Non-parametric Representation Learning with Kernels

Pascal Esser*, Maximilian Fleissner*, Debarghya Ghoshdastidar

Technical University of Munich, Germany
esser@cit.tum.de, fleissner@cit.tum.de, ghoshdas@cit.tum.de

Abstract

Unsupervised and self-supervised representation learning has become popular in recent years for learning useful features from unlabelled data. Representation learning has been mostly developed in the neural network literature, and other models for representation learning are surprisingly unexplored. In this work, we introduce and analyze several kernel-based representation learning approaches: Firstly, we define two kernel Self-Supervised Learning (SSL) models using contrastive loss functions and secondly, a Kernel Autoencoder (AE) model based on the idea of embedding and reconstructing data. We argue that the classical representer theorems for supervised kernel machines are not always applicable for (self-supervised) representation learning, and present new representer theorems, which show that the representations learned by our kernel models can be expressed in terms of kernel matrices. We further derive generalisation error bounds for representation learning with kernel SSL and AE, and empirically evaluate the performance of these methods in both small data regimes as well as in comparison with neural network based models.

Introduction

Representation learning builds on the idea that for most data, there exists a lower dimensional embedding that still retains most of the information useful for a downstream task (Bengio, Courville, and Vincent 2013). While early works relied on pre-defined representations, including image descriptors such as SURF (Bay, Tuytelaars, and Van Gool 2006) or SIFT (Lowe 1999) as well as bag-of-words approaches, over the past decade the focus has moved to representations learned from data itself. Across a wide range of tasks, including image classification and natural language processing (Bengio, Courville, and Vincent 2013), this approach has proven to be more powerful than the use of hand-crafted descriptors. In addition, representation learning has gained increasing popularity in recent years as it provides a way of taking advantage of unlabelled data in a partially labelled data setting. Since the early works, methods for representation learning have predominantly relied on neural networks and there has been little focus on other classes of models. This may be part of the reason why it is still mostly driven from an experimental perspective. In this work, we focus on the following two learning paradigms that both fall under the umbrella of representation learning:

Self-Supervised representation learning using contrastive loss functions has been established in recent years as an important method between supervised and unsupervised learning as it does not require explicit labels but relies on implicit knowledge of what makes samples semantically close to others. Therefore SSL builds on inputs and inter-sample relations $(\boldsymbol{X}, \overline{\boldsymbol{X}})$, where $\overline{\boldsymbol{X}}$ is often constructed through data-augmentations of $\boldsymbol{X}$ known to preserve input semantics such as additive noise or horizontal flip for an image (Kanazawa, Jacobs, and Chandraker 2016). While the idea of SSL is not new (Bromley et al. 1993) the main focus has been on deep SSL models, which have been highly successful in domains such as computer vision (Chen et al. 2020; Jing and Tian 2019) and natural language processing (Misra and Maaten 2020; Devlin et al. 2019).

Unsupervised representation learning through reconstruction relies only on a set of features $\boldsymbol{X}$ without having access to the labels. The high level idea is to map the data to a lower dimensional latent space, and then back to the features. The model is optimised by minimising the difference between the input data and the reconstruction. This has been formalized through principal component analysis (PCA) (Pearson 1901) and its nonlinear extension Kernel PCA (Schölkopf, Smola, and Müller 1998). While few approaches exist in traditional machine learning, the paradigm of representation through reconstruction has built the foundation of a large number of deep learning methods. Autoencoders (AE) (Kramer 1991) use a neural network for both the embedding into the latent space as well as for the reconstruction. The empirical success of autoencoders has given rise to a large body of work, developed for task specific regularisation (e.g. (Yang et al. 2017)), as well as for a wide range of applications such as image denoising (Buades, Coll, and Morel 2005a), clustering (Yang et al. 2017) or natural language processing (Zhang et al. 2022). However, their theoretical understanding is still limited to analyzing critical points and dynamics in shallow linear networks (Kunin et al. 2019; Pretorius, Kroon, and Kamper 2018; Refinetti and Goldt 2022).

*These authors contributed equally.

An extended version including proofs can be found at https://arxiv.org/abs/2309.02028

Kernel representation learning. In spite of the widespread use of deep learning, other models are still ubiquitous in data science. For instance, decision tree ensembles are competitive with neural networks in various domains (Shwartz-Ziv and Armon 2022; Roßbach 2018; Gu, Kelly, and Xiu 2020), and are preferred due to interpretability. Another well established approach is kernel methods, which we will focus on in this paper. At an algorithmic level, kernel methods rely on the pairwise similarities between datapoints, denoted by a kernel $k(\boldsymbol{x}, \boldsymbol{x}')$. When the map k is positive definite, $k(\boldsymbol{x}, \boldsymbol{x}')$ corresponds to the inner product between (potentially infinite-dimensional) nonlinear transformations of the data, and implicitly maps the data to a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ through a feature map $\phi : \mathcal{X} \to \mathcal{H}$ that satisfies $k(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle$. Thus, any algorithm that relies exclusively on inner products can implicitly be run in $\mathcal{H}$ by simply evaluating the kernel k. Kernel methods are among the most successful models in machine learning, particularly due to their inherently non-linear and non-parametric nature, that nonetheless allows for a sound theoretical analysis. Kernels have been used extensively in regression (Kimeldorf and Wahba 1971; Wahba 1990) and classification (Cortes and Vapnik 1995; Mika et al. 1999). Since representation learning, or finding suitable features, is a key challenge is many scientific fields, we believe there is considerable scope for developing such models in these fields. *The goal of this paper is to establish that one can construct non-parametric representation learning models, based on data reconstruction and contrastive losses.* By reformulating the respective optimisation problems for such models using positive definite kernels (Aronszajn 1950; Schölkopf and Smola 2002), we implicitly make use of non-linear feature maps $\phi(\cdot)$. Moreover, the presented approaches do not reduce to traditional (unsupervised) kernel methods. In the reconstruction based setting we define a Kernel AE and also present kernel-based self-supervised methods by considering two different contrastive loss functions. Thereby, our work takes a significant step towards this development by decoupling the representation learning paradigm from deep learning. To this end, kernel methods are an ideal alternative since (i) kernel methods are suitable for small data problems that are prevalent in many scientific fields (Xu et al. 2023; Todman, Bush, and Hood 2023; Chahal and Toner 2021); (ii) kernels are non-parametric, and yet considered to be quite interpretable (Ponte and Melko 2017; Hainmueller and Hazlett 2014); and (iii) as we show, there is a natural translation from deep SSL to kernel SSL, without compromising performance.

Contributions. The main contributions of this work is the development and analysis of kernel methods for reconstruction and contrastive SSL models. More specifically:

1. *Kernel Contrastive Learning.* We present *kernel variants of a single hidden layer network that minimises two popular contrastive losses.* For a simple contrastive loss (Saunshi et al. 2019), the optimisation is closely related to a kernel eigenvalue problem, while we show that the minimisation of *spectral contrastive loss* (HaoChen et al. 2021) in the kernel setting can be rephrased as a kernel matrix based optimisation.
2. *Kernel Autoencoder.* We present a Kernel AE where the encoder learns a low-dimensional representation. We show that a *Kernel AE* can be learned by solving a kernel matrix based optimisation problem.
3. *Theory.* We present an extension to the existing representer theorem under orthogonal constraints. Furthermore we derive generalisation error bounds for the proposed kernel models in which show that the prediction of the model improve with increased number of unlabelled data.
4. *Experiments.* We empirically demonstrate that the three proposed kernel methods perform on par or outperform classification on the original features as well as Kernel PCA and compare them to neural network representation learning models.

Related Work (Johnson, Hanchi, and Maddison 2022) show that minimising certain contrastive losses can be interpreted as learning kernel functions that approximate a fixed positive-pair kernel, and hence, propose an approach of combining deep SSL with Kernel PCA. Closer to our work appears to be (Kiani et al. 2022), where the neural network is replaced by a function learned on the RKHS of a kernel. However, their loss functions are quite different from ours. Moreover, by generalising the representer theorem, we can also enforce orthonormality on the embedding maps from the RKHS itself. (Zhai et al. 2023) studies the role of augmentations in SSL through the lense of the RKHS induced by an augmentation. (Shah et al. 2022) present a margin maximisation approach for contrastive learning that can be solved using kernel support vector machines. Their approach is close to our simple contrastive loss method (Definition 1), but not the same as we obtain a kernel eigenvalue problem. While (Johnson, Hanchi, and Maddison 2022; Shah et al. 2022) consider specific contrastive losses, we present a wider range of kernel SSL models, including Kernel AE, and provide generalisation error bounds for all proposed models.

Notation We denote matrices by bold capital letters $\boldsymbol{A}$, vectors as $\boldsymbol{a}$, and $\boldsymbol{I}_m$ for an identity matrix of size $m \in \mathbb{N}$. For a given kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, we denote $\phi : \mathbb{R}^d \to \mathcal{H}$ for its canonical feature map into the associated RKHS $\mathcal{H}$. Given data $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ collected in a matrix $\boldsymbol{X} \in \mathbb{R}^{d \times n}$, we write $\Phi := (\phi(\boldsymbol{x}_1), \dots, \phi(\boldsymbol{x}_n))$ and define $\mathcal{H}_X$ as the finite-dimensional subspace spanned by Φ. Recall that $\mathcal{H}$ can be decomposed as $\mathcal{H}_X \oplus \mathcal{H}_X^{\perp}$. We denote by $\boldsymbol{K} = \Phi^T \Phi \in \mathbb{R}^{n \times n}$ the kernel matrix, and define $k(\boldsymbol{x}', \boldsymbol{X}) = \Phi^T \phi(\boldsymbol{x}')$. Throughout the paper we assume n datapoints are used to train the representation learning model, which embeds from $\mathbb{R}^d$ into a h-dimensional space. On a formal level, the problem could be stated within the generalised framework of matrix-valued kernels $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{h \times h}$, because the vector-valued RKHS $\mathcal{H}(K)$ associated with a matrix-valued kernel K naturally contains functions $\boldsymbol{W}$ that map from $\mathbb{R}^d$ to $\mathbb{R}^h$. For the scope of this paper however, it is sufficient to assume $K(x, y) = \boldsymbol{I}_h \cdot k(x, y)$ for some scalar kernel $k(x, y)$ with real-valued RKHS $\mathcal{H}$. Then, the norm of any $\boldsymbol{W} \in \mathcal{H}(K)$ is simply the Hilbert-Schmidt norm that we denote as $\|\boldsymbol{W}\| = \|\boldsymbol{W}\|_{\mathcal{H}}$ (for finite-dimensional matrices, the Frobenius norm), and learning the embedding from $\mathbb{R}^d$ to $\mathbb{R}^h$ reduces to learning h individual vectors $\boldsymbol{w}_1, \dots, \boldsymbol{w}_h \in \mathcal{H}$. In other words, we can

interpret $\boldsymbol{W} \in \mathcal{H}(K)$ as a (potentially infinite-dimensional) matrix with columns $\boldsymbol{w}_1, \dots, \boldsymbol{w}_h \in \mathcal{H}$, sometimes writing $\boldsymbol{W} = (\boldsymbol{w}_1, \dots, \boldsymbol{w}_h)$ for notational convenience. To underline the similarity with the deep learning framework, we denote $\boldsymbol{W}^T\phi(x) = (\langle \boldsymbol{w}_t, \phi(x)\rangle)_{t=1}^h = (\boldsymbol{w}_1(x), \dots, \boldsymbol{w}_h(x)) \in \mathbb{R}^h$, where we invoke the reproducing property of the RKHS $\mathcal{H}$ in the last step. We denote by $\boldsymbol{W}^*$ the adjoint operator of $\boldsymbol{W}$ (in a finite-dimensional setting, $\boldsymbol{W}^*$ simply becomes the transpose $\boldsymbol{W}^T$). The constraint $\boldsymbol{W}^*\boldsymbol{W} = \boldsymbol{I}_h$ enforces orthonormality between all pairs $(\boldsymbol{w}_i, \boldsymbol{w}_j)_{i,j\le h}$.

Representer Theorems

In principle, kernel methods minimise a loss functional $\mathcal{L}$ over the entire, possibly infinite-dimensional RKHS. It is the celebrated representer theorem (Kimeldorf and Wahba 1971; Schölkopf, Herbrich, and Smola 2001) that ensures the practical feasibility of this approach: Under mild conditions on the loss $\mathcal{L}$, the optimiser is surely contained within the finite-dimensional subspace $\mathcal{H}_X$. For example, in standard kernel ridge regression, the loss functional $\mathcal{L}$ is simply the regularised empirical squared error

$$\mathcal{L}(\boldsymbol{w}) = \sum_{i=1}^{n} (\boldsymbol{w}(x_i) - y_i)^2 + \lambda\|\boldsymbol{w}\|$$

The fact that all minimisers of this problem indeed lie in $\mathcal{H}_X$ can be seen by simply decomposing $\mathcal{H} = \mathcal{H}_X \oplus \mathcal{H}_X^\perp$, observing that $\boldsymbol{w}(x_i) = 0$ for all $\boldsymbol{w} \in \mathcal{H}_X^\perp$, and concluding that projecting any $\boldsymbol{w}$ onto $\mathcal{H}$ can only ever decrease the functional $\mathcal{L}$. This very argument can be extended to representation learning, where regularisation is important to avoid mode collapse. We formally state the following result.

Theorem 1. *(Representer Theorem for Representation Learning) Given data $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$, denote by $\mathcal{L}_X(\boldsymbol{w}_1, \dots, \boldsymbol{w}_h)$ a loss functional on $\mathcal{H}^h$ that does not change whenever $\boldsymbol{w}_1, \dots, \boldsymbol{w}_h$ are projected onto the finite-dimensional subspace $\mathcal{H}_X$ spanned by the data. Then, any minimiser of the regularised loss functional*

$$\mathcal{L}(\boldsymbol{w}_1, \dots, \boldsymbol{w}_h) = \mathcal{L}_X(\boldsymbol{w}_1, \dots, \boldsymbol{w}_h) + \lambda\|\boldsymbol{W}\|_{\mathcal{H}}$$

consists of $\boldsymbol{w}_1, \dots, \boldsymbol{w}_h \in \mathcal{H}_X$.

This justifies the use of kernel methods when the norm of the embedding map is penalized. However, it does not address loss functionals $\mathcal{L}$ that instead impose an orthonormality constraint on the embedding $\boldsymbol{W}$. It is natural to ask when a representer theorem exist for these settings as well. Below, we give a necessary and sufficient condition.

Theorem 2 (Representer theorem under orthonormality constraints). *Given data $\boldsymbol{X}$ and an embedding dimension $h \in \mathbb{N}$, let $\mathcal{L} : \mathcal{H}^h \to \mathbb{R}$ be a loss function that vanishes on $\mathcal{H}_X^\perp$. Assume $\dim(\mathcal{H}_X^\perp) \ge h$. Consider the following constrained minimisation problem over $\boldsymbol{w}_1, \dots, \boldsymbol{w}_h \in \mathcal{H}$*

$$\begin{aligned} &\text{minimise } \mathcal{L}(\boldsymbol{w}_1, \dots, \boldsymbol{w}_h) \\ &\text{s.t. } \boldsymbol{W}^*\boldsymbol{W} = \boldsymbol{I}_h \end{aligned} \tag{1}$$

Furthermore, consider the inequality-constrained problem over $\mathcal{H}_X$

$$\begin{aligned} &\text{minimise } \mathcal{L}(\boldsymbol{w}_1, \dots, \boldsymbol{w}_h) \\ &\text{s.t. } \boldsymbol{W}^T\boldsymbol{W} \preceq \boldsymbol{I}_h \text{ and } \boldsymbol{w}_1, \dots, \boldsymbol{w}_h \in \mathcal{H}_X \end{aligned} \tag{2}$$

Then, every minimiser of (1) *is contained in $\mathcal{H}_X^h$ if and only if every minimiser of* (2) *satisfies $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_h$.*

In practice, the conditions (2) can often be verified directly by checking the gradient of $\mathcal{L}$ on $\mathcal{H}_X$, or under orthonormalization (see Appendix). Together with the standard representer theorem, this guarantees that kernel methods can indeed be extended to representation learning — without sacrificing the appealing properties that the representer theorem provides us with.

Representation Learning with Kernels

Building on this foundation, we can now formalize the previously discussed representation learning paradigms in the kernel setting — namely SSL using contrastive loss functions, as well as unsupervised learning through reconstruction loss.

Simple Contrastive Loss

For convenience, we restrict ourselves to a triplet setting with training samples $(\boldsymbol{x}_i, \boldsymbol{x}_i^+, \boldsymbol{x}_i^-), i = 1, \dots, n$. The idea is to consider an anchor image $\boldsymbol{x}_i$, a positive sample $\boldsymbol{x}_i^+$ generated using data augmentation techniques, as well as an independent negative sample $\boldsymbol{x}_i^-$. The goal is to align the anchor more with the positive sample than with the independent negative sample. In the following, we consider two loss functions that implement this idea.

In both cases, we kernelize a single hidden layer, mapping data $\boldsymbol{x} \in \mathbb{R}^d$ to an embedding $\boldsymbol{z} \in \mathbb{R}^h$.

$$\boldsymbol{x} \in \mathbb{R}^d \xrightarrow{\phi(\cdot)} \boldsymbol{r} \in \mathcal{H} \xrightarrow{\boldsymbol{W}} \boldsymbol{z} \in \mathbb{R}^h. \tag{3}$$

We start with a simple contrastive loss inspired by (Saunshi et al. 2019), with additional regularisation. Intuitively, this loss directly compares the difference in alignment between the anchor and the positive an the anchor and the negative sample. Formally, we define it as follows.

Definition 1 (Contrastive Kernel Learning). We learn a representation of the form $f_{\boldsymbol{W}}(\boldsymbol{x}) = \boldsymbol{W}^T\phi(\boldsymbol{x})$ (see mapping in Eq. 3) by optimising the objective function

$$\begin{aligned} \mathcal{L} \;&:= \sum_{i=1}^{n} f_{\boldsymbol{W}}(\boldsymbol{x}_i)^T \left(f_{\boldsymbol{W}}(\boldsymbol{x}_i^-) - f_{\boldsymbol{W}}(\boldsymbol{x}_i^+)\right) \\ &\text{s.t. } \boldsymbol{W}^*\boldsymbol{W} = \boldsymbol{I}_h \end{aligned} \tag{4}$$

By verifying the conditions of Theorem 2, we reduce the problem to a finite-dimensional optimisation. Theorem 3 then provides a closed from solution to the optimisation problem in Eq. 4.

Theorem 3 (Closed Form Solution and Inference at Optimal parameterization). *Consider the optimisation problem as stated in Definition 1. Let $\boldsymbol{X}, \boldsymbol{X}^+, \boldsymbol{X}^- \in \mathbb{R}^{d\times n}$ denote the data corresponding to the anchors, positive and negative samples, respectively. Define the kernel matrices*

$$\begin{aligned} \boldsymbol{K} &= \left[k(x_i, x_j)\right]_{i,j} & \boldsymbol{K}_- &= \left[k(x_i, x_j^-)\right]_{i,j} \\ \boldsymbol{K}_+ &= \left[k(x_i, x_j^+)\right]_{i,j} & \boldsymbol{K}_{--} &= \left[k(x_i^-, x_j^-)\right]_{i,j} \\ \boldsymbol{K}_{++} &= \left[k(x_i^+, x_j^+)\right]_{i,j} & \boldsymbol{K}_{-+} &= \left[k(x_i^-, x_j^+)\right]_{i,j} \end{aligned}$$

Furthermore, define the matrices $\boldsymbol{K}_3 = \boldsymbol{K}_- - \boldsymbol{K}_+$ *as well as*

$$\boldsymbol{K}_\Delta = \boldsymbol{K}_{--} + \boldsymbol{K}_{++} - \boldsymbol{K}_{-+} - \boldsymbol{K}_{-+}^T \quad \boldsymbol{K}_1 = \begin{bmatrix} \boldsymbol{K} & \boldsymbol{K}_3 \\ \boldsymbol{K}_3 & \boldsymbol{K}_\Delta \end{bmatrix}$$

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{K}_3 \\ \boldsymbol{K}_\Delta \end{bmatrix} \cdot \begin{bmatrix} \boldsymbol{K} & \boldsymbol{K}_- - \boldsymbol{K}_+ \end{bmatrix} \qquad \boldsymbol{K}_2 = -\frac{1}{2}\left(\boldsymbol{B} + \boldsymbol{B}^T\right).$$

Let $\boldsymbol{A}_2$ *consist of the top* h *eigenvectors of the matrix* $\boldsymbol{K}_1^{-1/2}\boldsymbol{K}_2\boldsymbol{K}_1^{-1/2}$*, which we assume to have* h *non-negative eigenvalues. Let* $\boldsymbol{A} = \boldsymbol{K}_1^{-1/2}\boldsymbol{A}_2$*. Then, at optimal parameterization, the embedding of any* $x^* \in \mathbb{R}^d$ *can be written in closed form as*

$$\boldsymbol{z}^* = \boldsymbol{A}^T \begin{bmatrix} k(x^*, \boldsymbol{X}) \\ k(x^*, \boldsymbol{X}^-) - k(x^*, \boldsymbol{X}^+) \end{bmatrix}$$

Spectral Contrastive Loss

Let us now consider a kernel contrastive learning based on an alternative, commonly used spectral contrastive loss function (HaoChen et al. 2021).

Definition 2 (Spectral Kernel Learning). We learn a representation of the form $f_{\boldsymbol{W}}(\boldsymbol{x}) = \boldsymbol{W}^T\phi(\boldsymbol{x})$ (see mapping in Eq. 3) by optimising the following objective function, $\mathcal{L}$:

$$\mathcal{L} = \sum_{i=1}^{n} -2 f_{\boldsymbol{W}}(\boldsymbol{x}_i)^T f_{\boldsymbol{W}}(\boldsymbol{x}_i^+) + \left(f_{\boldsymbol{W}}(\boldsymbol{x}_i)^T f_{\boldsymbol{W}}(\boldsymbol{x}_i^-)\right)^2 + \lambda \left\|\boldsymbol{W}\right\|_{\mathcal{H}}^2.$$

For universal kernels, we can directly rewrite the loss function using the kernel trick and optimise it using simple gradient descent. This allows us to state the following result, which yields an optimisation directly in terms of the embeddings $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n \in \mathbb{R}^h$.

Theorem 4 (Gradients and Inference at Optimal Parameterization). *Consider the optimisation problem as stated in Definition 2, with* $\boldsymbol{K}$ *denoting the kernel matrix of a universal kernel. Then, we can equivalently minimise the objective w.r.t. the embeddings* $\boldsymbol{Z} \in \mathbb{R}^{h\times 3n}$*. Denoting by* $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{3n}$ *the columns of* $\boldsymbol{Z}$*, the loss to be minimised becomes*

$$\min_{\boldsymbol{Z}\in\mathbb{R}^{h\times 3n}} \sum_{i=1}^{n} -2\boldsymbol{z}_i^T\boldsymbol{z}_{i+n} + \left(\boldsymbol{z}_i^T\boldsymbol{z}_{i+2n}\right)^2 + \lambda \cdot \mathrm{Tr}\left(\boldsymbol{Z}\boldsymbol{K}^{-1}\boldsymbol{Z}^T\right)$$

The gradient of the loss function in terms of $\boldsymbol{Z}$ *is therefore given by*

$$2\lambda\boldsymbol{Z}\boldsymbol{K}^{-1} + \begin{cases} -2\boldsymbol{z}_{i+n} + 2(\boldsymbol{z}_i^T\boldsymbol{z}_{i+2n})\boldsymbol{z}_{i+2n} & , i \in [n] \\ -2\boldsymbol{z}_{i-n} & , i \in [n+1, 2n] \\ 2(\boldsymbol{z}_i^T\boldsymbol{z}_{i-2n})\boldsymbol{z}_{i-2n} & , i \in [2n+1, 3n] \end{cases}$$

For any new point $\boldsymbol{x}^* \in \mathbb{R}^d$*, the trained model maps it to*

$$\boldsymbol{z}^* := \boldsymbol{Z}\boldsymbol{K}^{-1}k(\boldsymbol{X}, \boldsymbol{x}^*).$$

Kernel Autoencoders

In general, AE architectures involve mapping the input to a lower dimensional latent space (encoding), and then back to the reconstruction (decoding). In this work we propose a Kernel AE, where both encoder and decoder correspond to kernel machines, resulting in the mapping

$$\boldsymbol{x} \in \mathbb{R}^d \xrightarrow{\phi_1(\cdot)} \boldsymbol{r}_1 \in \mathcal{H}_1 \xrightarrow{\boldsymbol{W}_1} \boldsymbol{z} \in \mathbb{R}^h \xrightarrow{\phi_2(\cdot)} \boldsymbol{r}_2 \in \mathcal{H}_2 \xrightarrow{\boldsymbol{W}_2} \boldsymbol{x} \in \mathbb{R}^d$$

where typically $h < d$. While several materializations of this high-level idea come to mind, we define the Kernel AE as follows.

Definition 3 (Kernel AE). Given data $\boldsymbol{X} \in \mathbb{R}^{d\times n}$ and a regularisation parameter $\lambda > 0$, define the loss functional

$$\mathcal{L}(\boldsymbol{W}_1, \boldsymbol{W}_2) := \left\|\boldsymbol{X} - \boldsymbol{W}_2^T\phi_2\left(\boldsymbol{W}_1^T\phi_1(\boldsymbol{X})\right)\right\|_{\mathcal{H}}^2 + \lambda\left(\left\|\boldsymbol{W}_1\right\|_{\mathcal{H}}^2 + \left\|\boldsymbol{W}_2\right\|_{\mathcal{H}}^2\right)$$

The Kernel AE corresponds to the optimisation problem

$$\begin{aligned} \min_{\boldsymbol{W}_1, \boldsymbol{W}_2} \ & \mathcal{L}(\boldsymbol{W}_1, \boldsymbol{W}_2) \\ \text{s.t.} \ & \left\|\boldsymbol{W}_1^T\phi(\boldsymbol{x}_i)\right\|^2 = 1 \ \forall\, i \in [n] \end{aligned} \tag{5}$$

Let us justify our choice of architecture briefly. Firstly, we include norm regularisations on both the encoder as well as the decoder. This is motivated by the following observation: When the feature map ϕ_2 maps to the RKHS of a universal kernel, **any** choice of n distinct points $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ in the bottleneck allows for perfect reconstruction. We therefore encourage the Kernel AE to learn smooth maps by penalizing the norm in the RKHS. In addition, we include the constraint $\left\|\boldsymbol{W}_1^T\phi(\boldsymbol{x}_i)\right\|^2 = 1 \ \forall\, i \in [n]$ to prevent the Kernel AE from simply pushing the points $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ to zero. This happens whenever the impact of rescaling $\boldsymbol{z}_i$ affects the norm of the encoder $\boldsymbol{W}_1$ differently from the decoder $\boldsymbol{W}_2$ (as is the case for commonly used kernels such as Gaussian and Laplacian). Nonetheless, we stress that other choices of regularisation are also possible, and we explore some of them in the Appendix.

While a closed form solution of Definition 3 is difficult to obtain, we show that for universal kernels, the optimisation can again be rewritten in terms of kernel matrices.

Theorem 5 (Kernel formulation and inference at optimal parameterization). *For any bottleneck* $\boldsymbol{Z} \in \mathbb{R}^{h\times n}$*, define the reconstruction*

$$\boldsymbol{Q}(\boldsymbol{Z}) = \boldsymbol{X}(\boldsymbol{K}_Z + \lambda I_n)^{-1}\boldsymbol{K}_Z$$

For universal kernels, learning the Kernel AE from Definition 3 is then equivalent to minimising the following expression over all possible embeddings $\boldsymbol{Z} \in \mathbb{R}^{h\times n}$*:*

$$\begin{aligned} & \left\|\boldsymbol{Q}(\boldsymbol{Z}) - \boldsymbol{X}\right\|^2 + \lambda\,\mathrm{Tr}\left(\boldsymbol{Z}\boldsymbol{K}_X^{-1}\boldsymbol{Z}^T + \boldsymbol{Q}\boldsymbol{K}_Z^{-1}\boldsymbol{Q}^T\right) \\ & \text{s.t.} \ \ \left\|\boldsymbol{z}_i\right\|^2 = 1 \ \forall i \in [n] \end{aligned}$$

Given $\boldsymbol{Z}$*, any new* $\boldsymbol{x}^* \in \mathbb{R}^d$ *is embedded in the bottleneck as*

$$\boldsymbol{z}^* = \boldsymbol{Z}\boldsymbol{K}_X^{-1}k(\boldsymbol{x}^*, \boldsymbol{X})$$

and reconstructed as

$$\hat{\boldsymbol{x}}^* = \boldsymbol{X}\left(\boldsymbol{K}_Z + \lambda\boldsymbol{I}_n\right)^{-1}k(\boldsymbol{z}^*, \boldsymbol{Z})$$

Remark 1 (Connection to Kernel PCA). In light of the known connections between linear autoencoders and standard PCA, it is natural to wonder how above Kernel AE relates to Kernel PCA (Schölkopf, Smola, and Müller 1998). The latter performs PCA in the RKHS $\mathcal{H}$, and is hence equivalent

to minimising the reconstruction error over all orthogonal basis transformations $\boldsymbol{W}$ in $\mathcal{H}$

$$\mathcal{L}(W) = \sum_{i=1}^{n} \left\| \phi(\boldsymbol{x}_i) - \boldsymbol{W}^T \boldsymbol{P}_h \boldsymbol{W} \phi(\boldsymbol{x}_i) \right\|^2 \tag{6}$$

where $\boldsymbol{P}_h$ denotes the projection onto the first h canonical basis vectors, and we assume that the features $\phi(\boldsymbol{x}_i)$ are centered. How does the Kernel AE $\boldsymbol{W}_2^T \phi_2(\boldsymbol{W}_1^T \phi_1(\boldsymbol{x}))$ relate to this if we replace the regularisation terms on $\boldsymbol{W}_1, \boldsymbol{W}_2$ by an orthogonality constraint on both? For simplicity, let us assume $h = 1$. The optimisation problem then essentially becomes

$$\mathcal{L} = \sum_{i=1}^{n} \left\| \boldsymbol{x}_i - \boldsymbol{W}_2(\boldsymbol{W}_1(\boldsymbol{x}_i)) \right\|^2 \tag{7}$$

where $\boldsymbol{W}_1 : \mathbb{R}^d \to \mathbb{R}$ is a function from the RKHS over $\mathbb{R}^d$ (with unit norm), and $\boldsymbol{W}_2 : \mathbb{R} \to \mathbb{R}^d$ consists of d orthonormal functions from the RKHS over $\mathbb{R}$. Clearly, Eq. 7 evaluates the reconstruction error in the sample space, much in contrast to the loss function in Eq. 6 which computes distances in the RKHS. Additionally, the map $\boldsymbol{W}^T$ learned in Eq. 6 from the bottleneck back to $\mathcal{H}$ is given by the basis transformation $\boldsymbol{W}$ in Kernel PCA, whereas it is fixed as the feature map ϕ over $\mathbb{R}^h$ in the AE setting. Kernel PCA can be viewed as an AE architecture that maps solely within $\mathcal{H}$, via

$$\phi(\boldsymbol{x}) \to \boldsymbol{W}\phi(\boldsymbol{x}) \to \boldsymbol{P}_h \boldsymbol{W}\phi(\boldsymbol{x}) \to \boldsymbol{W}^T \boldsymbol{P}_h \boldsymbol{W}\phi(\boldsymbol{x}).$$

Notably, the results of Kernel PCA usually do not translate back to the sample space easily. Given a point $\boldsymbol{x} \in \mathbb{R}^d$, the projection of $\phi(\boldsymbol{x})$ onto the subspace spanned by Kernel PCA is not guaranteed to have a pre-image in $\mathbb{R}^d$, and a direct interpretation of the learned representations can therefore be difficult. In contrast, our method is quite interpretable, as it also provides an explicit formula for the reconstruction $\hat{\boldsymbol{x}}^*$ of unseen data points — not just their projection onto a subspace in an abstract Hilbert space. In particular, by choosing an appropriate kernel[1] and tuning the regularisation parameter λ, a practitioner may directly control the complexity of both decoder as well as the encoder.

Remark 2 (De-noising Kernel AE). In this section, we considered the standard setting where the model learns the reconstruction of the input data. A common extension is the *de-nosing* setting (e.g. (Buades, Coll, and Morel 2005b; Vincent et al. 2010)), which formally moves the model from a reconstruction to a SSL setting, where we replace the input with a noisy version of the data. The goal is now to learn a function that removes the noise and, in the process, learns latent representations. More formally, the mapping becomes

$$\overline{\boldsymbol{x}} \in \mathbb{R}^d \xrightarrow{\phi_1(\cdot)} \boldsymbol{r}_1 \in \mathcal{H}_1 \xrightarrow{\boldsymbol{W}_1} \boldsymbol{z} \in \mathbb{R}^h \xrightarrow{\phi_2(\cdot)} \boldsymbol{r}_2 \in \mathcal{H}_2 \xrightarrow{\boldsymbol{W}_2} \boldsymbol{x} \in \mathbb{R}^d.$$

where $\overline{\boldsymbol{x}}$ is given by $\overline{\boldsymbol{x}} := \boldsymbol{x} + \varepsilon$ with ε being the noise term. A precise formulation is provided in the Appendix. We again note that the simple extension to this setting further distinguishes our approach from Kernel PCA, where such augmentations are not as easily possible.

[1]The choice of kernel could be influenced by the type of functions that are considered interpretable in the domain of application.

Generalisation Error Bounds

Kernel methods in the supervised setting are well established and previous works offer rigorous theoretical analysis (Wahba 1990; Schölkopf and Smola 2002; Bartlett and Mendelson 2002). In this section, we show that the proposed kernel methods for contrastive SSL as well as for the reconstruction setting can be analysed in a similar fashion, and we provide generalisation error bounds for each of the proposed models.

Error Bound for Representation Learning Setting

In general we are interested in characterizing $\mathcal{L}(f) = \mathbb{E}_{X\sim\mathcal{D}}[l(f(\boldsymbol{X}))]$ where $f(\boldsymbol{X})$ is the representation function and $l(\cdot)$ is a loss function, which is either a contrastive loss or based on reconstruction. However, since we do not have access to the distribution of the data $\mathcal{D}$, we can only observe the empirical (training) error, $\widehat{\mathcal{L}}(f) = \frac{1}{n}\sum_{i=1}^{n} l(f(\boldsymbol{X}_i))$, where n is the number of unlabelled datapoints we can characterise the generalisation error as

$$\mathcal{L}(f) \leq \widehat{\mathcal{L}}(f) + \text{complexity term} + \text{slack term}$$

The exact form of the complexity and slack term depends on the embeddings and the loss. In the following, we precisely characterise them for all of the proposed models.

Theorem 6 (Error Bound for Kernel Contrastive Loss). *Let $\mathcal{F} := \left\{ \boldsymbol{X} \mapsto \boldsymbol{W}^T\phi(\boldsymbol{X}) : \|\boldsymbol{W}^T\|_{\mathcal{H}} \leq \omega \right\}$ be the class of embedding functions we consider in the contrastive setting. Define $\alpha := \left(\sqrt{h\,\mathrm{Tr}[\boldsymbol{K}_X]} + \sqrt{h\,\mathrm{Tr}[\boldsymbol{K}_{X^-}]} + \sqrt{h\,\mathrm{Tr}[\boldsymbol{K}_{X^+}]}\right)$ as well as $\kappa := \max\left\{ k(\boldsymbol{x}_i', \boldsymbol{x}_i') : \boldsymbol{x}_i' \in \{\boldsymbol{x}_i, \boldsymbol{x}_i^-, \boldsymbol{x}_i^+\}_{i=1}^n \right\}$. We then obtain the generalisation error for the proposed losses as follows.*

1. ***Simple Contrastive Loss.*** *Let the loss be given by Definition 1. Then, for any $\delta > 0$, the following statement holds with probability at least $1-\delta$ for any $f \in \mathcal{F}$:*

$$\mathcal{L}(f) \leq \widehat{\mathcal{L}}(f) + O\left(\frac{\omega^2\sqrt{\kappa}\alpha}{n} + \omega^2\kappa\sqrt{\frac{\log\frac{1}{\delta}}{n}} \right)$$

2. ***Spectral Contrastive Loss.*** *Let the loss be given by Definition 2. Then, for any $\delta > 0$, the following statement holds with probability at least $1-\delta$ for any $f \in \mathcal{F}$:*

$$\mathcal{L}(f) \leq \widehat{\mathcal{L}}(f) + O\left(\lambda\omega^2 + \frac{\omega^3\kappa^{\frac{3}{2}}\alpha}{n} + \omega^4\kappa^2\sqrt{\frac{\log\frac{1}{\delta}}{n}} \right)$$

Similarly to the contrastive setting, we obtain a generalisation error bound for the Kernel AE as follows.

Theorem 7 (Error Bound for Kernel AE). *Assume the optimisation be given by Definition 3 and define the class of encoders/decoders as*

$$\mathcal{F} := \Big\{ \boldsymbol{X} \mapsto \boldsymbol{W}_2^T\phi_2\left(\boldsymbol{W}_1^T\phi_1(\boldsymbol{X})\right) : \|\boldsymbol{W}_1^T\phi(\boldsymbol{x}_i)\|^2 = 1\ \forall\, i\,,\ \|\boldsymbol{W}_1^T\|_{\mathcal{H}} \leq \omega_1, \|\boldsymbol{W}_2^T\|_{\mathcal{H}} \leq \omega_2 \Big\}$$

Let $r := \lambda(\omega_1^2 + \omega_2^2)$ and $\gamma = \max_{\boldsymbol{s}\in\mathbb{R}^h}\left\{ k(\boldsymbol{s},\boldsymbol{s}) : \|\boldsymbol{s}\|^2 = 1 \right\}$. Then for any $\delta > 0$, the following statement holds with probability at least $1-\delta$ for any $f \in \mathcal{F}$:

$$\mathcal{L}^{AE}(f) \leq \widehat{\mathcal{L}}^{AE}(f) + O\left(r + \frac{\omega_2\sqrt{d\gamma}}{\sqrt{n}} + \sqrt{\frac{\log\frac{1}{\delta}}{n}} \right)$$

The above bounds demonstrate that with increasing number of unlabelled datapoints, the complexity term in the generalisation-error bound decreases. Thus, the proposed models follow the general SSL paradigm of increasing the number of unlabelled data to improve the model performance.

Error Bound for Supervised Downstream Task

While the above bounds provide us with insights on the generalisation of the representation learning setting, in most cases we are also interested in the performance on downstream tasks. Conveniently, we can use the setup presented in (Saunshi et al. 2019) to bound the error of the supervised downstream tasks in terms of the unsupervised loss, providing a bound of the form

$$\mathcal{L}_{sup}(f) \leq c_1 \widehat{\mathcal{L}}_{un}(f) + c_1 * \text{complexity term}$$

where c_1 and c_2 are data dependent constants. We present the formal version of this statement in the supplementary material for all presented models.

This highlights that a better representation (as given by a smaller loss of the unsupervised task) also improves the performance of the supervised downstream task.

Experiments

In this section we illustrate the empirical performance of the kernel-based representation learning models introduced in this paper. As discussed in the introduction, there is a wide range of representation learning models, that are often quite specific to the given task. We mainly consider classification in a setting with only partially labelled data at our disposal, as well as image de-noising using the Kernel AE. We state the main setup and results in the following[2].

Classification on Embedding

Data. In this section, we consider the following four datasets: *concentric circles, cubes* (Pedregosa et al. 2011), *Iris* (Fisher 1936) and *Ionosphere* (Sigillito. et al. 1989). We fix the following data split: *unlabelled* = 50%, *labelled* = 5% and *test* = 45%, and consider $h = 2$ as the embedding dimension.

Classification task using k nearest neighbours (k-nn) using embedding as features. We investigate classification as an example of a supervised downstream task. The setting is the following: We have access to $\boldsymbol{X}_{unlab.}$ and $\boldsymbol{X}_{lab.}$ datapoints, which we use to train the representation learning model without access to labels. Then, as the downstream classification model, we consider a k-nn model (with $k = 3$) learned on the embedding of $\boldsymbol{X}_{lab.}$, with corresponding labels $\boldsymbol{Y}_{lab.}$. We test on $\boldsymbol{X}_{test}, \boldsymbol{Y}_{test}$. As a benchmark, we compare to k-nn both on the original features as well as on the embeddings obtained by standard Kernel PCA.

Choice of kernel and their parameterization. For the proposed kernel methods as well as for Kernel PCA we consider three standard kernels, *Gaussian, Laplacian and linear kernels* as well as a 1-layer ReLU Kernel (Bietti and Bach 2021). For Gaussian and Laplacian kernel we choose the bandwidth using a grid search over 15 steps spaced logarithmically between 0.01 and 100. We perform leave-one-out validation on $\boldsymbol{X}_{lab.}$ to pick the bandwidth of the method applied to the test set. The classification experiments on the above listed datasets are present in Figure 1. All results show the mean and standard deviation over five splits of each dataset. It is apparent throughout the experiments that the choice of kernel plays a significant role in the overall performance of the model. This dependency is not surprising, as the performance of a specific kernel directly links to the underlying data-structure, and the choice of kernel is an essential part of the model design. This is in accordance with existing kernel methods — and an important future direction is to analyze what kernel characteristics are beneficial in a representation learning setting.

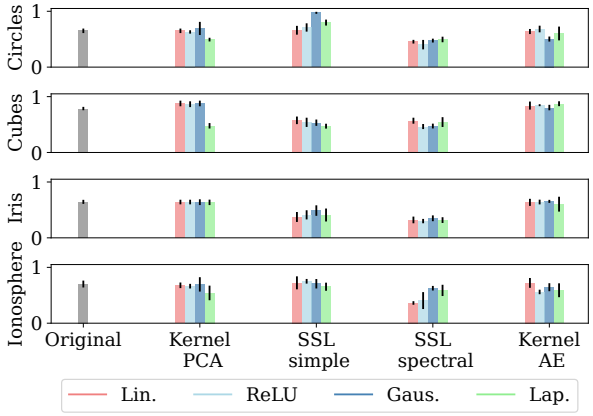

Figure 1: From left to right: we first consider k-nn on the original features followed by k-nn on embeddings obtained by Kernel PCA, and the proposed methods.

Comparison of supervised and representation learning. As stated in the introduction (and supported theoretically in the previous section), the main motivation for representation learning is to take advantage of unlabelled data by learning embeddings that outperform the original features on downstream tasks. To evaluate this empirically for the kernel representation learning models analyzed in this paper, we compare k-nn on the original data to k-nn on the embeddings as shown in Figure 1. We observe that for *Circles, Cubes, Iris and Ionosphere* there always exists an embedding that outperforms k-nn on the original data.

Comparing different embedding methods. Having observed that learning a representation before classification is beneficial, we now focus on the different embedding approaches. While the performed experiments do not reveal clear trends between different methods, we do note that the proposed methods overall perform on par or outperform Kernel PCA, underlining their relevance for kernel SSL.

[2]We provide all further details (as well as experiments on additional datasets) in the arxiv version.
We provide a Python implementation on https://github.com/pascalesser/Representation-Learning-with-Kernels.

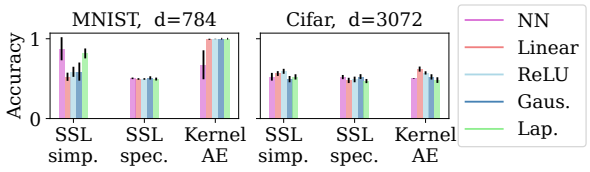

Figure 2: Comparison of kernel methods and neural network models for classification.

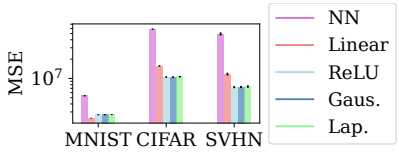

Figure 3: De-noising using NN AE with and Kernel AE.

Comparison to Neural Networks for Classification and De-noising

Representation learning has mainly been established in the context of deep neural networks. In this paper, we make a step towards decoupling the representation learning paradigm from the widely used deep learning models. Nonetheless, we can still compare the proposed kernel methods to neural networks. We construct the *corresponding NN model* by replacing the linear function in the reproducing kernel Hilbert space, $\boldsymbol{W}^\top \phi(\boldsymbol{x})$ by an one-hidden layer neural network $\boldsymbol{W}_2\sigma(\boldsymbol{W}_1\boldsymbol{x})$, where $\sigma(\cdot)$ is a non-linear activation function (and we still minimise a similar loss function).

Classification. We compare the performance of both representation learning approaches in Figure 2 for datasets *CIFAR-10* (Krizhevsky, Hinton et al. 2009), as well as a subset of the first two classes of *MNIST* (Deng 2012) (i.e. $n = 500$). We observe that the kernel methods perform on par with, or even outperform the neural networks. This indicates that there is not one dominant approach but one has to choose depending on the given task.

De-Noising. As a second task, we consider de-noising using (Kernel) AE. Data is sampled from the first five classes of *MNIST*, *CIFAR-10* and *SVHN* (Netzer et al. 2011) with $n = 225$ and the noisy version are generated by $\overline{\boldsymbol{x}} := \boldsymbol{x} + \varepsilon$, $\varepsilon_i \sim \mathcal{N}(0, 0.1)$. We compare the performance of kernel-based approaches with the neural network reconstructions in Figure 3 by plotting the mean square error on the test set between the AE output and the clean data. Kernel AE outperforms the neural network AE in all considered settings. Moreover, there is little variation among the different kernels. This indicates that at least in the presented settings, the proposed kernel methods pose a viable alternative to traditional neural network based representation learning.

Formal connection between Kernel and neural network model. While it is known that regression with infinite-width networks is equivalent to kernel regression with neural tangent kernel (NTK) (Jacot, Gabriel, and Hongler 2018; Arora et al. 2019), similar results are not known for SSL and this brings up the question: Is kernel SSL equivalent to SSL with infinitely-wide neural networks? It is possible to show that single-layer Kernel AE with NTK is the infinite-width limit of over-parameterized AE (Nguyen, Wong, and Hegde 2021; Radhakrishnan, Belkin, and Uhler 2020). We believe that the same equivalence also holds for kernel contrastive learning (Definition 1) with NTK, but leave this as an open problem. We do not know if Definition 3 with NTK is the limit for bottleneck deep learning AE since, as we note earlier, there is no unique formulation for Kernel AE.

Discussions and Outlook

In this paper, we show that new variants of representer theorem allows one to rephrase SSL optimisation problems or the learned representations in terms of kernel functions. The resulting kernel SSL models provide natural tools for theoretical analysis. *We believe that presented theory and method provide both scope for precise analysis of SSL and can also be extended to other SSL principles, such as other pretext tasks or joint embedding methods* (Saunshi et al. 2019; Bardes, Ponce, and LeCun 2022; Grill et al. 2020; Chen and He 2020). We conclude with some additional discussions.

Computational limitations and small dataset setting. Exactly computing kernel matrices is not scalable, however random feature (RF) approximations of kernel methods are well suited for large data (Rahimi and Recht 2007; Carratino, Rudi, and Rosasco 2018). While one may construct scalable kernel representation learning methods using RF, it should be noted that RF models are lazy-trained networks (Ghorbani et al. 2019). So fully-trained deep representation learning models may be more suitable in such scenarios. However representation learning is relevant in all problems with availability of partially labelled data. This does not only apply to the big data regime where deep learning approaches are predominantly used, but also to *small data settings* where kernel methods are traditionally an important tool (Fernández-Delgado et al. 2014). *The practical significance of developing kernel approaches is to broaden the scope of the representation learning paradigm beyond the deep learning community.*

Kernel SSL vs. non-parametric data embedding. Several non-parametric generalisations of PCA, including functional PCA, kernel PCA, principle curves etc., have been studied over decades and could be compared to Kernel AEs. However, unlike kernel SSL, embedding methods are typically not inductive. As shown previously, the inductive representation learning by Kernel AE and contrastive learning make them suitable for downstream supervised tasks.

Kernel SSL vs. SSL with infinite-width neural networks. While it is known that regression with infinite-width networks is equivalent to kernel regression with neural tangent kernel (NTK) (Jacot, Gabriel, and Hongler 2018), similar results are not known for SSL. We believe that a study of the learning dynamics of neural network based SSL would show their equivalence with our kernel contrastive models with NTK. However, it is unclear to us whether a similar result can exist for kernel AE, as NTK approximations typically do not hold in the presence of bottleneck layers (Liu, Zhu, and Belkin 2020).

Acknowledgments

This work has been supported by the German Research Foundation (Priority Program SPP 2298, project GH 257/2-1, and Research Grant GH 257/4-1) and the DAAD programme Konrad Zuse Schools of Excellence in Artificial Intelligence, sponsored by the Federal Ministry of Education and Research.

References

Aronszajn, N. 1950. Theory of reproducing kernels. *Transactions of the American mathematical society*.

Arora, S.; Du, S. S.; Hu, W.; Li, Z.; Salakhutdinov, R.; and Wang, R. 2019. On Exact Computation with an Infinitely Wide Neural Net. In *Advances in neural information processing systems*.

Bardes, A.; Ponce, J.; and LeCun, Y. 2022. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. In *International Conference on Learning Representations*.

Bartlett, P. L.; and Mendelson, S. 2002. Rademacher and Gaussian Complexities: Risk Bounds and Structural Results. *Journal of Machine Learning Research*.

Bay, H.; Tuytelaars, T.; and Van Gool, L. 2006. SURF: Speeded Up Robust Features. In *Computer Vision – ECCV 2006*. Springer Berlin Heidelberg.

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*.

Bietti, A.; and Bach, F. 2021. Deep Equals Shallow for ReLU Networks in Kernel Regimes. In *International Conference on Learning Representations*.

Bromley, J.; Guyon, I.; LeCun, Y.; Säckinger, E.; and Shah, R. 1993. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*.

Buades, A.; Coll, B.; and Morel, J.-M. 2005a. A review of image denoising algorithms, with a new one. *Multiscale modeling & simulation*.

Buades, A.; Coll, B.; and Morel, J.-M. 2005b. A review of image denoising algorithms, with a new one. *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*.

Carratino, L.; Rudi, A.; and Rosasco, L. 2018. Learning with SGD and Random Features. In *Advances in Neural Information Processing Systems 31*.

Chahal, H.; and Toner, H. 2021. 'Small Data' Are Also Crucial for Machine Learning. *Scientific American*.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*.

Chen, X.; and He, K. 2020. Exploring Simple Siamese Representation Learning. *arXiv preprint arXiv:2011.10566*.

Cortes, C.; and Vapnik, V. 1995. Support-vector networks. *Machine learning*.

Deng, L. 2012. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,*.

Fernández-Delgado, M.; Cernadas, E.; Barro, S.; and Amorim, D. 2014. Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*.

Fisher, R. A. 1936. The use of multiple measurements in taxonomic problems. *Annals of eugenics*.

Ghorbani, B.; Mei, S.; Misiakiewicz, T.; and Montanari, A. 2019. Limitations of Lazy Training of Two-layers Neural Network. In *Advances in Neural Information Processing Systems 32*.

Grill, J.-B.; Strub, F.; Altché, F.; Tallec, C.; Richemond, P. H.; Buchatskaya, E.; Doersch, C.; Pires, B. A.; Guo, Z. D.; Azar, M. G.; Piot, B.; Kavukcuoglu, K.; Munos, R.; and Valko, M. 2020. Bootstrap Your Own Latent a New Approach to Self-Supervised Learning. In *Advances in neural information processing systems*.

Gu, S.; Kelly, B.; and Xiu, D. 2020. Empirical asset pricing via machine learning. *The Review of Financial Studies*.

Hainmueller, J.; and Hazlett, C. 2014. Kernel regularized least squares: Reducing misspecification bias with a flexible and interpretable machine learning approach. *Political Analysis*.

HaoChen, J. Z.; Wei, C.; Gaidon, A.; and Ma, T. 2021. Provable Guarantees for Self-Supervised Deep Learning with Spectral Contrastive Loss. In *Advances in neural information processing systems*.

Jacot, A.; Gabriel, F.; and Hongler, C. 2018. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. In *Advances in neural information processing systems*.

Jing, L.; and Tian, Y. 2019. Self-Supervised Visual Feature Learning With Deep Neural Networks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Johnson, D. D.; Hanchi, A. E.; and Maddison, C. J. 2022. Contrastive Learning Can Find An Optimal Basis For Approximately View-Invariant Functions. *arXiv preprint arXiv:2210.01883*.

Kanazawa, A.; Jacobs, D. W.; and Chandraker, M. 2016. Warpnet: Weakly supervised matching for single-view reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Kiani, B. T.; Balestriero, R.; Chen, Y.; Lloyd, S.; and LeCun, Y. 2022. Joint embedding self-supervised learning in the kernel regime. *arXiv preprint arXiv:2209.14884*.

Kimeldorf, G.; and Wahba, G. 1971. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*.

Kramer, M. A. 1991. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Kunin, D.; Bloom, J.; Goeva, A.; and Seed, C. 2019. Loss Landscapes of Regularized Linear Autoencoders. In *International Conference on Machine Learning*.

Liu, C.; Zhu, L.; and Belkin, M. 2020. On the linearity of large non-linear models: when and why the tangent kernel is constant. In *Advances in Neural Information Processing Systems 33*.

Lowe, D. G. 1999. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*.

Mika, S.; Ratsch, G.; Weston, J.; Scholkopf, B.; and Mullers, K.-R. 1999. Fisher discriminant analysis with kernels. In *Neural networks for signal processing IX: Proceedings of the 1999 IEEE signal processing society workshop*.

Misra, I.; and Maaten, L. v. d. 2020. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*.

Nguyen, T. V.; Wong, R. K. W.; and Hegde, C. 2021. Benefits of Jointly Training Autoencoders: An Improved Neural Tangent Kernel Analysis. *IEEE Transactions on Information Theory*.

Pearson, K. 1901. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*.

Ponte, P.; and Melko, R. G. 2017. Kernel methods for interpretable machine learning of order parameters. *Physical Review B*.

Pretorius, A.; Kroon, S.; and Kamper, H. 2018. Learning Dynamics of Linear Denoising Autoencoders. In *International Conference on Machine Learning*.

Radhakrishnan, A.; Belkin, M.; and Uhler, C. 2020. Overparameterized neural networks implement associative memory. *Proceedings of the National Academy of Sciences of the United States of America*.

Rahimi, A.; and Recht, B. 2007. Random Features for Large-Scale Kernel Machines. In *Advances in Neural Information Processing Systems 20*.

Refinetti, M.; and Goldt, S. 2022. The dynamics of representation learning in shallow, non-linear autoencoders. In *International Conference on Machine Learning*.

Roßbach, P. 2018. Neural networks vs. random forests–does it always have to be deep learning.

Saunshi, N.; Plevrakis, O.; Arora, S.; Khodak, M.; and Khandeparkar, H. 2019. A Theoretical Analysis of Contrastive Unsupervised Representation Learning. In *Proceedings of the 36th International Conference on Machine Learning*.

Schölkopf, B.; Herbrich, R.; and Smola, A. J. 2001. A Generalized Representer Theorem. In *Annual Conference on Computational Learning Theory*.

Schölkopf, B.; and Smola, A. J. 2002. *Learning with Kernels: support vector machines, regularization, optimization, and beyond*. Adaptive computation and machine learning series. MIT Press.

Schölkopf, B.; Smola, A.; and Müller, K.-R. 1998. Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Computation*.

Shah, A.; Sra, S.; Chellappa, R.; and Cherian, A. 2022. Max-Margin Contrastive Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Shwartz-Ziv, R.; and Armon, A. 2022. Tabular data: Deep learning is not all you need. *Inf. Fusion*.

Sigillito., V.; S., W.; L., H.; ; and K., B. 1989. Ionosphere. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5W01B.

Todman, L. C.; Bush, A.; and Hood, A. S. 2023. 'Small Data'for big insights in ecology. *Trends in Ecology & Evolution*.

Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; Manzagol, P.-A.; and Bottou, L. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*.

Wahba, G. 1990. *Spline Models for Observational Data*. SIAM.

Xu, P.; Ji, X.; Li, M.; and Lu, W. 2023. Small data machine learning in materials science. *npj Computational Materials*.

Yang, B.; Fu, X.; Sidiropoulos, N. D.; and Hong, M. 2017. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *international conference on machine learning*, 3861–3870. PMLR.

Zhai, R.; Liu, B.; Risteski, A.; Kolter, Z.; and Ravikumar, P. 2023. Understanding Augmentation-based Self-Supervised Representation Learning via RKHS Approximation. *arXiv preprint arXiv:2306.00788*.

Zhang, C.; Zhang, C.; Song, J.; Yi, J. S. K.; Zhang, K.; and Kweon, I. S. 2022. A survey on masked autoencoder for self-supervised learning in vision and beyond. *arXiv preprint arXiv:2208.00173*.