models in the first panel and the averaged CKA for each model across its layers. As shown in Table. I, the correlations between layer averaged CKAs for both training and test sets indeed correlate better with the test accuracy. This hints at the fact that a well generalizing model should not only have good alignment with the penultimate layer, but it should also learn important features of the target throughout the entire network.

VI. NTK and Target Alignment

As mentioned previously, study of layer kernels and their CKA with the target function helps to understand the structure of representation learning from a spectral point of view, however it is not enough to understand generalization because 1) the final output of a neural network is not given by the kernel regressor on the layer kernels, and 2) even if we considered kernel regression on the layer kernels, their generalization performance cannot be reduced to a single scalar like CKA and one needs to consider the effect of the spectral bias and task-model alignment, which are implied by the analytical formulas for generalization error studied in [9] and others.

Recently, however, there were developments on describing a trained neural network's output in terms of kernel regression

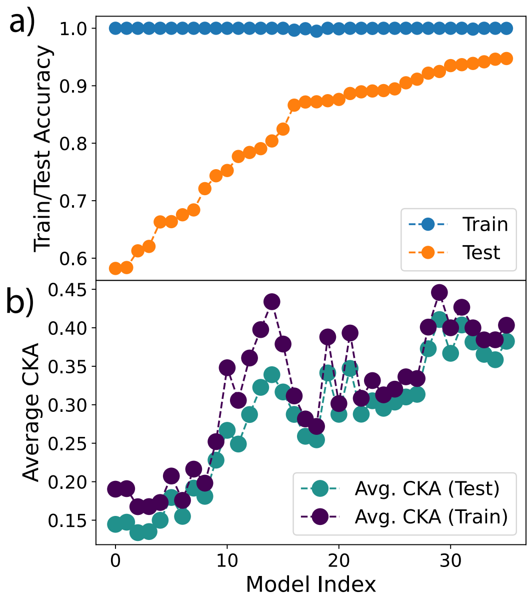

Fig. 6. **a)** Training and test accuracy of the models considered. **b)** Average CKA for each model across layers. Generalization correlates well with overall target alignment with all layers.

with the empirical NTK towards the end of training (e.g. [28], [27], [51]). The findings roughly show that after a certain amount of epochs T during training, the loss landscape becomes flatter and the neural network output can be represented as its Taylor expansion around the parameters θ_T at time T, after which the training resembles performing a kernel regression on the static tangent features (defined below) evaluated at θ_T. Motivated by these observations, we now empirically study the alignment properties of NTK at all times during training.

The continuous time limit of gradient descent training (gradient flow) gives rise to the following training dynamics on a multi-class network function $\mathbf{f} \in \mathbb{R}^C$:

$$\partial_t \mathbf{f}_i(\mathbf{x}) = -\sum_{\mu=1}^{P}\sum_{j=1}^{C} \mathbf{K}_{ij}^{NTK}(\mathbf{x},\mathbf{x}^\mu)\left(\mathbf{f}_j(\mathbf{x}^\mu) - \bar{\mathbf{f}}_j(\mathbf{x}^\mu)\right), \quad (5)$$

where $\{\mathbf{x}^\mu\}_{\mu=1}^P$ are training inputs, $\bar{\mathbf{f}} \in \mathbb{R}^C$ is the target function and $\mathbf{K}^{NTK}(\mathbf{x},\mathbf{x}') : \mathcal{X}\times\mathcal{X} \to \mathbb{R}^{C\times C}$ is the operator-valued NTK given in terms of the tangent features $\boldsymbol{\psi}_i(\mathbf{x}) = \nabla_\theta \mathbf{f}_i(\mathbf{x})$ as:

$$\mathbf{K}_{ij}^{NTK}(\mathbf{x},\mathbf{x}') = \boldsymbol{\psi}_i(\mathbf{x})^\top \boldsymbol{\psi}_j(\mathbf{x}') = \nabla_\theta \mathbf{f}_i(\mathbf{x})^\top \nabla_\theta \mathbf{f}_j(\mathbf{x}'). \quad (6)$$

Note that throughout training, NTK evolves differently for different class pairs and mixes them in a non-trivial way. This is in contrast to the scalar layer kernels $K^{(\ell)}$ which are class agnostic, as discussed above. Again considering networks of the form $\mathbf{f}(\mathbf{x}) = \mathbf{W}\boldsymbol{\psi}^{(L)}(\mathbf{x})$, where $\boldsymbol{\psi}^{(L)}$ is the outputs of the penultimate layer, NTK can be reduced to the form:

$$\mathbf{K}_{ij}^{NTK}(\mathbf{x},\mathbf{x}') = K^{(L)}(\mathbf{x},\mathbf{x}')\mathbf{I}_{ij} + \mathbf{K}_{ij}^{\nabla}(\mathbf{x},\mathbf{x}'), \quad (7)$$

where $\mathbf{K}_{ij}^{\nabla}(\mathbf{x},\mathbf{x}') = \sum_{kl} \mathbf{W}_{ik}\mathbf{W}_{jl}\nabla_\theta \psi_k^{(L)}(\mathbf{x})^\top \nabla_\theta \psi_l^{(L)}(\mathbf{x}')$ is the *gradient kernel*. Notice that the first term of NTK is the penultimate layer kernel on the block diagonals of NTK that does not mix different labels and the second term includes the tangent features coming from gradients with respect to the parameters of earlier layers and is the source of classwise mixing.

While a proper treatment of NTK requires the study of its alignment with the rank-1 target kernel $\mathbf{K}_{\bar{\mathbf{f}}}^{opt}(\mathbf{x},\mathbf{x}') = \bar{\mathbf{f}}(\mathbf{x})\bar{\mathbf{f}}(\mathbf{x}')^\top$ introduced in Sec. III, for simplicity we will consider the trace of NTK over the class indices:

$$K^{NTK}(\mathbf{x},\mathbf{x}') = K^{(L)}(\mathbf{x},\mathbf{x}') + \frac{1}{C}\operatorname{Tr}\mathbf{K}^{\nabla}(\mathbf{x},\mathbf{x}') \qquad (8)$$

and study its alignment with the target kernel $K_{\bar{\mathbf{f}}} = \bar{\mathbf{f}}(\mathbf{x})^\top\bar{\mathbf{f}}(\mathbf{x}')$. Recall that $K^{(L)}(\mathbf{x},\mathbf{x}')$ becomes low rank with near-perfect alignment to $K_{\bar{\mathbf{f}}}$ whose rank is C. Hence, in the following experiments, we study if the $K^{NTK}(\mathbf{x},\mathbf{x}')$ inherits this low-rank representation throughout training. This analysis also gives us a chance to characterize the spectral properties of the gradient kernel K^{∇}.

Since the numerical computation of NTK is both computationally and memory expensive [52], [53], we consider a 4 hidden layer feedforward neural network with $N = 200$ units at each layer whose weights are initialized according to the distribution $\mathbf{W}_{ij} \sim \mathcal{N}(0, \frac{1}{N})$ and biases set to zero. We train this neural network with 10000 samples from MNIST dataset [54] with batch size 500, and test it on another 10000 samples from the same dataset. The network was trained with ADAM optimizer with learning rate 0.1 for 3000 epochs. A similar experiment has been conducted in [55] where the alignments were measured with respect to the full operator valued NTK, and it was observed that the alignment increases throughout training. Instead, here we consider the total alignment of NTK by summing it over the class indices and consider the overall alignment with target.

In Figure 7, we show that this network reaches almost $\%95$ test accuracy on MNIST (first panel). We also show NTK's alignment with the target kernel, as well as its alignment with the network kernel $K_{\mathbf{f}}(\mathbf{x},\mathbf{x}') = \mathbf{f}(\mathbf{x})^\top\mathbf{f}(\mathbf{x}')$ throughout training (second panel). NTK seems to approach to the target kernel asymptotically, which implies that it inherits the low-rank structure of the penultimate layer kernel $K^{(L)}$ as described below Eq. (8).

In Figure 8, we supplement this observation further with the evolution of the spectrum of NTK and the cumulative power of the target when projected on the tangent features throughout training. Indeed, NTK acquires the low-rank structure coming from the alignment of its first term ($K^{(L)}$), and furthermore the effect of this term seems to be the dominating one. In future works, it would be interesting to derive this effect theoretically.

VII. Conclusion

In this paper, inspired by the recent theoretical developments in generalization in kernel regression [8], [9], [13], we performed empirical analysis of representation learning in deep neural networks by inspecting their spectral properties. We found that the target alignment with individual layer

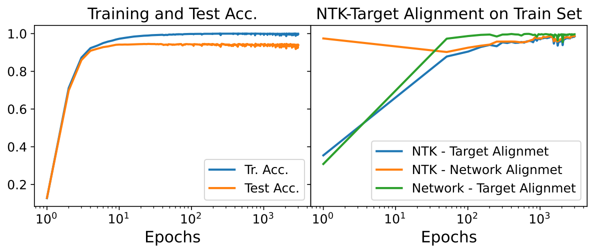

Fig. 7. **(Left Panel)** Training and test accuracy throughout training for the 4-layer feedforward neural network. **(Right Panel)** Alignments between the pairs K^{NTK} - $K_{\bar{\mathbf{f}}}$ (NTK - Target Alignment), K^{NTK} - $K_{\mathbf{f}}$ (NTK - Network Alignment) and $K_{\bar{\mathbf{f}}}$ - $K_{\mathbf{f}}$ (Network - Target Alignment).

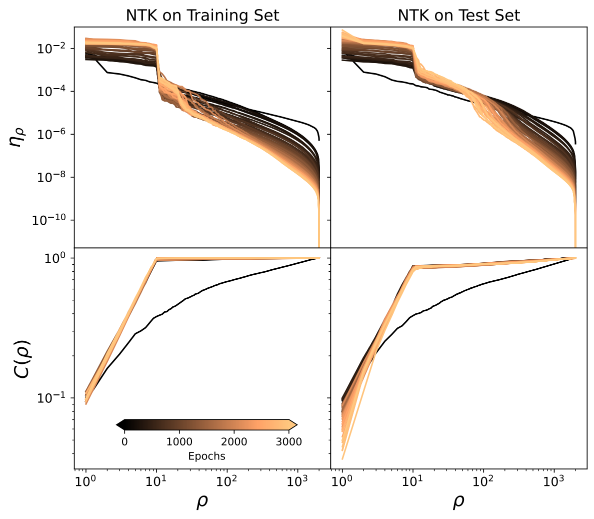

Fig. 8. Eigenspectrum of NTK on both training and test sets as well as the cumulative power of the target when projected on the tangent features of NTK during training. NTK acquires the low-rank structure of the target kernel.

kernels on the training set does not always correlate well with generalization, but rather there needs to be a notion of overall network alignment with the target including all layers. Here, we simply introduced average CKA across all layers and found that this quantity correlates well with generalization.

Furthermore, we studied the representations learned by the tangent features of deep neural networks by studying the spectral properties of NTK. We found that NTK, a rather complicated object compared to layer kernels, also acquires a low-rank structure during training and aligns well with the target. This is in sharp contrast to the perturbative treatments of NTK in the infinite width limit where the low rank corrections stay small [17], [20], [22].

References

[1] D. Mumford and A. Desolneux, *Pattern theory: the stochastic analysis of real-world signals*. CRC Press, 2010.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] A. Jacot, F. Gabriel, and C. Hongler, "Neural tangent kernel: Convergence and generalization in neural networks," in *Advances in neural information processing systems*, 2018, pp. 8571–8580.

[4] S. Arora, S. S. Du, W. Hu, Z. Li, R. R. Salakhutdinov, and R. Wang, "On exact computation with an infinitely wide neural net," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[5] J. Lee, L. Xiao, S. Schoenholz, Y. Bahri, R. Novak, J. Sohl-Dickstein, and J. Pennington, "Wide neural networks of any depth evolve as linear models under gradient descent," *Advances in neural information processing systems*, vol. 32, 2019.

[6] L. Chizat, E. Oyallon, and F. Bach, "On lazy training in differentiable programming," 2018.

[7] Y. Cao, Z. Fang, Y. Wu, D.-X. Zhou, and Q. Gu, "Towards understanding the spectral bias of deep learning," 2019.

[8] B. Bordelon, A. Canatar, and C. Pehlevan, "Spectrum dependent learning curves in kernel regression and wide neural networks," in *Proceedings of the 37th International Conference on Machine Learning*, 2020.

[9] A. Canatar, B. Bordelon, and C. Pehlevan, "Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks," *Nature Communications*, vol. 12, no. 1, pp. 1–12, 2021.

[10] A. Jacot, B. Simsek, F. Spadaro, C. Hongler, and F. Gabriel, "Kernel alignment risk estimator: Risk prediction from training data," 2020.

[11] K. Xu, M. Zhang, J. Li, S. S. Du, K. ichi Kawarabayashi, and S. Jegelka, "How neural networks extrapolate: From feedforward to graph neural networks," 2020.

[12] Y. Dandi and A. Jacot, "Understanding layer-wise contributions in deep neural networks through spectral analysis," 2021.

[13] J. B. Simon, M. Dickens, and M. R. DeWeese, "A theory of the inductive bias and generalization of kernel regression and wide neural networks," 2021.

[14] G. Yang and E. J. Hu, "Tensor programs iv: Feature learning in infinite-width neural networks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 11 727–11 737.

[15] A. Christmann and I. Steinwart, "Self-consistent dynamical field theory of kernel evolution in wide neural networks," in *Advances in Neural Information Processing Systems*, 2022.

[16] E. Dyer and G. Gur-Ari, "Asymptotics of wide networks from feynman diagrams," in *International Conference on Learning Representations*, 2019.

[17] S. Yaida, "Non-gaussian processes and neural networks at finite widths," in *Mathematical and Scientific Machine Learning*. PMLR, 2020, pp. 165–192.

[18] J. Huang and H.-T. Yau, "Dynamics of deep neural networks and neural tangent hierarchy," in *International conference on machine learning*. PMLR, 2020, pp. 4542–4551.

[19] B. Hanin and M. Nica, "Finite depth and width corrections to the neural tangent kernel," in *International Conference on Learning Representations*, 2019.

[20] J. Zavatone-Veth, A. Canatar, B. Ruben, and C. Pehlevan, "Asymptotics of representation learning in finite bayesian neural networks," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[21] G. Naveh, O. B. David, H. Sompolinsky, and Z. Ringel, "Predicting the outputs of finite deep neural networks trained with noisy gradients," *Physical Review E*, vol. 104, no. 6, p. 064301, 2021.

[22] Q. Li and H. Sompolinsky, "Statistical mechanics of deep linear neural networks: The backpropagating kernel renormalization," *Physical Review X*, vol. 11, no. 3, p. 031059, 2021.

[23] D. A. Roberts, S. Yaida, and B. Hanin, *The principles of deep learning theory*. Cambridge University Press, 2022.

[24] J. Zavatone-Veth and C. Pehlevan, "Exact marginal prior distributions of finite bayesian neural networks," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[25] A. Lewkowycz, Y. Bahri, E. Dyer, J. Sohl-Dickstein, and G. Gur-Ari, "The large learning rate phase of deep learning: the catapult mechanism," *arXiv preprint arXiv:2003.02218*, 2020.

[26] V. Papyan, X. Han, and D. L. Donoho, "Prevalence of neural collapse during the terminal phase of deep learning training," *Proceedings of the National Academy of Sciences*, vol. 117, no. 40, pp. 24 652–24 663, 2020.

[27] A. Atanasov, B. Bordelon, and C. Pehlevan, "Neural networks as kernel learners: The silent alignment effect," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=1NvflqAdoom

[28] S. Fort, G. K. Dziugaite, M. Paul, S. Kharaghani, D. M. Roy, and S. Ganguli, "Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel," 2020.

[29] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola, "On kernel-target alignment," in *Advances in Neural Information Processing Systems*, T. Dietterich, S. Becker, and Z. Ghahramani, Eds., vol. 14. MIT Press, 2001. [Online]. Available: https://proceedings.neurips.cc/paper/2001/file/1f71e393b3809197ed66df836fe833e5-Paper.pdf

[30] G. Alain and Y. Bengio, "Understanding intermediate layers using linear classifier probes," 2016.

[31] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[32] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, no. 3/4, pp. 321–377, 1936.

[33] J. Ramsay, J. ten Berge, and G. Styan, "Matrix correlation," *Psychometrika*, vol. 49, no. 3, pp. 403–423, 1984.

[34] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural computation*, vol. 16, no. 12, pp. 2639–2664, 2004.

[35] P. Geladi and B. R. Kowalski, "Partial least-squares regression: a tutorial," *Analytica chimica acta*, vol. 185, pp. 1–17, 1986.

[36] F. R. Bach and M. I. Jordan, "Kernel independent component analysis," *Journal of machine learning research*, vol. 3, no. Jul, pp. 1–48, 2002.

[37] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, "Measuring statistical dependence with hilbert-schmidt norms," in *International conference on algorithmic learning theory*. Springer, 2005, pp. 63–77.

[38] C. Cortes, M. Mohri, and A. Rostamizadeh, "Algorithms for learning kernels based on centered alignment," *The Journal of Machine Learning Research*, vol. 13, pp. 795–828, 2012.

[39] L. Sun, S. Ji, S. Yu, and J. Ye, "On the equivalence between canonical correlation analysis and orthonormalized partial least squares," in *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

[40] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," 2019.

[41] F. Ding, J.-S. Denain, and J. Steinhardt, "Grounding representation similarity with statistical testing," 2021.

[42] A. S. Morcos, M. Raghu, and S. Bengio, "Insights on representational similarity in neural networks with canonical correlation," 2018.

[43] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," 2014.

[44] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein, "Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability," 2017.

[45] J. Mercer, "Functions of positive and negative type, and their connection the theory of integral equations," *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, vol. 209, no. 441-458, pp. 415–446, 1909.

[46] A. Canatar, E. Peters, C. Pehlevan, S. M. Wild, and R. Shaydulin, "Bandwidth enables generalization in quantum kernel models," 2022.

[47] C. Carmeli, E. De Vito, and A. Toigo, "Vector valued reproducing kernel hilbert spaces of integrable functions and mercer theorem," *Analysis and Applications*, vol. 4, no. 04, pp. 377–408, 2006.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[49] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[50] G. Leclerc, A. Ilyas, L. Engstrom, S. M. Park, H. Salman, and A. Madry, "ffcv," https://github.com/libffcv/ffcv/, 2022, commit xxxxxxx.

[51] N. Vyas, Y. Bansal, and P. Nakkiran, "Limitations of the ntk for understanding generalization in deep learning," 2022.

[52] R. Novak, L. Xiao, J. Hron, J. Lee, A. A. Alemi, J. Sohl-Dickstein, and S. S. Schoenholz, "Neural tangents: Fast and easy infinite neural networks in python," 2019. [Online]. Available: https://arxiv.org/abs/1912.02803

[53] R. Novak, J. Sohl-Dickstein, and S. S. Schoenholz, "Fast finite width neural tangent kernel," 2022.

[54] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[55] A. Baratin, T. George, C. Laurent, R. D. Hjelm, G. Lajoie, P. Vincent, and S. Lacoste-Julien, "Implicit regularization via neural feature alignment," 2020.