A Kernel Perspective for the Decision Boundary of Deep Neural Networks

Yifan Zhang and Shizhong Liao*
College of Intelligence and Computing
Tianjin University
Tianjin 300350, China
szliao@tju.edu.cn

***Abstract*—Deep learning has achieved great success in many fields, but they still lack theoretical understandings. Although some recent theoretical and experimental results have investigated the representation power of deep learning, little effort has been devoted to analyzing the generalization ability of deep learning. In this paper, we analyze deep neural networks from a kernel perspective and use kernel methods to investigate the effect of the implicit regularization introduced by gradient descent on the generalization ability. Firstly, we argue that the multi-layer nonlinear feature transformation in deep neural networks is equivalent to a kernel feature mapping and analyze our point from the perspective of the unique mathematical advantages of kernel methods and the method of constructing multi-layer kernel machines, respectively. Secondly, using the representer theorem, we analyze the decision boundary of deep neural networks and prove that the last hidden layers of deep neural networks converge to nonlinear SVMs. Systematical experiments demonstrate that the decision boundaries of neural networks converge to those of nonlinear SVMs.**

***Keywords*-deep neural network; kernel method; generalization ability; gradient descent; decision boundary**

I. Introduction

Recent work in machine learning has given a revival of attention to deep learning due to the increase of data, the enhancement of computing power and the progress of learning algorithms. Deep learning technology has made important advances in pattern recognition, computer vision, natural language processing and other fields. Deep learning is increasingly used to solve general artificial intelligence problems such as inference and decision due to its powerful capabilities. Deep neural networks, classical deep learning models, are huge-dimensional and typically involve more parameters than training samples, which presents challenges in achieving good generalization. While a significant amount of research has been dedicated to developing high-efficiency and flexible deep learning machines [1] [2] with the state-of-the-art performance on numerous datasets, deep networks are however not understood theoretically [3] [4].

In parallel to the success of deep neural networks, the nonlinear mapping introduced by kernel methods [5] shows obvious computational advantages over deep architectures in model training. The Reproducing Kernel Hilbert Space (RKHS) implicitly defined by the kernel naturally introduces the prior knowledge about the learning problem [6]. Kernel functions measure the similarity between two points in the input distribution, and implicitly define the corresponding feature mappings, which ensures the analysis of many complex real-world problems in the kernel framework [7]. Therefore, it is naturally of interest to extend the elegance of kernel methods to the analysis of deep neural networks.

As we all know, the *generalization ability*, i.e., the performance of learning machines trained on certain datasets on unseen data, of learning machines is an important question of theoretical research in machine learning. Empirical risk minimization is a typical learning strategy for dealing with the generalization, which suggests finding the function that minimizes the empirical loss. There are many mathematical measures of hypothesis space complexity that yield generalization error bounds, such as VC dimensions, Rademacher complexity, cover numbers, etc. [8]. Regularization is also a typical method to improve the generalization ability of learning machines, which avoids overfitting by limiting the complexity of the model. However, through experimental results, Zhang et al. [3] showed that the traditional generalization error analysis based on VC dimensions and Rademacher complexity cannot explain the phenomenon that deep neural networks are heavily over-parametrized but still have good generalization. Moreover, the implicit regularization, which limits the amount of computation for better performance, introduced by gradient-based optimization is largely responsible for good generalization performance, rather than the explicit regularization [9]. All these approaches attempt to find a good trade-off between the representation ability and the complexity of models, which leads to the learning machine with better generalization ability.

In this paper, we analyze deep neural networks from a kernel perspective and use kernel methods to investigate the generalization ability of deep neural networks, or more precisely, to investigate the effect of the implicit regularization introduced by gradient descent (GD) algorithm on generalization ability. Firstly, we argue that the multi-layer nonlinear feature transformation in deep networks is equivalent to a kernel feature mapping and analyze our point of view from shallow networks and deep networks, respectively. For the case of shallow networks, since kernel machines can be interpreted as shallow networks naturally,

* Corresponding author.

we illustrate the unique mathematical advantages of using kernel methods to analyze neural networks from the elegance of the representer theorem. For the case of deep networks, we expound the rationality of using kernel methods to analyze deep neural networks from the method of constructing multi-layer kernel machines (MKMs). Secondly, using the representer theorem, we investigate the decision boundary of deep neural networks, then we propose the theorem that the last hidden layers of neural networks converge to nonlinear SVMs. Finally, we conduct experiments on classification tasks and demonstrate that the decision boundaries of neural networks converge to those of nonlinear SVMs on three simulated datasets. We also evaluate the performance of fully connected neural networks compared with nonlinear SVMs and analyze the experimental results in detail.

II. Related Work

Recently, there have been more and more research try to use kernel methods to analyze deep learning, and have produced several meaningful theoretical results. Meanwhile, a number of deep kernels and MKMs have been proposed to link kernel methods with deep learning. We review the related work of kernel methods for deep learning in terms of theory and method respectively.

The theoretical research of kernel methods for deep learning mainly includes the representer theorem for deep learning and the generalization theory of deep neural networks from a kernel perspective. Belkin et al. [4] illustrated the importance of the representer theorem for the theoretical research of deep learning from the perspective of inductive bias. Bohn et al. [10] established the relationship between the representer theorem and MKMs. In terms of the generalization theory, Bietti et al. [2] proposed to use RKHS norm to regularize deep neural networks from the perspective of regularization. Suzuki et al. [11] defined the corresponding RKHS for each layer of the deep network and derived generalization error bounds. Li et al. [12] investigated generalization from the perspective of implicit regularization of optimization algorithms. Besides, Kornblith et al. [13] introduced centralized kernel alignment (CKA) as a similarity index to measure the relationship between representations in networks. Montavon et al. [14] investigated the layer-wise evolution of the representation for deep networks in RKHS. Nevertheless, there is still a lack of new theory for kernel methods to analyze deep learning [4].

Moreover, some methods have been proposed to introduce kernels into deep learning and further construct deep kernels even MKMs. Cho et al. [15] designed MKMs based on Arc-cosine kernels which can also be used for Gaussian processes (GP). Hazan et al. [16] introduced stochastic kernels which are derived from GP and encode the information of two infinite layers, Lee et al. [17] further derived GP kernels that can effectively mimic deep neural networks. Mairal et al. [18] used reproducing kernels to encode the invariance of image representations, they devised convolution kernels and constructed convolution kernel networks (CKNs), they [19] also suggested improving CKNs with supervision and produced a new convolutional neural network. Furthermore, Wang et al. [1] constructed SVM-based deep stacking networks. Xue et al. [20] proposed deep spectral kernel networks based on inverse Fourier transform. However, these methods are frequently difficult to generalize.

III. Problem Statement

We have no strict requirements for the input data and network architecture. The last hidden layer of the network and the output layer are fully connected, which is the common case. Given a dataset $\{(\boldsymbol{x}_n, y_n), n = 1, \ldots, N\}$, where $\boldsymbol{x}_n \in \mathbb{R}^d$ and $y_n \in \{-1, 1\}$ for binary classification, $y_n \in \{1, 2, \ldots, K\}$ for multi-class classification. We denote the data matrix as $\mathbf{X} \in \mathbb{R}^{d\times N}$.

Let the number of layers of the network be l, excluding the output layer. For binary classification, let $f(\boldsymbol{x}; \theta) : \mathbb{R}^d \to \{-1, 1\}$ be the function defined by the network, θ is the parameter set of the network. The transformation of the first $l-1$ layers can be written as $\phi(\boldsymbol{x}) = h(\boldsymbol{x}; \sigma)$, σ is the parameter set of the first $l-1$ layers. Therefore, $\phi(\boldsymbol{x})$ is the input of the l-th layer in the network.

The form of minimizing the empirical loss is as follows:

$$\mathcal{L}(\theta) = \sum_{n=1}^{N} \ell\left(f\left(\boldsymbol{x}_n; \theta\right), y_n\right),$$

where $\ell(\cdot)$ is the loss function, e.g., logistic loss, exponential loss, sigmoidal loss and cross-entropy loss.

For exponential loss, the empirical loss is

$$\mathcal{L}(\theta) = \sum_{n=1}^{N} e^{-y_n f(\boldsymbol{x}_n;\theta)}. \tag{1}$$

The goal of optimization is $\theta = \arg\min_\theta \mathcal{L}(\theta)$.

We use gradient descent (GD) algorithm to minimize (1) with a constant learning rate η, and the iterative step is that $\theta(t) = \theta(t-1) - \eta\nabla\mathcal{L}\left(\theta(t-1)\right)$.

IV. Main Results

In this section, We first argue that the multi-layer nonlinear feature transformation in deep networks is equivalent to a kernel feature mapping and analyze our point for the case of shallow networks and deep networks in detail, respectively. Subsequently, we propose the decision boundary convergence theorem by the representer theorem.

A. Nonlinear transformation in Deep Networks v.s. Kernel Feature Mapping

Deep learning is widely used to solve representation learning problems in machine learning because of deep architectures. At present, the models used in deep learning are mainly neural networks. Deep architectures learn complex

mappings of inputs through multi-layer nonlinear transformations [21]. They involve complex nonlinear optimization and many heuristics, this makes them usually difficult to train. In parallel to deep networks, kernel methods offer an elegant framework to analyze data representation. SVMs learn nonlinear classifiers through "kernel trick", they are trained by solving simple quadratic programming problems. In recent years, a considerable amount of work has shown that kernel methods also benefit from the advantages of deep networks [15] [19] [1]. Therefore, we argue that the multi-layer nonlinear feature transformation in deep networks is equivalent to a kernel feature mapping. We will analyze our point in detail.

For the case of shallow networks, by large numbers of experiments, [4] has shown that kernel machines perform very well on many datasets even in the presence of label noise, which also indicates that kernel machines also have strong generalization performance comparable to deep networks. As a matter of fact, kernel machines can be regarded as the two-layer neural networks with a fixed first layer, so it is natural to use kernel methods to analyze neural networks.

Furthermore, we can use various mathematical analysis techniques to analyze kernel machines analytically due to the elegance of the kernel framework. More generally, the well-known representer theorem, which is the important theory for analyzing kernel methods, shows that for the general loss function and regularization, the optimal solution of the optimization problem can be expressed as a linear combination of kernel functions.

Given a dataset $\{(\boldsymbol{x}_n, y_n), n = 1, \ldots, N\}$ drawn independently from a probability distribution P on $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}$. Let $\mathcal{H} := \mathcal{H}(\mathcal{X}, \mathcal{Y})$ be a Reproducing Kernel Hilbert Space of real-value functions on $\mathcal{X}$. The minimal norm interpolant is

$$f^* := \arg\min_{f \in \mathcal{H}} \|f\|_{\mathcal{H}} \text{ such that } f(x_i) = y_i \quad \forall i = 1, \ldots, N.$$

The classical representer theorem [22] shows that the minimum norm solution in RKHS can be written as a finite linear combination of kernel functions $\{K(\boldsymbol{x}_1, \cdot), \ldots, K(\boldsymbol{x}_n, \cdot)\}$, namely

$$f^*(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i^* K(\boldsymbol{x}_i, \boldsymbol{x}),$$

where $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ denotes the kernel function of $\mathcal{H}$ and $\alpha_i^* \in \mathbb{R}, i = 1, \ldots, N$ are the corresponding coefficients. We can compute $\boldsymbol{\alpha}_i^* = (\alpha_1^*, \ldots, \alpha_N^*)^\top$ by solving the following formula

$$\mathbf{K}\boldsymbol{\alpha}^* = \boldsymbol{y},$$

where $\mathbf{K}$ is the kernel matrix, $\boldsymbol{y} = (y_1, \ldots, y_N)^\top$. The kernel matrix $\mathbf{K}$ is invertible if the kernel K is positive definite. As such, compared with neural networks, the representer theorem ensures the minimum norm solution of kernel machines can be computed analytically. This unique and elegant mathematical property is an important reason that we use kernel methods to analyze neural networks. Our theoretical analysis for deep networks is also closely related to the representer theorem.

For the case of deep neural networks, we can also construct corresponding MKMs which benefit from deep architectures. The kernel function produces an implicit feature map $\boldsymbol{x} \mapsto \phi(\boldsymbol{x})$ and computes the inner product in the induced RKHS. Although the mapping of a kernel function is implicit, we can avoid the limitation and directly construct MKMs by combining nonlinear mappings in a composite way:

$$K^l(\boldsymbol{x}, \boldsymbol{x}') = \left\langle \phi^l\left(\phi^{l-1}\left(\cdots \phi^1(\boldsymbol{x})\right)\right), \phi^l\left(\phi^{l-1}\left(\cdots \phi^1(\boldsymbol{x}')\right)\right)\right\rangle,$$

where K^l denotes the l-layer MKMs, ϕ^l denotes the l-th layer kernel mapping. We can further write K^l as:

$$K^l(\boldsymbol{x}, \boldsymbol{x}') = \left\langle \Phi^l(\boldsymbol{x}), \Phi^l(\boldsymbol{x}')\right\rangle,$$

where the l-th layer kernel mapping is:

$$\Phi^l(\boldsymbol{x}) = \phi^l\left(\phi^{l-1}\left(\cdots \phi^l(\boldsymbol{x})\right)\right).$$

The motivation is very intuitive, since the kernel function can simulate the computation in a single hidden layer neural network, then the composite mapping can mimic the computation in a deep neural network. Moreover, [10] studied the representer theorem of deep kernel learning. There are also some research related to deep kernels, such as [15] [20] [17] [18]. All these work illustrate that it is feasible to construct MKMs comparable to deep networks.

For the case of shallow networks and deep networks, we analyze our point from the perspective of the unique mathematical advantages of kernel methods and the method of constructing MKMs, respectively. Subsequently, We will analyze the convergence between the decision boundaries of deep neural networks and those of nonlinear SVMs in the RKHS induced by the multi-layer nonlinear transformation of deep networks.

B. Theoretical Results

According to our point of view, the multi-layer nonlinear feature transformation in deep networks is equivalent to a kernel feature mapping. Consequently, a kernel mapping equivalent to the transformation of the first $l-1$ layers exists and can be found. When emphasizing the l-th layer, the neural network is transformed into minimizing a logistic regression problem using GD in RKHS, and the input data of the logistic regression problem is the transformed output of the first $l-1$ layers, which is the feature vector in RKHS, i.e., $\boldsymbol{\Phi} = [\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_n)] \in \mathbb{R}^{p \times N}$, $\phi(\boldsymbol{x}_n) \in \mathcal{H}$. Then the form of experience loss is transformed as follows:

$$\mathcal{L}(\boldsymbol{\gamma}) = \sum_{n=1}^{N} \ell\left(y_n \boldsymbol{\gamma}^\top \phi(\boldsymbol{x}_n)\right),$$

where $\boldsymbol{\gamma}$ is the weight vector of l-th layer, $\boldsymbol{\gamma} \in \mathcal{H}$. We assume that $\forall n : y_n = 1$ without loss of generality (we can re-define $y_n \phi(\boldsymbol{x}_n)$ as $\phi(\boldsymbol{x}_n)$).

According to the reproducibility of kernel functions, i.e.,

$$f(\boldsymbol{x}) = f(\cdot) \cdot K(\cdot, \boldsymbol{x}),\ K(\boldsymbol{x}, \boldsymbol{z}) = K(\cdot, \boldsymbol{x}) \cdot K(\cdot, \boldsymbol{z}),$$

and the representer theorem, i.e.,

$$f(\cdot) = \sum_{i=1}^{N} \alpha_i K(\cdot, \boldsymbol{x}_i),$$

where $\boldsymbol{x}, \boldsymbol{z} \in \mathcal{X}$, $\alpha_i \in \mathbb{R}, i = 1, 2, \ldots, N$. For the weight vector of the l-th layer, we have

$$\boldsymbol{\gamma} = \sum_{i=1}^{N} \xi_i K(\cdot, \phi(\boldsymbol{x}_i)).$$

For this reason, the GD update for $\boldsymbol{\gamma}$ in RKHS is different from the common case in Euclidean space, but for the sake of simplicity, we take $\boldsymbol{\gamma}$ as a whole and do not expand it, which does not affect the final result, see [23]. The iterative steps of $\boldsymbol{\gamma}$ are as follows:

$$\boldsymbol{\gamma}(t) = \boldsymbol{\gamma}(t-1) - \eta \sum_{n=1}^{N} \ell'(\boldsymbol{\gamma}(t-1)^\top \phi(\boldsymbol{x}_n)) \phi(\boldsymbol{x}_n). \quad (2)$$

[24] investigated linearly separable logistic regression problems and showed that:

Lemma 1. *For a linearly separable logistic regression problem and β-smooth decreasing loss function, i.e., its derivative is β-Lipshitz, whose weight vector is $\mathbf{w} \in \mathbb{R}^d$. Let $\mathbf{w}(t)$ be the iterates of gradient descent with $\eta < 2\beta^{-1}\sigma_{\max}^{-2}(\mathbf{X})$ and any starting point $\mathbf{w}(0)$, where $\sigma_{\max}(\mathbf{X})$ is the maximal singular value of the data matrix $\mathbf{X} \in \mathbb{R}^{d \times N}$. We have: (1) $\lim_{t\to\infty} \mathcal{L}(\mathbf{w}(t)) = 0$, (2) $\lim_{t\to\infty} \|\mathbf{w}(t)\| = \infty$, and (3) $\forall n : \lim_{t\to\infty} \mathbf{w}(t)^\top \mathbf{x}_n = \infty$.*

The direction of $\mathbf{w}(t)$ converges to the solution of the hard margin SVM.

Lemma 2. *For any linearly separable dataset (except for measuring zero) and any β-smooth decreasing loss function with an exponential tail (the loss function is bounded by two exponential functions), any step size $\eta < 2\beta^{-1}\sigma_{\max}^{-2}(\mathbf{X})$ and any starting point $\mathbf{w}(0)$, the gradient descent will behave as:*

$$\mathbf{w}(t) = \hat{\mathbf{w}} \log t + \boldsymbol{\rho}(t),$$

where $\hat{\mathbf{w}}$ is the L_2 max margin vector:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w} \in \mathbb{R}^d} \|\mathbf{w}\|^2 \ \textit{s.t.}\ \mathbf{w}^\top \mathbf{x}_n \geq 1,$$

the residual is bounded and grows at most as $\|\boldsymbol{\rho}(t)\| = O(\log\log(t))$, and so

$$\lim_{t\to\infty} \frac{\mathbf{w}(t)}{\|\mathbf{w}(t)\|} = \frac{\hat{\mathbf{w}}}{\|\hat{\mathbf{w}}\|}.$$

For l-layer neural networks, we argue that the nonlinear transformation of the first $l-1$ layers is equivalent to a kernel mapping, furthermore, the kernel mapping exists and can be found. Then we have the following main result:

Theorem 1. *Let $\mathbf{X}$ be the data matrix, $\boldsymbol{\gamma}$ be the weight vector of the last hidden layer. For any neural network for binary classification, any β-smooth decreasing loss function with an exponential tail, small enough step size $\eta < 2\beta^{-1}\sigma_{\max}^{-2}(\mathbf{X})$ and any start point $\boldsymbol{\gamma}(0)$, the direction of the neural network's last hidden layer converges:*

$$\lim_{t\to\infty} \frac{\boldsymbol{\gamma}(t)}{\|\boldsymbol{\gamma}(t)\|} = \frac{\hat{\mathbf{W}}}{\|\hat{\mathbf{W}}\|},$$

where $\hat{\mathbf{W}}$ is the maximum margin solution to nonlinear SVM:

$$\hat{\mathbf{W}} = \arg\min_{\mathbf{W} \in \mathbb{R}^{p \times 1}} \|\mathbf{W}\|^2$$
$$\textit{subject to}\ \sum_{i=1}^{N} \mathbf{W} \phi(\boldsymbol{x}_n) \geq 1, \quad (3)$$

where $\phi(\boldsymbol{x}_n)$ is the re-defined input of the last hidden layer, $\phi(\cdot)$ is the implicitly introduced mapping of the kernel function $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle$.

Remark 1. *The kernel function implicitly yields a kernel feature mapping $\boldsymbol{x} \mapsto \phi(\boldsymbol{x})$, the mapped feature vector is consistent with the result of the nonlinear feature transformation of the first $l-1$ layers in deep networks, as we argued previously. Since $\phi(\boldsymbol{x})$ is the feature vector in RKHS induced by the kernel function, it can be seen from (3) that $\hat{\mathbf{W}}$ is the weight of the nonlinear SVM. Theorem 1 shows that when $t \to \infty$, the direction of the neural network's last hidden layer converges to that of the nonlinear SVM, i.e., the decision boundary of the neural network converges to that of the nonlinear SVM.*

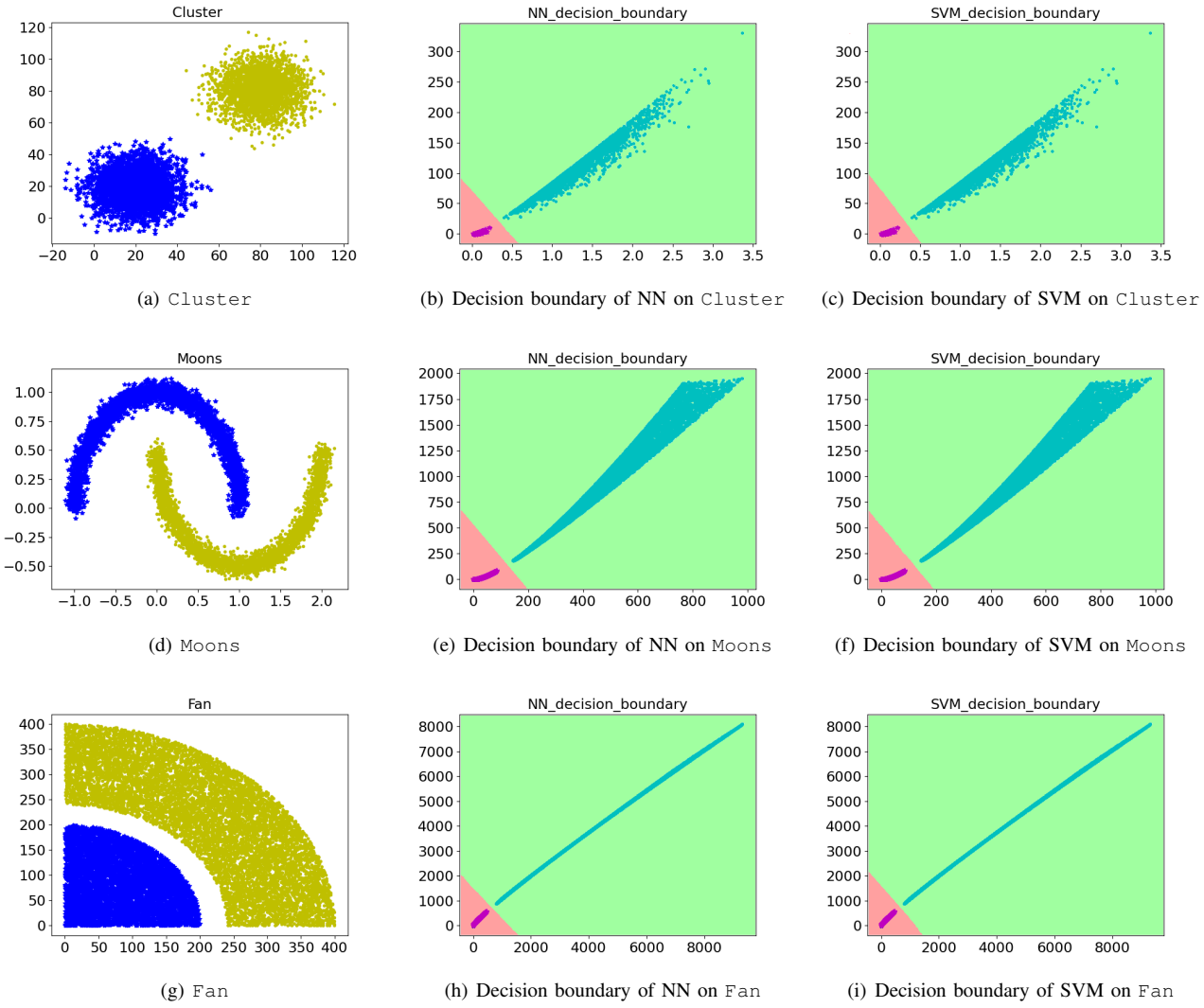
For multi-classification cases, we can consider the generalization of the logistic loss, i.e., cross-entropy loss, even exponential multi-classification loss to extend our theorem, and we can get a similar result. The elaborate proof of Theorem 1 is shown in Appendix.

V. Experiments

In this section, we experimentally demonstrate that the decision boundaries of neural networks converge to those of nonlinear SVMs. We also evaluate the performance of fully connected neural networks compared with several nonlinear SVMs with commonly used kernel functions, and analyze the experimental results.

A. Experimental Setup

We mainly conduct experiments on three simulated 2D datasets. One dataset is linearly separable, the rest ones are non-linearly separable. The size of each dataset is 6000.



(a) `Cluster` (b) Decision boundary of NN on `Cluster` (c) Decision boundary of SVM on `Cluster`

(d) `Moons` (e) Decision boundary of NN on `Moons` (f) Decision boundary of SVM on `Moons`

(g) `Fan` (h) Decision boundary of NN on `Fan` (i) Decision boundary of SVM on `Fan`

Figure 1. Results on three simulated 2D datasets. The first column illustrates the distribution of simulated datasets. The coordinates indicate locations of data points. The second and third columns show the decision boundaries of the last hidden layers of neural networks (NNs) and nonlinear SVMs, respectively. The intersection of background colors is the decision boundary. The locations of data points in the feature space have been changed after nonlinear feature transformation, so the coordinates have also been scaled. The cyan dots are test data which are predicted to be of the same class as the yellow training data, the magenta stars are test data which are predicted to be of the same class as the blue training data.

According to the priori knowledge, common kernel functions are all used to train nonlinear SVMs, mainly including the linear kernel, polynomial kernel, Gaussian kernel, and Laplace kernel. We selected the best result for comparison with neural networks. We used a 4-layer fully connected neural network with 2500 nodes in the middle fully connected layer for all experiments, Cross-entropy loss is applied to the loss function and ReLU is applied to the activation function in the neural network. The neural network is optimized by GD. The accuracy is chosen as the evaluation criteria for classification tasks, which reflect the average performance.

Since the feature vectors in RKHS may be infinite-dimensional, in experiments, we can use KPCA to obtain the feature vectors after kernel mapping and select several leading components of the kernel feature space to compute the decision boundary of nonlinear SVMs.

B. Datasets

We systematically validate our theoretical results and evaluate the performance of neural networks and nonlinear SVMs on classification tasks. The three simulated datasets used for experiments include `Cluster`, `Moons`, and `Fan`, where `Cluster` dataset is linearly separable, `Moons` and `Fan` datasets are non-linearly separable, which are shown in Fig. 1. Each dataset contains 6000 randomly generated training data, they are yellow dots and blue stars in Fig. 1. The test data are uniformly sampled across the whole space with the size of 600, they are cyan dots and magenta stars

Table I
CLASSIFICATION ACCURACY OF NEURAL NETWORKS AND NONLINEAR SVMS ON SIMULATED DATASETS

Dataset	Neural Networks	Nonlinear SVMs
Cluster	100%	100%
Round	99.35%	96.17%
Fan	99.83%	98.27%

in Fig. 1. The yellow and cyan dots have the same label, the blue and magenta stars share the same label as well.

C. Experimental Results

Experimental results are shown in Fig. 1 and Table I. The first column of Fig. 1 illustrates the distribution of simulated datasets. The coordinates indicate locations of data points. The second column shows the decision boundaries of the last hidden layers of neural networks. The third column demonstrates the decision boundaries of nonlinear SVMs. The intersection of background colors is the decision boundary. The locations of data points in the feature space have been changed after nonlinear feature transformation, so the coordinates have also been scaled. The cyan dots are test data which are predicted to be of the same classification as the yellow training data, the magenta stars are test data which are predicted to be of the same classification as the blue training data.

Table I illustrates the classification accuracy of neural networks and nonlinear SVMs on three simulated datasets. It can be seen that they both perform well on the linearly separable dataset, and nonlinear SVMs are slightly inferior to neural networks on nonlinearly separable datasets.

D. Experimental Analysis

Since our experiments are designed to verify that the decision boundaries of neural networks converge to those of nonlinear SVMs, the classification accuracy is merely used as a reference to demonstrate the relationship between the multi-layer nonlinear transformation of neural networks and the kernel feature mapping under the similar accuracy. We have reason to believe that the classification accuracy of nonlinear SVMs can be improved through multiple kernel fusion or using multi-layer (deep) kernels which are constructed by the method we have shown above or even using the general form of spectral kernels in [20] and other various deep kernels [15] [20] [17] [18], so that the kernel feature mapping will be closer to the multi-layer nonlinear transformation of deep networks, then the decision boundary convergence theorem we proposed will be more valid.

On the one hand, our experimental results illustrate the rationality of using kernel methods to analyze deep networks. The unique advantages of kernel methods in generalization ability and the mathematical analysis make it easier to analyze than deep networks, which also prompted us to use kernel methods to further investigate deep networks.

On the other hand, we cannot ignore the classification accuracy. The higher classification accuracy of neural networks motivates us to design better (deep) kernel functions and (multi-layer) kernel machines. The poor performance of nonlinear SVMs does not mean that kernel machines are not as good as neural networks, the performance of kernel machines on specific tasks may not be the same, as stated by "No Free Lunch Theorem". Besides, kernel methods have unique advantages that deep networks cannot reach in terms of generalization and interpretability [25].

VI. CONCLUSION

We analyze deep neural networks from a kernel perspective and use kernel methods to investigate the effect of the implicit regularization introduced by gradient descent on generalization ability. We argue that the multi-layer nonlinear feature transformation in deep neural networks is equivalent to a kernel feature mapping and analyze our point of view in the case of shallow and deep neural networks, respectively. Furthermore, using the representer theorem, we prove that the last hidden layers of deep neural networks converge to nonlinear SVMs. Systematical experiments demonstrate that the decision boundaries of neural networks converge to those of nonlinear SVMs as well.

Our theoretical result will inspire us to further develop new kernel theories to analyze the generalization of deep neural networks in the RKHS induced by the designed kernel and explore the construction of deep kernels. This will also drive us to design new deep kernel algorithms in combination with deep learning to further construct end-to-end MKMs with performance comparable to deep neural networks.

ACKNOWLEDGMENT

This work was supported in part by the National Natural Science Foundation of China (No. 61673293).

APPENDIX

In this section, we give the whole proof for Theorem 1.

PROOF OF THEOREM 1

Proof: For the l-th layer of the neural network, since the nonlinear transformation of the first $l-1$ layers is equivalent to a kernel mapping, the input of the l-th layer, i.e., $\mathbf{\Phi} = [\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_n)] \in \mathbb{R}^{p\times N}$, is the feature vector in Reproducing Kernel Hilbert Space, $\mathbf{\Phi} \subset \mathcal{H}$. Hence, the dataset $\{\phi(\boldsymbol{x}_n), y_n\}_{n=1}^N$ is linearly separable, i.e., $\exists \boldsymbol{\gamma}'$ such that $\forall n : {\boldsymbol{\gamma}'}^\top \phi(\boldsymbol{x}_n) > 0$.

For simplicity, we first use the exponential loss function to analyze the asymptotic behavior of the weight $\boldsymbol{\gamma}$ in RKHS during gradient descent. However, our proof process is based on β-smooth decreasing loss function instead of the exponential loss function, which ensures that our theorem holds for any β-smooth decreasing loss function, including logistic loss, exponential loss, cross-entropy loss, and so on.

If $\lim_{t\to\infty}\boldsymbol{\gamma}(t)/\|\boldsymbol{\gamma}(t)\|$ exists and converges to a certain value $\boldsymbol{\gamma}_*$, we can denote $\boldsymbol{\gamma}(t)=g(t)\boldsymbol{\gamma}_*+\boldsymbol{\rho}(t)$, where $g(t)\to\infty$, $\lim_{t\to\infty}\boldsymbol{\rho}(t)/g(t)=0$, $\forall n:{\boldsymbol{\gamma}_*}^\top\phi(\boldsymbol{x}_n)>0$. The form of the negative gradient is as follows:

$$\begin{aligned}&-\nabla\mathcal{L}(\boldsymbol{\gamma})=\sum_{n=1}^{N}\exp\left(-\boldsymbol{\gamma}(t)^\top\phi(\boldsymbol{x}_n)\right)\phi(\boldsymbol{x}_n)\\=&\sum_{n=1}^{N}\exp\left(-g(t){\boldsymbol{\gamma}_*}^\top\phi(\boldsymbol{x}_n)\right)\exp\left(-\boldsymbol{\rho}(t)^\top\phi(\boldsymbol{x}_n)\right)\phi(\boldsymbol{x}_n).\end{aligned}\tag{4}$$

Since $\forall n:{\boldsymbol{\gamma}_*}^\top\phi(\boldsymbol{x}_n)>0$ and $g(t)\to\infty$, it can be seen from (4) that only those samples which maximize exponents will be useful for the gradient, i.e., the samples that minimize the value of ${\boldsymbol{\gamma}_*}^\top\phi(\boldsymbol{x}_n)$, and these samples are the support vectors.

As the number of iterations increases, $\|\boldsymbol{\gamma}(t)\|\to\infty$, $-\nabla\mathcal{L}(\boldsymbol{\gamma})$ will become a linear combination of support vectors, $\boldsymbol{\gamma}_*$ will be dominated by these gradients, thus $\boldsymbol{\gamma}_*$ will also become a non-negative linear combination of support vectors. We scale $\boldsymbol{\gamma}_*$ and denote it as $\hat{\mathbf{W}}=\boldsymbol{\gamma}_*/\left(\min_n{\boldsymbol{\gamma}_*}^\top\phi(\boldsymbol{x}_n)\right)$, then we have

$$\begin{aligned}&\hat{\mathbf{W}}=\sum_{n=1}^{N}\alpha_n\phi(\boldsymbol{x}_n),\\\forall n:\quad&\alpha_n\geq 0,\quad\hat{\mathbf{W}}^\top\phi(\boldsymbol{x}_n)=1,\\&\alpha_n=0,\quad\hat{\mathbf{W}}^\top\phi(\boldsymbol{x}_n)>1.\end{aligned}\tag{5}$$

Since $\phi(\boldsymbol{x})$ is the data after kernel mapping, so (5) is the KKT condition of the nonlinear SVM, and $\hat{\mathbf{W}}$ is its corresponding solution. In addition, we can denote $\boldsymbol{\gamma}(t)=\hat{\mathbf{W}}\log t+\boldsymbol{\rho}(t)$.

We then need to prove the existence of the limit of $\boldsymbol{\gamma}(t)/\|\boldsymbol{\gamma}(t)\|$ and bound the residual term $\boldsymbol{\rho}(t)$, so as to show the effect of non-support vectors on gradients.

First of all, we define

$$\mathbf{r}(t)=\boldsymbol{\gamma}(t)-\hat{\mathbf{W}}\log t-\tilde{\mathbf{W}}$$

for any $\boldsymbol{\gamma}(t)$, where $\hat{\mathbf{W}}$ is the maximum margin solution to nonlinear SVM, $\tilde{\mathbf{W}}=\lim_{t\to\infty}\boldsymbol{\rho}(t)$ and $\tilde{\mathbf{W}}$ is unique. We denote $\mathcal{S}\in\mathbb{R}^{d\times S}$ as a subset of $\mathbf{X}$, whose columns are support vectors and S is the number of support vectors. Furthermore, we denote

$$\theta=\min_{\boldsymbol{x}_n\notin\mathcal{S}}\hat{\mathbf{W}}^\top\phi(\boldsymbol{x}_n)>1$$

and define $\mathbf{P}_1\in\mathbb{R}^{d\times d}$ as the orthogonal projection matrix to the subspace spanned by support vectors, $\mathbf{P}_2=\mathbf{I}-\mathbf{P}_1$ as the complementary projection.

Since $\boldsymbol{\rho}(t)=\mathbf{r}(t)+\tilde{\mathbf{W}}$, as long as we prove that $\|\mathbf{r}(t)\|$ is bounded, then $\boldsymbol{\rho}(t)$ is bounded. To prove this, we need the following lemmas [24]:

Lemma 3. *Let $\mathcal{L}(\mathbf{w})$ be a β-smooth non-negative objective. If $\eta<2\beta^{-1}$, then, for any $\mathbf{w}(0)$ with the gradient descent iterative steps:*

$$\mathbf{w}(t+1)=\mathbf{w}(t)-\eta\nabla\mathcal{L}(\mathbf{w}(t)),$$

we have that $\sum_{u=0}^{\infty}\|\nabla\mathcal{L}(\mathbf{w}(u))\|^2<\infty$ and therefore $\lim_{t\to\infty}\|\nabla\mathcal{L}(\mathbf{w}(t))\|^2=0$.

Lemma 4. *$\exists C_1,t_1,\forall t>t_1$, we have*

$$(\mathbf{r}(t+1)-\mathbf{r}(t))^\top\mathbf{r}(t)\leq C_1t^{-\min(\theta,-1-1.5\mu_+,-1-0.5\mu_-)}.$$

In addition, $\forall\epsilon_1>0,t>t_2$, $\exists C_2,t_2$, if $\|\mathbf{P}_1\mathbf{r}(t)\|\geq\epsilon_1$, then we have the following formula:

$$(\mathbf{r}(t+1)-\mathbf{r}(t))^\top\mathbf{r}(t)\leq-C_2t^{-1}<0,$$

where $C_1,C_2,t_1,t_2,\epsilon_1,\mu_+,\mu_-$ are all positive constants and independent of t.

Then, we will bound the following equation:

$$\begin{aligned}&\|\mathbf{r}(t+1)\|^2\\=&\|\mathbf{r}(t+1)-\mathbf{r}(t)\|^2+2(\mathbf{r}(t+1)-\mathbf{r}(t))^\top\mathbf{r}(t)+\|\mathbf{r}(t)\|^2.\end{aligned}\tag{6}$$

We will bound the terms of the equation separately. According to (2) and the following formulas:

$$\begin{aligned}\forall x>0:x\geq\log(1+x)>0,\\\log(t+1)-\log(t)\approx\frac{1}{t},\\\ell'(u)\leq 0,\hat{\mathbf{W}}^\top\phi(\boldsymbol{x}_n)\geq 1,\end{aligned}$$

we have

$$\hat{\mathbf{W}}^\top\nabla\mathcal{L}(\boldsymbol{\gamma}(t))=\sum_{n=1}^{N}\ell'\left(\boldsymbol{\gamma}(t)^\top\phi(\boldsymbol{x}_n)\right)\hat{\mathbf{W}}^\top\phi(\boldsymbol{x}_n)\leq 0,$$

so

$$\begin{aligned}&\|\mathbf{r}(t+1)-\mathbf{r}(t)\|^2\\=&\|\boldsymbol{\gamma}(t+1)-\hat{\mathbf{W}}\log(t+1)-\tilde{\mathbf{W}}-\boldsymbol{\gamma}(t)+\hat{\mathbf{W}}\log(t)+\tilde{\mathbf{W}}\|^2\\=&\|-\eta\nabla\mathcal{L}(\boldsymbol{\gamma}(t))-\hat{\mathbf{W}}[\log(t+1)-\log(t)]\|^2\\=&\eta^2\|\nabla\mathcal{L}(\boldsymbol{\gamma}(t))\|^2+2\eta\hat{\mathbf{W}}^\top\nabla\mathcal{L}(\boldsymbol{\gamma}(t))[\log(t+1)-\log(t)]+\\&\|\hat{\mathbf{W}}\|^2\left(\log(t+1)-\log(t)\right)^2\\\leq&\eta^2\|\nabla\mathcal{L}(\boldsymbol{\gamma}(t))\|^2+\|\hat{\mathbf{W}}\|^2\frac{1}{t^2}.\end{aligned}\tag{7}$$

According to Lemma 3, we have

$$\|\nabla\mathcal{L}(\boldsymbol{\gamma}(t))\|^2=o(1),\ \sum_{t=0}^{\infty}\|\nabla\mathcal{L}(\boldsymbol{\gamma}(t))\|^2<\infty.\tag{8}$$

Since $\frac{1}{t^2}$ is convergent, substituting (8) into (7), we know that $\exists C_0\in\mathbb{R}$:

$$\begin{aligned}\|\mathbf{r}(t+1)-\mathbf{r}(t)\|^2=o(1),\\\sum_{t=0}^{\infty}\|\mathbf{r}(t+1)-\mathbf{r}(t)\|^2=C_0<\infty,\end{aligned}\tag{9}$$

which means that $\forall \epsilon_0$, $\exists t_0$ such that

$$\forall t > t_0 : |\|\mathbf{r}(t+1)\| - \|\mathbf{r}(t)\|| < \epsilon_0.$$

For the second term in (6), according to Lemma 4, we know that $\exists C_1, t_1$, $\forall t > t_1$:

$$\begin{aligned}&(\mathbf{r}(t+1) - \mathbf{r}(t))^\top \mathbf{r}(t)\\ \leq& C_1 t^{-\min(\theta, -1-1.5\mu_+, -1-0.5\mu_-)},\end{aligned} \tag{10}$$

where $\theta = \min_{\boldsymbol{x}_n \notin \mathcal{S}} \hat{\mathbf{W}}^\top \phi(\boldsymbol{x}_n) > 1$.

Substituting (9) and (10) into (6), we know that $\|\mathbf{r}(t+1)\|^2 - \|\mathbf{r}(t)\|^2$ is bounded, then

$$\begin{aligned}&\|\mathbf{r}(t)\|^2 - \|\mathbf{r}(t_1)\|^2 = \sum_{u=t_1}^{t-1} \left(\|\mathbf{r}(u+1)\|^2 - \|\mathbf{r}(u)\|^2\right)\\ \leq& C_0 + 2\sum_{u=t_1}^{t-1} C_1 u^{-\min(\theta, -1-1.5\mu_+, -1-0.5\mu_-)}\end{aligned}$$

is bounded. Hence, $\|\mathbf{r}(t)\|$ is bounded. So far, we have completed the proof. ■

REFERENCES

[1] J. Wang, K. Feng, and J. Wu, "Svm-based deep stacking networks," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 2019, pp. 5273–5280.

[2] A. Bietti, G. Mialon, D. Chen, and J. Mairal, "A kernel perspective for regularizing deep neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 664–674.

[3] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *Proceedings of the 5th International Conference on Learning Representations*, 2017.

[4] M. Belkin, S. Ma, and S. Mandal, "To understand deep learning we need to understand kernel learning," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 540–548.

[5] B. Schölkopf and A. J. Smola, *Learning with Kernels: support vector machines, regularization, optimization, and beyond*. MIT Press, 2002.

[6] B. Scholkopf, S. Mika, C. J. C. Burges, P. Knirsch, K. . Muller, G. Ratsch, and A. J. Smola, "Input space versus feature space in kernel-based methods," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1000–1017, 1999.

[7] K. . Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf, "An introduction to kernel-based learning algorithms," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 181–201, 2001.

[8] M. Anthony and P. L. Bartlett, *Neural Network Learning - Theoretical Foundations*. Cambridge University Press, 2002.

[9] G. Raskutti, M. J. Wainwright, and B. Yu, "Early stopping and non-parametric regression: an optimal data-dependent stopping rule," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 335–366, 2014.

[10] B. Bohn, C. Rieger, and M. Griebel, "A representer theorem for deep kernel learning," *Journal of Machine Learning Research*, vol. 20, no. 64, pp. 1–32, 2019.

[11] T. Suzuki, "Fast generalization error bound of deep learning from a kernel perspective," in *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, 2018, pp. 1397–1406.

[12] Y. Li, L. Ding, and X. Gao, "On the decision boundary of deep neural networks," *arXiv:1808.05385v3*, 2019.

[13] S. Kornblith, M. Norouzi, H. Lee, and G. E. Hinton, "Similarity of neural network representations revisited," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 3519–3529.

[14] G. Montavon, M. L. Braun, and K. Müller, "Kernel analysis of deep networks," *Journal of Machine Learning Research*, vol. 12, pp. 2563–2581, 2011.

[15] Y. Cho and L. K. Saul, "Kernel methods for deep learning," in *Advances in Neural Information Processing Systems 22*, 2009, pp. 342–350.

[16] T. Hazan and T. S. Jaakkola, "Steps toward deep kernel methods from infinite neural networks," *arXiv:1508.05133v2*, 2015.

[17] J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein, "Deep neural networks as gaussian processes," in *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[18] J. Mairal, P. Koniusz, Z. Harchaoui, and C. Schmid, "Convolutional kernel networks," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 2627–2635.

[19] J. Mairal, "End-to-end kernel learning with supervised convolutional kernel networks," in *Advances in Neural Information Processing Systems 29*, 2016, pp. 1399–1407.

[20] H. Xue, Z. Wu, and W. Sun, "Deep spectral kernel learning," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019, pp. 4019–4025.

[21] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[22] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American mathematical society*, vol. 68, no. 3, pp. 337–404, 1950.

[23] A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, "Parsimonious online learning with kernels via sparse projections in function space," *Journal of Machine Learning Research*, vol. 20, no. 3, pp. 1–44, 2019.

[24] D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, and N. Srebro, "The implicit bias of gradient descent on separable data," *Journal of Machine Learning Research*, vol. 19, no. 70, pp. 1–57, 2018.

[25] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, vol. 3, pp. 463–482, 2002.