Layer-wise Learning via Kernel Embedding

*Aria Masoomi[1], *Chieh Wu[1], Arthur Gretton[2], and Jennifer Dy[1]

[1]*Department of Electrical and Computer Engineering, Northeastern University*
[2]*Gatsby Computational Neuroscience Unit, University College London*

Abstract

Due to a recent debate over the biological plausibility of backpropagation (BP), finding an alternative network optimization strategy has become a topic of interest. We design a kernel network that is solved greedily to theoretically answer several questions. First, if BP is difficult to simulate in the brain, are there *trivial network weights* (requiring minimum computation) that allow a greedily trained network to exhibit Universal Approximation? Second, can a greedily trained network converge to a kernel? What kernel will it converge to? We proved that the Kernel Embedding is the trivial solution that compels the greedy procedure to converge to a kernel with Universal property. This workshop paper is part two of the full paper. For the full paper, see `https://arxiv.org/abs/2006.08539`.

1 Introduction

Due to the brain-inspired architecture of Multi-layered Perceptrons (MLP), a more direct relation between MLPs and our brains has been a research area of interest [1, 2]. This attempt has triggered a controversy [3] around the neural plausibility of backpropagation (BP), i.e., while some contend that brains cannot simulate BP [4, 5], others have recently proposed counterclaims with a new generation of models [6, 7, 8, 9, 10, 11, 12]. This debate has inspired the search for alternative optimization strategies. Among them, a layer-wise strategy has been proposed as a more likely candidate to match existing understandings in Neuroscience [13]. Inspired by a layer-wise training strategy, we design a new type of kernel network (*KChain*) that is solved greedily. We then leverage this setup to answer several theoretical questions.

1. Is there a *trivial solution* that enables Universal Approximation?
2. Can a layer-wise construct converge to a fixed kernel? What kernel exactly is that?

The first question seeks to characterize the expressiveness of *KChain* function class. While much work has already been dedicated to MLP's expressiveness [14, 15, 16]. Indeed, the *Universal Approximation Theorems* [17, 18, 19] have already been discovered. However, while these research assumed traditional activation functions of $\psi : \mathbb{R} \to \mathbb{R}$, we generalize the image of the activation function to the Reproducing Kernel Hilbert Space (RKHS) where the activation function can also be $\psi : \mathbb{R} \to \mathbb{R}^m$. Given some unknown trivial weights, it is not immediately obvious that a recurrent usage of these weights can also render a network with Universality. Our work proves that besides Stochastic Gradient Descent (SGD) with BP, Universality is also achievable greedily using the kernel embedding ($\mathcal{U}$) as the trivial solution. Since $\mathcal{U}$ without normalization only requires the summation operation, the triviality of the weights is particularly interesting for Neuroscience.

The 2nd question answers how modern MLPs relates to composition of kernels? The importance of linking kernel methods to neural network has been emphasized by Belkin and Niyogi [20], and in this respect, work by [21, 22, 13] have analyzed the various composition of kernels. MLPs have also been

*Signifies equal contribution.

34th NeurIPS Workshop (BEYOND BACKPROPAGATION) 2020, Vancouver, Canada.

modeled as a Gaussian process (GP) [23, 24, 25]. This kernel interpretation has inspired a significant amount of research [26, 27, 28, 29, 30, 31]. While *KChain* also views the network as a composition of kernels, our model differ in 2 key respects. First, we replace the conventional activation function with the feature map of a Gaussian kernel (GK), resulting in an infinitely wide network that is solved via the kernel trick. Second, instead of using a composition of *fixed* feature maps, we parameterized the GK via a set of weights W, resulting in a model that is always choosing the optimal RKHS for each layer. We prove that this seemingly minor adjustment is key to Universality.

2 Network Formulation

Our network structure is as a Multi-layered Perceptron (MLP) where the data $X \in \mathbb{R}^{n\times d}$ is passed through a sequence of layers. Each layer consists of a linear projection $W_l \in \mathbb{R}^{m\times q}$ followed by an activation function $\psi : \mathbb{R}^{n\times q} \to \mathbb{R}^{n\times m}$. The input and output at the l^{th} layer are $R_{l-1} \in \mathbb{R}^{n\times m}$ and $R_l \in \mathbb{R}^{n\times m}$, implying each layer as $R_l = \psi(R_{l-1}W_l)$. For each layer, the i^{th} row of its input R_{l-1} is $r_i \in \mathbb{R}^m$ and it represents the i^{th} input sample. We denote $\mathcal{W}_l$ as a function where $\mathcal{W}_l(R_{l-1}) = R_{l-1}W_l$; consequently, each layer is also a function $\phi_l = \psi \circ \mathcal{W}_l$. By stacking L layers together, the entire network itself becomes a function ϕ where $\phi = \phi_L \circ ... \circ \phi_1$. The output of the network is then used to minimize an empirical risk $(\mathcal{H})$.

We propose to find W_l at each layer greedily; this is equivalent to solving a sequence of *single-layered* networks where the previous network output becomes the current layer's input. At each layer, instead of maximizing a traditional classification objective, we find the W_l that maximizes the dependency between the layer output and the label via the Hilbert Schmidt Independence Criterion (HSIC) [32]:

$$\max_{W_l} \ \mathrm{Tr}\left(\Gamma \ \left[\psi(R_{l-1}W_l)\psi^T(R_{l-1}W_l)\right]\right) \quad \text{where} \quad \psi(R_{l-1}W_l) = \phi_l(R_{l-1}). \tag{1}$$

We refer to this network as a Kernel Chain (*KChain*), constructed by composing a chain of kernels. It departs from traditional MLP by generalizing the concept of the activation function to a kernel feature map, i.e., while traditional activation functions are $\psi : \mathbb{R} \to \mathbb{R}$, we now have $\psi : \mathbb{R}^q \to \mathbb{R}^m$ where m can potentially be ∞. Specifically, we use the Gaussian Kernel (GK) feature map as ψ, simulating an infinitely wide network. Conveniently, HSIC leverages the kernel trick to spare us the direct computation of $\psi(R_{l-1}W_l)\psi^T(R_{l-1}W_l)$; we instead compute

$$\mathcal{K}(W_l^T r_i, W_l^T r_j) = \exp\{-\frac{||W_l^T r_i - W_l^T r_j||^2}{2\sigma^2}\}. \tag{2}$$

The objective of this work is to identify a trivial solution W_l at each layer such that

1. the network converges to achieve the global optimum of Eq. (1).
2. ϕ converges to a featue map of a fixed kernel where $\mathcal{K}^*(x_i, x_j) = \langle\phi(x_i), \phi(x_j)\rangle$.

3 Theoretical Background

Let the composition of the first l layers be $\phi_{l^\circ} = \phi_l \circ ... \circ \phi_1$ where $l \leq L$. This notation enables us to connect the data directly to the layer output where $R_l = \phi_{l^\circ}(X)$. Since *KChain* is greedy, it solves MLPs by learning the weights of a growing network denoted as a sequence of functions $\{\phi_{l^\circ}\}_{l=1}^L$; we refer to this as the *Kernel Sequence*. As the network grows, we also solve a sequence of different empirical risks $\{\mathcal{H}_l\}_{l=1}^L$; we refer to this as $\mathcal{H}$*-Sequence*.

The $\mathcal{H}$*-Sequence* is the central theoretical theme of *KChain*. While MLP is traditionally solved with all the layers jointly via BP, *KChain* focus on greedily discovering a *Kernel Sequence* that compels the $\mathcal{H}$*-Sequence* to exhibit key behaviors that enable classification. Specifically, we seek an $\mathcal{H}$*-Sequence* that finitely converges to the global optimum of Eq. (1), or $\mathcal{H}^* = \sum_{i,j\in\mathcal{S}} \Gamma_{i,j}$. To accomplish this, we discovered that the kernel embedding of each class is the trivial solution W_l for each layer. That is, if we let r_ι^j be the ι^{th} input sample in class j for layer l, then the trivial solution W_s is

$$W_s = \frac{1}{\sqrt{\zeta}}\left[\sum_\iota r_\iota^{(1)} \quad \sum_\iota r_\iota^{(2)} \quad ... \quad \sum_\iota r_\iota^{(\tau)}\right] = \left[\mathcal{U}^{(1)} \quad \mathcal{U}^{(2)} \quad ... \quad \mathcal{U}^{(\tau)}\right] \tag{3}$$

with $\mathcal{U}^{(c)}$ as the Kernel Embedding of class c and ζ is an unnecessary normalizing constant. Note that since $r_\iota^j(l) = \phi_l(r_\iota^j(l-1))$, the averaging of all the samples in a class c in RKHS results in $\mathcal{U}^c$ [33]. Given this definition, we prove in App. A the following theorem.

Theorem 1. *For any $\mathcal{H}_0$, there exists a set of bandwidths σ_l and a Kernel Sequence $\{\phi_{l^\circ}\}_{l=1}^{L}$ parameterized by W_l in Eq. (3) such that for any $L > 1$ and $\delta > 0$ we can achieve*

$$\mathcal{H}^* - \mathcal{H}_L \leq \delta, \tag{4}$$

The theorem identifies the trivial solution W_s and provides an alternative strategy to optimize a network. Instead of finding the gradient, or needing to propagate information backwards, a layer-wise strategy can also achieve $\mathcal{H}^*$ by simply choosing one value at each layer, the σ_l. Remember that $\mathcal{U}^c$ embeds the entire distributional information of class c into a single point in RKHS. Therefore, the multiplication of W_s constitutes a projection of each sample onto its own class distribution. After this projection, σ_l controls the acceptable pairwise distance between 2 samples of the same class.

In concrete terms, given a classification of dogs/cats, we use a large set of features to identify the *average* dog and cat. The σ_l value then controls the Gaussian distribution around this average concept of a dog/cat. At the next layer, we again pick another set of features to create another average dog/cat. By continuously choosing a range of acceptance σ_l at each layer, perfect dependence can be achieved. When a previously unseen dog is shown, classification is possible simply by comparing its features against the average dog/cat of the last layer with the nearest neighbor as the winner.

Relating $\mathcal{H}^*$ to Classification. As $\mathcal{H}_l \to \mathcal{H}^*$, the maximization of Eq. (1) demonstrates how we learn the kernel. However, we have yet formalize why learning the kernel also induces an optimal classification. To understand this deeper, let us first clarify some key notations and concepts. We refer to the image of ψ and ϕ as the Reproducing Kernel Hilbert Space (RKHS) and the image of $\mathcal{W}$ as the Images of the RKHS Dual Space (IDS). This distinction is crucial because each space dictates the geometric orientations that lead to classification. Keeping this in mind, each layer's output can be measured by the within S_w^l and between S_b^l class scatter matrices defined as

$$S_w^l = \sum_{i,j \in \mathcal{S}} W_l^T (r_i - r_j)(r_i - r_j)^T W_l \quad \text{and} \quad S_b^l = \sum_{i,j \in \mathcal{S}^c} W_l^T (r_i - r_j)(r_i - r_j)^T W_l. \tag{5}$$

These matrices are historically important [34, 35] because their trace ratio $\mathcal{T} = \operatorname{Tr}(S_w)/\operatorname{Tr}(S_b)$ measures class separability, i.e., a small $\mathcal{T}$ implies a small variance around its $\mathcal{U}^c$ under Euclidean distance. Since distance here is measure via Maximum Mean Discrepancy (MMD), a test sample is consequently classified by identifying the closest class kernel embedding, i.e., picking the distribution that best match the test sample. Therefore, a small $\mathcal{T}$ is a crucial criteria for a classifier. In this respect, HSIC is related because by maximizing $\mathcal{H}$, $\mathcal{T}$ is minimized as a byproduct. In fact, we prove that as $\mathcal{H}_l \to \mathcal{H}^*$, $\mathcal{T}$ approaches 0, and ϕ will map samples of different classes into distinct points.

Note that since $\mathcal{T}$ is computed with samples from the image of $\mathcal{W}$, this particular relationship resides in IDS. Concurrently, since the inner product defines similarity in RKHS, samples are simultaneously being partitioned via the angular distance. Indeed, our proof indicates that as $\mathcal{H}_l \to \mathcal{H}^*$, RKHS samples within $\mathcal{S}$ achieves perfect alignment while samples in $\mathcal{S}^c$ become orthogonal to each other, separated by a maximum angle of $\pi/2$. This dual relationship between IDS and RKHS produces a vastly different network output right before and immediately after the final activation function, where different notions of distance (Euclidean and angular) are employed to partition samples. We formally state these results in the following theorem with the proof in App. B.

Theorem 2. *As $l \to \infty$ and $\mathcal{H}_l \to \mathcal{H}^*$, the following properties are satisfied:*

I. the scatter trace ratio $\mathcal{T}$ approaches 0 where

$$\lim_{l \to \infty} \frac{\operatorname{Tr}(S_w^l)}{\operatorname{Tr}(S_b^l)} = 0 \tag{6}$$

II. the Kernel Sequence converges to the following kernel:

$$\lim_{l \to \infty} \mathcal{K}(x_i, x_j)^l = \mathcal{K}^*(x_i, x_j)^l = \begin{cases} 0 & \forall i, j \in \mathcal{S}^c \\ 1 & \forall i, j \in \mathcal{S} \end{cases}. \tag{7}$$

As corollaries to Theorem 2, the resulting partition of samples under Euclidean and angular distance implicitly satisfies different classification objectives in each space. In IDS, *KChain* will map a dataset of τ classes into τ distinct points. While these τ points may not match the original label, this difference is inconsequential. In contrast, samples in RKHS at convergence will reside along

τ orthogonal axes on a unit sphere. By realigning these results to the standard bases, solutions that simulate the softmax are generated to solve CE. Therefore, as $\mathcal{H}_l \to \mathcal{H}^*$, the maximization of Eq. (1) simultaneously minimizes MSE and CE in different spaces without matching the actual labels itself: instead, we match the underlying geometry of the network output. We state the two corollaries below with their proof in App. C.

Corollary 1. *Given $\mathcal{H}_l \to \mathcal{H}^*$, the network output in IDS solves MSE via a translation of labels.*

Corollary 2. *Given $\mathcal{H}_l \to \mathcal{H}^*$, the network output in RKHS solves CE via a change of bases.*

4 Experiments

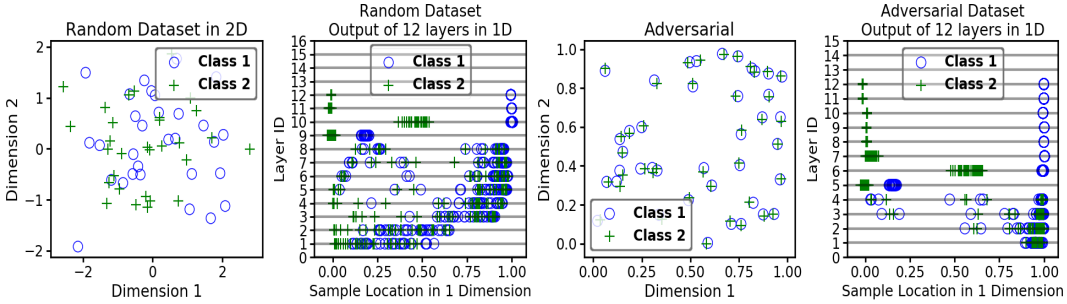

Figure 1: Simulation of Thm. 1 on Random and Adversarial datasets. The 2D representation is shown, and next to it, the 1D output of each layer is displayed over each line. Both datasets achieved the global optimum $\mathcal{H}^*$ at the 12_{th} layers. Refer to App. F for additional results.

Since Thm. 1 guarantees an optimal convergence for *any* dataset with a suboptimal W_s, we designed an Adversarial dataset to trick the network, i.e., the samples pairs in $\mathcal{S}^c$ are significantly closer than samples pairs in $\mathcal{S}$. We next designed a Random dataset with completely random labels. We then simulated Thm. 1 in Python and plot out the sample behavior in Fig. 1. The original 2-dimensional data is shown next to its 1-dimensional IDS results: each line represents the 1-dimensional output at that layer. As predicted by the theorem, our network converged at the 12th layer and perfectly separated the samples based on labels.

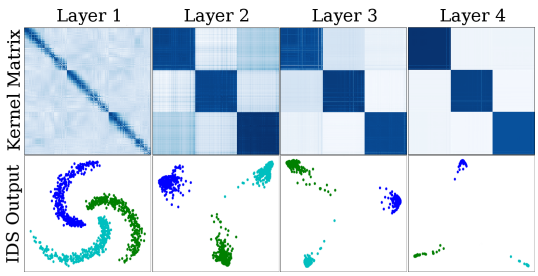

Figure 2: A visual confirmation of Thm. 2. The kernel matrices per layer produced by the *Kernel Sequence* are displayed in the top row with their corresponding outputs in IDS in the bottom row.

Data	Layer Num	σ	Train Acc	Test Acc	Time
divorce	1.90 ± 0.30	3.52 ± 1.35	0.99 ± 0.00	0.93 ± 0.16	1.08 ± 0.15
wine	3.50 ± 0.50	1.43 ± 0.03	0.98 ± 0.01	0.93 ± 0.05	3.40 ± 0.21
face	3.00 ± 1.1	2.92 ± 5.22	0.98 ± 0.01	0.77 ± 0.26	15.10 ± 2.59
car	3.70 ± 0.46	2.09 ± 0.04	1.00 ± 0.00	1.00 ± 0.00	25.83 ± 1.22
cancer	1.20 ± 0.40	2.42 ± 0.05	0.98 ± 0.01	0.97 ± 0.03	7.09 ± 0.249

Figure 3: Running 10-fold Cross Validation with *KChain* on 5 UCI datasets; the average and standard deviation are reported. *KChain* achieves relatively high Test Accuracy except for the high dimensional face data, suggesting that *KChain* may not suitable for images.

To examine Thm. 2, we output the *Kernel Sequence* layer-wised for a synthetic spiral dataset in Fig. 2. We rearrange the samples of the same class to be adjacent to each other. This allows us to quickly evaluate the kernel quality via its block diagonal structure. Since GK is restricted to values between 0 and 1, we let white and dark blue be 0 and 1 respectively where the gradients reflect values in between. Our proof predicts that the *Kernel Sequence* should converge to the optimal kernel $\mathcal{K}^*$, i.e., the *Kernel Sequence* should evolve from an uninformative kernel into a highly discriminating kernel of perfect block diagonal structures. Corresponding to the top row, the bottom row plots out the samples in IDS at each layer. As predicted by Thm. 2, the samples of the same class incrementally converge towards a single point in IDS. Corroborating with corollary 1 and 2, if we replace the HSIC objective from a trained *KChain* with MSE or CE, the resulting loss both achieves approximately 0.

To see if the W_s promotes generalization, in Fig. 3, we apply 10-Fold Crossvalidation with *KChain* on 5 popular UCI datasets (Dataset details in App. D . The average number of layers indicates how quickly the *Kernel Sequence* converges. The average σ used by the Gaussian kernel is shown in the σ column; the relatively large σ values indicates smoothness of ϕ. Notably, the Test Accuracy are reasonably high for most datasets but performed poorly against the image data *face*.

References

[1] Anthony M Zador. A critique of pure learning and what artificial neural networks can learn from animal brains. *Nature communications*, 10(1):1–7, 2019.

[2] Edgar Y Walker. Deep neural networks uncover what the brain likes to see. *Nature Neuroscience*, 10(1):1–7, 2019.

[3] James CR Whittington and Rafal Bogacz. Theories of error back-propagation in the brain. *Trends in cognitive sciences*, 2019.

[4] Francis Crick. The recent excitement about neural networks. *Nature*, 337(6203):129–132, 1989.

[5] Stephen Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive science*, 11(1):23–63, 1987.

[6] Geoffrey Hinton. How to do backpropagation in a brain. In *Invited talk at the NIPS'2007 deep learning workshop*, volume 656, 2007.

[7] T.P. Lillicrap, A. Santoro, L. Marris, C. Akerman, and G. Hinton. Backpropagation and the brain. In *Nat Rev Neurosci (2020)*, volume 656, 2007.

[8] Qianli Liao, Joel Z Leibo, and Tomaso Poggio. How important is weight symmetry in backpropagation? In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[9] Yoshua Bengio, Thomas Mesnard, Asja Fischer, Saizheng Zhang, and Yuhuai Wu. Stdp-compatible approximation of backpropagation in an energy-based model. *Neural computation*, 29(3):555–577, 2017.

[10] Jordan Guerguiev, Timothy P Lillicrap, and Blake A Richards. Towards deep learning with segregated dendrites. *Elife*, 6:e22901, 2017.

[11] João Sacramento, Rui Ponte Costa, Yoshua Bengio, and Walter Senn. Dendritic cortical microcircuits approximate the backpropagation algorithm. In *Advances in Neural Information Processing Systems*, pages 8721–8732, 2018.

[12] James CR Whittington and Rafal Bogacz. An approximation of the error backpropagation algorithm in a predictive coding network with local hebbian synaptic plasticity. *Neural computation*, 29(5):1229–1262, 2017.

[13] Wan-Duo Kurt Ma, JP Lewis, and W Bastiaan Kleijn. The hsic bottleneck: Deep learning without back-propagation. *arXiv preprint arXiv:1908.01580*, 2019.

[14] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks: A view from the width. In *Advances in neural information processing systems*, pages 6231–6239, 2017.

[15] Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pages 2924–2932, 2014.

[16] Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *Advances in neural information processing systems*, pages 3360–3368, 2016.

[17] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[18] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

[19] Ding-Xuan Zhou. Universality of deep convolutional neural networks. *Applied and computational harmonic analysis*, 48(2):787–794, 2020.

[20] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[21] Julien Mairal, Piotr Koniusz, Zaid Harchaoui, and Cordelia Schmid. Convolutional kernel networks. In *Advances in neural information processing systems*, pages 2627–2635, 2014.

[22] Grasgoire Montavon, Mikio L Braun, and Klaus-Robert Matller. Kernel analysis of deep networks. *Journal of Machine Learning Research*, 12(Sep):2563–2581, 2011.

[23] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[24] Alexander G de G Matthews, Mark Rowland, Jiri Hron, Richard E Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. *arXiv preprint arXiv:1804.11271*, 2018.

[25] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

[26] Soufiane Hayou, Arnaud Doucet, and Judith Rousseau. On the impact of the activation function on deep neural networks training. *arXiv preprint arXiv:1902.06853*, 2019.

[27] Jaehoon Lee, Lechao Xiao, Samuel S Schoenholz, Yasaman Bahri, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720*, 2019.

[28] Greg Yang. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv preprint arXiv:1902.04760*, 2019.

[29] David Duvenaud, Oren Rippel, Ryan Adams, and Zoubin Ghahramani. Avoiding pathologies in very deep networks. In *Artificial Intelligence and Statistics*, pages 202–210, 2014.

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

[31] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *arXiv preprint arXiv:1904.11955*, 2019.

[32] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *International conference on algorithmic learning theory*, pages 63–77. Springer, 2005.

[33] Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, and Bernhard Schölkopf. Kernel mean embedding of distributions: A review and beyond. *arXiv preprint arXiv:1605.09522*, 2016.

[34] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

[35] Geoffrey J McLachlan. *Discriminant analysis and statistical pattern recognition*, volume 544. John Wiley & Sons, 2004.

[36] Alessio Ansuini, Alessandro Laio, Jakob H Macke, and Davide Zoccolan. Intrinsic dimension of data representations in deep neural networks. *arXiv preprint arXiv:1905.12784*, 2019.

Appendix A Proof for Theorem 1

Theorem 1: *For any $\mathcal{H}_0$, there exists a Kernel Sequence $\{\phi_{l^\circ}\}_{l=1}^{L}$ parameterized by a set of weights W_l and a set of bandwidths σ_l such that:*

I. $\mathcal{H}_L$ can approach arbitrarily close to $\mathcal{H}^*$ such that for any $L > 1$ and $\delta > 0$ we can achieve

$$\mathcal{H}^* - \mathcal{H}_L \leq \delta, \tag{8}$$

II. as $L \to \infty$, the $\mathcal{H}$-*Sequence* converges to the global optimum, that is

$$\lim_{L\to\infty} \mathcal{H}_L = \mathcal{H}^*, \tag{9}$$

III. the convergence is monotonic where

$$\mathcal{H}_l > \mathcal{H}_{l-1} \quad \forall l. \tag{10}$$

Lemma 1. *Given σ_0 and σ_1 as the σ values from the last layer and the current layer, then there exists a lower bound for $\mathcal{H}_l$, denoted as $\mathscr{L}(\sigma_0, \sigma_1)$ such that*

$$\mathcal{H}_l \geq \mathscr{L}(\sigma_0, \sigma_1). \tag{11}$$

Basic Background, Assumptions, and Notations.

1. The simulation of this theorem for Adversarial and Random data is also publicly available on `https://github.com/anonymous`.
2. Here we show that this bound can be established given the last 2 layers.
3. σ_0 is the σ value of the previous layer
4. σ_1 is the σ value of the current layer
5. τ is the number of classes
6. n is total number of samples
7. n_i is number of samples in the i^{th} class
8. $\mathcal{S}$ is a set of all i, j sample pairs where r_i and r_j belong to the same class.
9. $\mathcal{S}^c$ is a set of all i, j sample pairs where r_i and r_j belong to different same classes.
10. $\mathcal{S}^\beta$ is a set of all i, j sample pairs that belongs to the same β^{th} classes.
11. $r_i^{(\alpha)}$ is the i^{th} sample in the α^{th} class among τ classes.
12. We assume no $r_i \neq r_j$ pair are equal $\forall i \neq j$.
13. Among all $r_i \neq r_j$ pairs, there exists an optimal r_i^*, r_j^* pair where $\langle r_i^*, r_j^* \rangle \geq \langle r_i, r_j \rangle$ $\forall r_i \neq r_i^*$ and $r_j \neq r_j^*$. We denote this maximum inner product as
$$u_{\sigma_0} = \langle r_i^*, r_j^* \rangle. \tag{12}$$
14. In *KChain*, each r_i sample is assumed to be a sample in the RKHS of the Gaussian kernel, therefore all inner products are bounded such that
$$0 \leq \langle r_i, r_j \rangle \leq u_{\sigma_0}. \tag{13}$$
15. We let W be
$$W_s = \frac{1}{\sqrt{\zeta}} \begin{bmatrix} \sum_\iota r_\iota^{(1)} & \sum_\iota r_\iota^{(2)} & \dots & \sum_\iota r_\iota^{(\tau)} \end{bmatrix}. \tag{14}$$
Instead of using an optimal W^* defined as $W^* = \arg\max_W H_l(W)$, we use a suboptimal W_s where each dimension is simply the average direction of each class: $\frac{1}{\sqrt{\zeta}}$ is a normalizing constant $\zeta = ||W_s||_2^2$ that ensures $W_s^T W_s = I$. By using W_s, this implies that the $\mathcal{H}$ we obtain is already a lower bound compare $\mathcal{H}$ obtained by W^*. But, we will use this suboptimal W_s to identify an even lower bound. Note that based on the definition W^*, we have the property $\mathcal{H}(W^*) \geq \mathcal{H}(W)\, \forall W$.

16. We note that the objective $\mathcal{H}$ is

$$\mathcal{H} = \underbrace{\sum_{i,j \in \mathcal{S}} \Gamma_{i,j} e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma_1^2}}}_{\mathscr{W}} - \underbrace{\sum_{i,j \in \mathcal{S}^c} |\Gamma_{i,j}| e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma_1^2}}}_{\mathscr{B}} \tag{15}$$

where we let $\mathscr{W}$ be the summation of terms associated with the within cluster pairs, and let $\mathscr{B}$ be the summation of terms associated with the between cluster pairs.

Proof.

The equation is divided into smaller parts organized into multiple sections.

For sample pairs in $\mathcal{S}$. The first portion of the function can be split into multiple classes where

$$\mathscr{W} = \underbrace{\sum_{\mathcal{S}^1} \Gamma_{i,j} e^{-\frac{(r_i^{(1)} - r_j^{(1)})^T W W^T (r_i^{(1)} - r_j^{(1)})}{2\sigma_1^2}}}_{\mathscr{W}_1} + ... + \underbrace{\sum_{\mathcal{S}^\tau} \Gamma_{i,j} e^{-\frac{(r_i^{(\tau)} - r_j^{(\tau)})^T W W^T (r_i^{(\tau)} - r_j^{(\tau)})}{2\sigma_1^2}}}_{\mathscr{W}_\tau} \tag{16}$$

Realize that to find the lower bound, we need to determine the minimum possible value of each term which translates to **maximum** possible value of each exponent. Without of loss of generality we can find the lower bound for one term and generalize its results to other terms due to their similarity. Let us focus on the numerator of the exponent from $\mathscr{W}_1$. Given W_s as W, our goal is identify the **maximum** possible value for

$$\underbrace{(r_i^{(1)} - r_j^{(1)})^T W}_{\Pi_1} \underbrace{W^T (r_i^{(1)} - r_j^{(1)})}_{\Pi_2}. \tag{17}$$

Zoom in further by looking only at Π_1, we have the following relationships

$$\Pi_1 = \underbrace{r_i^{(1)^T} W}_{\xi_1} - \underbrace{r_j^{(1)^T} W}_{\xi_2} \tag{18}$$

$$\xi_1 = \frac{1}{\sqrt{\zeta}} r_i^{(1)^T} \begin{bmatrix} \sum_\iota r_\iota^{(1)} & \sum_\iota r_\iota^{(2)} & ... & \sum_\iota r_\iota^{(\tau)} \end{bmatrix} \tag{19}$$

$$= \frac{1}{\sqrt{\zeta}} r_i^{(1)^T} \begin{bmatrix} (r_1^{(1)} + ... + r_{n_1}^{(1)}) & ... & (r_1^{(\tau)} + ... + r_{n_\tau}^{(\tau)}) \end{bmatrix} \tag{20}$$

$$\xi_2 = \frac{1}{\sqrt{\zeta}} r_j^{(1)^T} \begin{bmatrix} \sum_\iota r_\iota^{(1)} & \sum_\iota r_\iota^{(2)} & ... & \sum_\iota r_\iota^{(\tau)} \end{bmatrix} \tag{21}$$

$$= \frac{1}{\sqrt{\zeta}} r_j^{(1)^T} \begin{bmatrix} (r_1^{(1)} + ... + r_{n_1}^{(1)}) & ... & (r_1^{(\tau)} + ... + r_{n_\tau}^{(\tau)}) \end{bmatrix} \tag{22}$$

By knowing that the inner product is constrained between $[0, u_{\sigma_0}]$, we know the maximum possible value for ξ_1 and the minimum possible value for ξ_2 to be

$$\xi_1 = \frac{1}{\sqrt{\zeta}} \begin{bmatrix} 1 + (n_1 - 1)u_{\sigma_0} & n_2 u_{\sigma_0} & n_3 u_{\sigma_0} & ... & n_\tau u_{\sigma_0} \end{bmatrix} \tag{23}$$

$$\xi_2 = \frac{1}{\sqrt{\zeta}} \begin{bmatrix} 1 & 0 & 0 & ... & 0 \end{bmatrix}. \tag{24}$$

Which leads to

$$\Pi_1 = \frac{1}{\sqrt{\zeta}} (\xi_1 - \xi_2) = \frac{1}{\sqrt{\zeta}} \begin{bmatrix} (n_1 - 1)u_{\sigma_0} & n_2 u_{\sigma_0} & n_3 u_{\sigma_0} & ... & n_\tau u_{\sigma_0} \end{bmatrix} \tag{25}$$

Since $\Pi_2^T = \Pi_1$ we have

$$\Pi_1 \Pi_2 = \frac{1}{\zeta} [(n_1 - 1)^2 u_{\sigma_0}^2 + n_2^2 u_{\sigma_0}^2 + n_3^2 u_{\sigma_0}^2 + ... + n_\tau^2 u_{\sigma_0}^2] \tag{26}$$

$$= \frac{1}{\zeta} [(n_1 - 1)^2 + n_2^2 + n_3^2 + ... + n_\tau^2] u_{\sigma_0}^2 \tag{27}$$

The lower bound for just the $\mathscr{W}_1$ term emerges as

$$\mathscr{W}_1 \geq \sum_{\mathcal{S}^1} \Gamma_{i,j} e^{-\frac{[(n_1-1)^2+n_2^2+n_3^2+\ldots+n_\tau^2]u_{\sigma_0}^2}{2\zeta\sigma_1^2}}. \tag{28}$$

To further condense the notation, we define the following constant

$$\mathscr{N}_g = \frac{1}{2\zeta}[n_1^2 + n_2^2 + \ldots + (n_g - 1)^2 + \ldots + n_\tau^2]. \tag{29}$$

Therefore, the lower bound for $\mathscr{W}_1$ can be simplified as

$$\mathscr{W}_1 \geq \sum_{\mathcal{S}^1} \Gamma_{i,j} e^{-\frac{\mathscr{N}_1 u_{\sigma_0}^2}{\sigma_1^2}} \tag{30}$$

and the general pattern for any $\mathscr{W}_g$ becomes

$$\mathscr{W}_g \geq \sum_{\mathcal{S}^i} \Gamma_{i,j} e^{-\frac{\mathscr{N}_g u_{\sigma_0}^2}{\sigma_1^2}}. \tag{31}$$

The lower bound for the entire set of $\mathcal{S}$ then becomes

$$\sum_{i,j\in\mathcal{S}} \Gamma_{i,j} e^{-\frac{(r_i-r_j)^T WW^T(r_i-r_j)}{2\sigma_1^2}} = \mathscr{W}_1 + \ldots + \mathscr{W}_\tau \geq \underbrace{\sum_{g=1}^{\tau}\sum_{\mathcal{S}^g} \Gamma_{i,j} e^{-\frac{\mathscr{N}_g u_{\sigma_0}^2}{\sigma_1^2}}}_{\text{Lower bound}}. \tag{32}$$

For sample pairs in $\mathcal{S}^c$. To simplify the notation, we note that

$$-\mathscr{B}_{g_1,g_2} = -\sum_{i\in\mathcal{S}^{g_1}}\sum_{j\in\mathcal{S}^{g_2}} |\Gamma_{i,j}| e^{-\frac{(r_i^{(g_1)}-r_j^{(g_2)})^T WW^T(r_i^{(g_1)}-r_j^{(g_2)})}{2\sigma_1^2}} \tag{33}$$

$$= -\sum_{i\in\mathcal{S}^{g_1}}\sum_{j\in\mathcal{S}^{g_2}} |\Gamma_{i,j}| e^{-\frac{\mathrm{Tr}(W^T((r_i^{(g_1)}-r_j^{(g_1)})((r_i^{(g_1)}-r_j^{(g_2)}))^T W)}{2\sigma_1^2}} \tag{34}$$

$$= -\sum_{i\in\mathcal{S}^{g_1}}\sum_{j\in\mathcal{S}^{g_2}} |\Gamma_{i,j}| e^{-\frac{\mathrm{Tr}(W^T A_{i,j}^{(g_1,g_2)} W)}{2\sigma_1^2}} \tag{35}$$

$$\tag{36}$$

We now derived the lower bound for the sample pairs in $\mathcal{S}^c$. We start by writing out the entire summation sequence for $\mathscr{B}$.

$$\begin{aligned}
\mathscr{B} = &\underbrace{-\sum_{i\in\mathcal{S}^1}\sum_{j\in\mathcal{S}^2} |\Gamma_{i,j}| e^{-\frac{\mathrm{Tr}(W^T A_{i,j}^{(1,2)} W)}{2\sigma_1^2}}}_{\mathscr{B}_{1,2}} - \underbrace{\ldots}_{\mathscr{B}_{g_1\neq g_2}} - \underbrace{\sum_{i\in\mathcal{S}^1}\sum_{j\in\mathcal{S}^\tau} |\Gamma_{i,j}| e^{-\frac{\mathrm{Tr}(W^T A_{i,j}^{(1,\tau)} W)}{2\sigma_1^2}}}_{\mathscr{B}_{1,\tau}} \\
&\underbrace{-\sum_{i\in\mathcal{S}^2}\sum_{j\in\mathcal{S}^1} |\Gamma_{i,j}| e^{-\frac{\mathrm{Tr}(W^T A_{i,j}^{(2,1)} W)}{2\sigma_1^2}}}_{\mathscr{B}_{2,1}} - \underbrace{\ldots}_{\mathscr{B}_{g_1\neq g_2}} - \underbrace{\sum_{i\in\mathcal{S}^2}\sum_{j\in\mathcal{S}^\tau} |\Gamma_{i,j}| e^{-\frac{\mathrm{Tr}(W^T A_{i,j}^{(2,\tau)} W)}{2\sigma_1^2}}}_{\mathscr{B}_{2,\tau}} \\
&\ldots \\
&\underbrace{-\sum_{i\in\mathcal{S}^\tau}\sum_{j\in\mathcal{S}^1} |\Gamma_{i,j}| e^{-\frac{\mathrm{Tr}(W^T A_{i,j}^{(\tau,1)} W)}{2\sigma_1^2}}}_{\mathscr{B}_{\tau,1}} - \underbrace{\ldots}_{\mathscr{B}_{g_1\neq g_2}} - \underbrace{\sum_{i\in\mathcal{S}^{\tau-1}}\sum_{j\in\mathcal{S}^\tau} |\Gamma_{i,j}| e^{-\frac{\mathrm{Tr}(W^T A_{i,j}^{(\tau-1,\tau)} W)}{2\sigma_1^2}}}_{\mathscr{B}_{\tau-1,\tau}}
\end{aligned} \tag{37}$$

Using a similar approach with the terms from $\mathscr{W}$, note that $\mathscr{B}$ is a negative value, so we need to maximize this term to obtain a lower bound. Consequently, the key is to determine the **minimal** possible values for each exponent term. Since every one of them will behave very similarly, we can simply look at the numerator of the exponent from $\mathscr{B}_{1,2}$ and then arrive to a more general conclusion. Given W_s as W, our goal is to identify the **minimal** possible value for

$$\underbrace{(r_i^{(1)} - r_j^{(2)})^T W}_{\Pi_1} \underbrace{W^T (r_i^{(1)} - r_j^{(2)})}_{\Pi_2}. \tag{38}$$

Zoom in further by looking only at Π_1, we have the following relationships

$$\Pi_1 = \underbrace{r_i^{(1)^T} W}_{\xi_1} - \underbrace{r_j^{(2)^T} W}_{\xi_2} \tag{39}$$

$$\xi_1 = \frac{1}{\sqrt{\zeta}} r_i^{(1)^T} \begin{bmatrix} \sum_\iota r_\iota^{(1)} & \sum_\iota r_\iota^{(2)} & ... & \sum_\iota r_\iota^{(\tau)} \end{bmatrix} \tag{40}$$

$$= \frac{1}{\sqrt{\zeta}} r_i^{(1)^T} \begin{bmatrix} (r_1^{(1)} + ... + r_{n_1}^{(1)}) & ... & (r_1^{(\tau)} + ... + r_{n_\tau}^{(\tau)}) \end{bmatrix} \tag{41}$$

$$\xi_2 = \frac{1}{\sqrt{\zeta}} r_j^{(2)^T} \begin{bmatrix} \sum_\iota r_\iota^{(1)} & \sum_\iota r_\iota^{(2)} & ... & \sum_\iota r_\iota^{(\tau)} \end{bmatrix} \tag{42}$$

$$= \frac{1}{\sqrt{\zeta}} r_j^{(2)^T} \begin{bmatrix} (r_1^{(1)} + ... + r_{n_1}^{(1)}) & ... & (r_1^{(\tau)} + ... + r_{n_\tau}^{(\tau)}) \end{bmatrix} \tag{43}$$

By knowing that the inner product is constrained between $[0, u_{\sigma_0}]$, we know the **minimum** possible value for ξ_1 and the **maximum** possible value for ξ_2 to be

$$\xi_1 = \frac{1}{\sqrt{\zeta}} \begin{bmatrix} 1 & 0 & 0 & ... & 0 \end{bmatrix} \tag{44}$$

$$\xi_2 = \frac{1}{\sqrt{\zeta}} \begin{bmatrix} n_1 u_{\sigma_0} & 1 + (n_2 - 1) u_{\sigma_0} & n_3 u_{\sigma_0} & ... & n_\tau u_{\sigma_0} \end{bmatrix} \tag{45}$$

Which leads to

$$\Pi_1 = \frac{1}{\sqrt{\zeta}} (\xi_1 - \xi_2) = \frac{1}{\sqrt{\zeta}} \begin{bmatrix} 1 - n_1 u_{\sigma_0} & -(1 + (n_2 - 1) u_{\sigma_0}) & -n_3 u_{\sigma_0} & ... & -n_\tau u_{\sigma_0} \end{bmatrix} \tag{46}$$

Since $\Pi_2^T = \Pi_1$ we have

$$\Pi_1 \Pi_2 = \frac{1}{\zeta} [(1 - n_1 u_{\sigma_0})^2 + (1 + (n_2 - 1) u_{\sigma_0})^2 + n_3^2 u_{\sigma_0}^2 + ... + n_\tau^2 u_{\sigma_0}^2]. \tag{47}$$

The lower bound for just the $\mathscr{B}_{1,2}$ term emerges as

$$-\mathscr{B}_{1,2} \geq -\sum_{\mathcal{S}^1} \sum_{\mathcal{S}^2} |\Gamma_{i,j}| e^{-\frac{(1 - n_1 u_{\sigma_0})^2 + (1 + (n_2 - 1) u_{\sigma_0})^2 + n_3^2 u_{\sigma_0}^2 + ... + n_\tau^2 u_{\sigma_0}^2}{2\zeta \sigma_1^2}}. \tag{48}$$

To further condense the notation, we define the following function

$$\begin{aligned} \mathscr{N}_{g_1, g_2}(u_{\sigma_0}) = \frac{1}{2\zeta} [&n_1^2 u_{\sigma_0}^2 + n_2^2 u_{\sigma_0}^2 + ... \\ &+ (1 - n_{g_1} u_{\sigma_0})^2 + ... + (1 + (n_{g_2} - 1) u_{\sigma_0})^2 \\ &+ ... + n_\tau^2 u_{\sigma_0}^2]. \end{aligned} \tag{49}$$

Note that while for $\mathcal{S}$, the u_{σ_0} term can be separated out. But here, we cannot, and therefore $\mathscr{N}$ here must be a function of u_{σ_0}. Therefore, the lower bound for $\mathscr{B}_{1,2}$ can be simplified into

$$-\mathscr{B}_{1,2} \geq -\sum_{\mathcal{S}^1} \sum_{\mathcal{S}^2} |\Gamma_{i,j}| e^{-\frac{\mathscr{N}_{1,2}(u_{\sigma_0})}{\sigma_1^2}} \tag{50}$$

and the general pattern for any $\mathscr{B}_{g_1,g_2}$ becomes

$$-\mathscr{B}_{g_1,g_2} \geq -\sum_{\mathcal{S}^{g1}}\sum_{\mathcal{S}^{g2}} \Gamma_{i,j} e^{-\frac{\mathscr{N}_{g_1,g_2}(u_{\sigma_0})}{\sigma_1^2}}. \tag{51}$$

The lower bound for the entire set of $\mathcal{S}^c$ then becomes

$$-\sum_{i,j\in\mathcal{S}^c} |\Gamma_{i,j}| e^{-\frac{(r_i-r_j)^T W W^T (r_i-r_j)}{2\sigma_1^2}} = -\mathscr{B}_{1,2} - \mathscr{B}_{1,3} - ... - \mathscr{B}_{\tau-1,\tau} \tag{52}$$

$$\geq \underbrace{-\sum_{g_1\neq g_2}^{\tau}\sum_{i\in\mathcal{S}^{g_1}}\sum_{j\in\mathcal{S}^{g_2}} |\Gamma_{i,j}| e^{-\frac{\mathscr{N}_{g_1,g_2}(u_{\sigma_0})}{\sigma_1^2}}}_{\text{Lower bound}}. \tag{53}$$

Putting $\mathcal{S}$ and $\mathcal{S}^c$ Together.

$$\mathcal{H} = \mathscr{W} + \mathscr{B} \tag{54}$$

$$\geq \underbrace{\sum_{g=1}^{\tau}\sum_{\mathcal{S}^g} \Gamma_{i,j} e^{-\frac{\mathscr{N}_g u_{\sigma_0}^2}{\sigma_1^2}}}_{\text{Lower bound of } \mathscr{W}} - \underbrace{\sum_{g_1\neq g_2}^{\tau}\sum_{i\in\mathcal{S}^{g_1}}\sum_{j\in\mathcal{S}^{g_2}} |\Gamma_{i,j}| e^{-\frac{\mathscr{N}_{g_1,g_2}(u_{\sigma_0})}{\sigma_1^2}}}_{\text{Lower bound of } \mathscr{B}}. \tag{55}$$

Therefore, we have identified a lower bound that is a function of σ_0 and σ_1 where

$$\mathscr{L}(\sigma_0,\sigma_1) = \sum_{g=1}^{\tau}\sum_{\mathcal{S}^g} \Gamma_{i,j} e^{-\frac{\mathscr{N}_g u_{\sigma_0}^2}{\sigma_1^2}} - \sum_{g_1\neq g_2}^{\tau}\sum_{i\in\mathcal{S}^{g_1}}\sum_{j\in\mathcal{S}^{g_2}} |\Gamma_{i,j}| e^{-\frac{\mathscr{N}_{g_1,g_2}(u_{\sigma_0})}{\sigma_1^2}}. \tag{56}$$

From the lower bound, it is obvious why it is a function of σ_1. The lower bound is also a function of σ_0 because u_{σ_0} is actually a function of σ_0. To specifically clarify this point, we have the next lemma.

□

Lemma 2. *The u_{σ_0} used in Lemma 1 is a function of σ_0 where u_{σ_0} approaches to zero as σ_0 approaches to zero, i.e.*

$$\lim_{\sigma_0\to 0} u_{\sigma_0} = 0. \tag{57}$$

Assumptions and Notations.

1. We use Fig. 4 to help clarify the notations. We here only look at the last 2 layers.
2. We let $\mathcal{H}_0$ be the $\mathcal{H}$ of the last layer, and $\mathcal{H}_1$, the $\mathcal{H}$ of the current layer.
3. The input of the data is X with each sample as x_i, and the output of the previous layer are denoted as r_i. ψ_{σ_0} is the feature map of the previous layer using σ_0 and ψ_{σ_1} corresponds to the current layer.

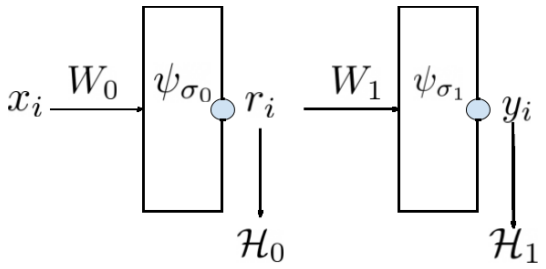

Figure 4: Figure of a 2 layer network.

4. As defined from Lemma 1, among all $r_i \neq r_j$ pairs, there exists an optimal r_i^*, r_j^* pair where $\langle r_i^*, r_j^* \rangle \geq \langle r_i, r_j \rangle \; \forall r_i \neq r_i^*$ and $r_j \neq r_j^*$. We denote this maximum inner product as

$$u_{\sigma_0} = \langle r_i^*, r_j^* \rangle. \tag{58}$$

Proof.

Given Fig. 4, the equation for $\mathcal{H}_0$ is

$$\mathcal{H}_0 = \sum_{i,j \in \mathcal{S}} \Gamma_{i,j} e^{-\frac{(x_i - x_j)^T W W^T (x_i - x_j)}{2\sigma_0^2}} - \sum_{i,j \in \mathcal{S}^c} |\Gamma_{i,j}| e^{-\frac{(x_i - x_j)^T W W^T (x_i - x_j)}{2\sigma_0^2}} \tag{59}$$

$$= \sum_{i,j \in \mathcal{S}} \Gamma_{i,j} \langle \psi_{\sigma_0}(x_i), \psi_{\sigma_0}(x_j) \rangle - \sum_{i,j \in \mathcal{S}^c} |\Gamma_{i,j}| \langle \psi_{\sigma_0}(x_i), \psi_{\sigma_0}(x_j) \rangle \tag{60}$$

Notice that as $\sigma_0 \to 0$, we have

$$\lim_{\sigma_0 \to 0} \langle \psi_{\sigma_0}(x_i), \psi_{\sigma_0}(x_j) \rangle = \begin{cases} 0 & \forall i \neq j \\ 1 & \forall i = j \end{cases}. \tag{61}$$

In other words, as $\sigma_0 \to 0$, the samples r_i in the RKHS of a Gaussian kernel approaches orthogonal to all other samples. Given this fact, it also implies that the σ_0 controls the inner product magnitude in RKHS space of the maximum sample pair r_i^*, r_j^*. We define this maximum inner product as

$$\langle \psi_{\sigma_0}(x_i^*), \psi_{\sigma_0}(x_j^*) \rangle \geq \langle \psi_{\sigma_0}(x_i), \psi_{\sigma_0}(x_j) \rangle \tag{62}$$

or equivalently

$$\langle r_i^*, r_j^* \rangle \geq \langle r_i, r_j \rangle \tag{63}$$

Therefore, given a σ_0, it controls the upper bound of the inner product. Notice that as $\sigma_0 \to 0$, every sample in RKHS becomes orthogonal. Therefore, the upper bound of $\langle r_i, r_j \rangle$ also approaches 0 when $r_i \neq r_j$. From this, we see the relationship

$$\lim_{\sigma_0 \to 0} u_{\sigma_0} = \lim_{\sigma_0 \to 0} \exp -(|.|/\sigma_0^2) = 0 \tag{64}$$

, where |.| is bounded and has a minimum and maximum, because we have finite number of samples.

□

Lemma 3. *Given any fixed $\sigma_1 > 0$, the lower bound $\mathscr{L}(\sigma_0, \sigma_1)$ is a function with respect to σ_0 and as $\sigma_0 \to 0$, $\mathscr{L}(\sigma_0, \sigma_1)$ approaches the function*

$$\mathscr{L}(\sigma_1) = \sum_{g=1}^{\tau} \sum_{\mathcal{S}^g} \Gamma_{i,j} - \sum_{g_1 \neq g_2}^{\tau} \sum_{i \in \mathcal{S}^{g_1}} \sum_{j \in \mathcal{S}^{g_2}} |\Gamma_{i,j}| e^{-\frac{1}{\zeta \sigma_1^2}}. \tag{65}$$

At this point, if we let $\sigma_1 \to 0$, we have

$$\lim_{\sigma_1 \to 0} \mathscr{L}(\sigma_1) = \sum_{i,j \in \mathcal{S}} \Gamma_{i,j} \tag{66}$$

$$= \mathcal{H}^*. \tag{67}$$

Proof.

Given Lemma 2, we know that

$$\lim_{\sigma_0 \to 0} u_{\sigma_0} = 0. \tag{68}$$

Therefore, having $\sigma_0 \to 0$ is equivalent to having $u_{\sigma_0} \to 0$. Since Lemma 1 provide the equation of a lower bound that is a function of u_{σ_0}, this lemma is proven by simply evaluating $\mathscr{L}(\sigma_0, \sigma_1)$ as $u_{\sigma_0} \to 0$. Following these steps, we have

$$\mathscr{L}(\sigma_1) = \lim_{u_{\sigma_0} \to 0} \sum_{g=1}^{\tau} \sum_{\mathcal{S}^g} \Gamma_{i,j} e^{-\frac{\mathscr{N}_g u_{\sigma_0}^2}{\sigma_1^2}} - \sum_{g_1 \neq g_2}^{\tau} \sum_{i \in \mathcal{S}^{g_1}} \sum_{j \in \mathcal{S}^{g_2}} |\Gamma_{i,j}| e^{-\frac{\mathscr{N}_{g_1,g_2}(u_{\sigma_0})}{\sigma_1^2}}, \tag{69}$$

$$= \sum_{g=1}^{\tau} \sum_{\mathcal{S}^g} \Gamma_{i,j} - \sum_{g_1 \neq g_2}^{\tau} \sum_{i \in \mathcal{S}^{g_1}} \sum_{j \in \mathcal{S}^{g_2}} |\Gamma_{i,j}| e^{-\frac{1}{\zeta \sigma_1^2}}. \tag{70}$$

At this point, as $\sigma_1 \to 0$, our lower bound reaches the global maximum

$$\lim_{\sigma_1 \to 0} \mathscr{L}(\sigma_1) = \sum_{g=1}^{\tau} \sum_{\mathcal{S}^g} \Gamma_{i,j} = \sum_{i,j \in \mathcal{S}} \Gamma_{i,j} \tag{71}$$

$$= \mathcal{H}^*. \tag{72}$$

□

Lemma 4. *Given any $\mathcal{H}_{l-2}$, $\delta > 0$, there exists a $\sigma_0 > 0$ and $\sigma_1 > 0$ such that*

$$\mathcal{H}^* - \mathcal{H}_l \leq \delta. \tag{73}$$

Proof.

Observation 1.

Note that the objective of $\mathcal{H}_l$ is

$$\begin{aligned} \mathcal{H}_l = \max_{W} \sum_{i,j \in \mathcal{S}} \Gamma_{i,j} e^{-\frac{(r_i^{(\mathcal{S})} - r_j^{(\mathcal{S})})^T W W^T (r_i^{(\mathcal{S})} - r_j^{(\mathcal{S})})}{2\sigma_1^2}} \\ - \sum_{i,j \in \mathcal{S}^c} |\Gamma_{i,j}| e^{-\frac{(r_i^{(\mathcal{S}^c)} - r_j^{(\mathcal{S}^c)})^T W W^T (r_i^{(\mathcal{S}^c)} - r_j^{(\mathcal{S}^c)})}{2\sigma_1^2}}. \end{aligned} \tag{74}$$

Since the Gaussian kernel is bounded between 0 and 1, the theoretical maximum of $\mathcal{H}^*$ is when the kernel is 1 for $\mathcal{S}$ and 0 for $\mathcal{S}^c$ with the theoretical maximum as $\mathcal{H}^* = \sum_{i,j \in \mathcal{S}} \Gamma_{i,j}$. Therefore Eq. (73) inequality is equivalent to

$$\sum_{i,j \in \mathcal{S}} \Gamma_{i,j} - \mathcal{H}_l \leq \delta. \tag{75}$$

Observation 2.

If we choose a σ_0 such that

$$\mathscr{L}^*(\sigma_1) - \mathscr{L}(\sigma_0, \sigma_1) \leq \frac{\delta}{2} \quad \text{and} \quad \mathcal{H}^* - \mathscr{L}^*(\sigma_1) \leq \frac{\delta}{2} \tag{76}$$

then we have identified the condition where $\sigma_0 > 0$ and $\sigma_1 > 0$ such that

$$\sum_{i,j \in \mathcal{S}} \Gamma_{i,j} - \mathscr{L}(\sigma_0, \sigma_1) \leq \delta. \tag{77}$$

Note that the $\mathscr{L}^*(\sigma_1)$ is a continuous function of σ_1. Therefore, a σ_1 exists such that $\mathscr{L}^*(\sigma_1)$ can be set arbitraty close to $\mathcal{H}^*$. Hence, we choose an σ_1 that has the following property:

$$\mathcal{H}^* - \mathscr{L}^*(\sigma_1) \leq \frac{\delta}{2}. \tag{78}$$

We next fix σ_1, we also know $\mathscr{L}(\sigma_0, \sigma_1)$ is a continuous function of σ_0, and it has a limit $\mathscr{L}^*(\sigma_1)$ as σ_0 approaches to 0, hence there exits a σ_0, where

$$\mathscr{L}^*(\sigma_1) - \mathscr{L}(\sigma_0, \sigma_1) \leq \frac{\delta}{2} \tag{79}$$

Then we have:

$$\mathscr{L}^*(\sigma_1) - \mathscr{L}(\sigma_0, \sigma_1) \leq \frac{\delta}{2} \quad \text{and} \quad \mathcal{H}^* - \mathscr{L}^*(\sigma_1) \leq \frac{\delta}{2}. \tag{80}$$

By adding the two $\frac{\delta}{2}$, we conclude the proof.

□

Lemma 5. *There exists a Kernel Sequence $\{\phi_{l^\circ}\}_{l=1}^{L}$ parameterized by a set of weights W_l and a set of bandwidths σ_l such that*

$$\lim_{l\to\infty} \mathcal{H}_l = \mathcal{H}^*, \quad \mathcal{H}_{l+1} > \mathcal{H}_l \quad \forall l \tag{81}$$

Before, the proof, we use the following figure, Fig. 5, to illustrate the relationship between *Kernel Sequence* $\{\phi_{l^\circ}\}_{l=1}^{L}$ that generates the $\mathcal{H}$*-Sequence* $\{\mathcal{H}_l\}_{l=1}^{L}$. By solving a network greedily, we separate the network into L separable problems. At each additional layer, we rely on the weights learned from the previous layer. At each network, we find σ_{l-1}, σ_l, and W_l for the next network. We also note that since we only need to prove the existence of a solution, this proof is done by ***Proof by Construction***, i.e, we only need to show an example of its existence. Therefore, this proof consists of us constructing a $\mathcal{H}$*-Sequence* which satisfies the lemma.

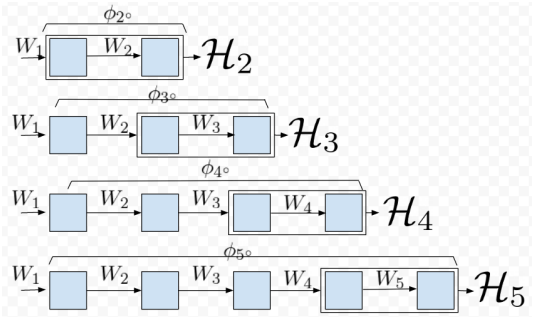

Figure 5: Relating *Kernel Sequence* to $\mathcal{H}$*-Sequence*.

Proof.

We first note that from Lemma 4, we have previously proven given any $\mathcal{H}_{l-2}$, $\delta > 0$, there exists a $\sigma_0 > 0$ and $\sigma_1 > 0$ such that

$$\mathcal{H}^* - \mathcal{H}_l \leq \delta_l. \tag{82}$$

This implies that based on Fig. 5, at any given layer, we could reach arbitrarily close to $\mathcal{H}^*$. Given this, we list the 2 steps to build the $\mathcal{H}$*-Sequence*.

Step 1: Define $\{\mathcal{E}_n\}_{n=1}^{\infty}$ as a sequence of numbers $\mathcal{H}^* - \frac{\mathcal{H}^* - \mathcal{H}_0}{n}$ on the real line. We have the following properties for this sequence:

$$\lim_{n\to\infty} \mathcal{E}_n = \mathcal{H}^*, \quad \mathcal{E}_1 = \mathcal{H}_0. \tag{83}$$

Using these two properties, for any $\mathcal{H}_{l-1} \in [\mathcal{H}_0, \mathcal{H}^*]$ there exist an unique n, where

$$\mathcal{E}_n \leq \mathcal{H}_{l-1} < \mathcal{E}_{n+1}. \tag{84}$$

Step 2: For any given l, we choose δ_l to satisfies Eq. (82) by the following procedure, First find an n that satisfies

$$\mathcal{E}_n \leq \mathcal{H}_{l-1} < \mathcal{E}_{n+1}, \tag{85}$$

and second define δ_l to be

$$\delta_l = \mathcal{H}^* - \mathcal{E}_{n+1}. \tag{86}$$

To satisfy Eq. (82), the following must be true.

$$\mathcal{H}^* - \mathcal{H}_{l-1} \leq \delta_{l-1}. \tag{87}$$

and further we found n such that

$$\mathcal{E}_n \leq \mathcal{H}_{l-1} < \mathcal{E}_{n+1} \implies \mathcal{H}^* - \mathcal{E}_n \geq \mathcal{H}^* - \mathcal{H}_{l-1} > \mathcal{H}^* - \mathcal{E}_{n+1}. \tag{88}$$

Thus combining Eq. (86), Eq. (87), and Eq. (88) we have

$$\delta_{l-1} > \delta_l. \tag{89}$$

Therefore, $\{\delta_l\}$ is a decreasing sequence.

Step 3: Note that $\{\mathcal{E}_n\}$ is a converging sequence where

$$\lim_{n\to\infty} \mathcal{H}^* - \frac{\mathcal{H}^* - \mathcal{H}_0}{n} = \mathcal{H}^*. \tag{90}$$

Therefore, $\{\Delta_n\} = \mathcal{H}^* - \{\mathcal{E}_n\}$ is also a converging sequence where

$$\lim_{n\to\infty} \mathcal{H}^* - \mathcal{H}^* + \frac{\mathcal{H}^* - \mathcal{H}_0}{n} = 0 \tag{91}$$

and $\{\delta_l\}$ is a subsequence of $\{\Delta_l\}$. Since any subsequence of a converging sequence also converges to the same limit, we know that

$$\lim_{l\to\infty} \delta_l = 0. \tag{92}$$

Following this construction, if we always choose $\mathcal{H}_l$ such that

$$\mathcal{H}^* - \mathcal{H}_l \leq \delta_l. \tag{93}$$

As $l \to \infty$, the inequality becomes

$$\mathcal{H}^* - \lim_{l\to\infty} \mathcal{H}_l \leq \lim_{l\to\infty} \delta_l, \tag{94}$$

$$\leq 0. \tag{95}$$

Since we know that

$$\mathcal{H}^* - \mathcal{H}_l \geq 0 \,\forall l. \tag{96}$$

The condition of

$$0 \leq \mathcal{H}^* - \lim_{l\to\infty} \mathcal{H}_l \leq 0 \tag{97}$$

is true only if

$$\mathcal{H}^* - \lim_{l\to\infty} \mathcal{H}_l = 0. \tag{98}$$

This allows us to conclude

$$\mathcal{H}^* = \lim_{l\to\infty} \mathcal{H}_l. \tag{99}$$

Proof of the Monotonic Improvement.

Given Eq. (84) and Eq. (86), at each step we have the following:

$$\mathcal{H}_{l-1} < \mathcal{E}_{n+1} \tag{100}$$

$$\leq \mathcal{H}^* - \delta_l. \tag{101}$$

Rearranging this inequality, we have

$$\delta_l < \mathcal{H}^* - \mathcal{H}_{l-1}. \tag{102}$$

By combining the inequalities from Eq. (102) and Eq. (93), we have the following relationships.

$$\mathcal{H}^* - \mathcal{H}_l \leq \delta_l < \mathcal{H}^* - \mathcal{H}_{l-1} \tag{103}$$

$$\mathcal{H}^* - \mathcal{H}_l < \mathcal{H}^* - \mathcal{H}_{l-1} \tag{104}$$

$$-\mathcal{H}_l < -\mathcal{H}_{l-1} \tag{105}$$

$$\mathcal{H}_l > \mathcal{H}_{l-1}, \tag{106}$$

$$\tag{107}$$

which concludes the proof of theorem.

□

Lemma 6. *Given $\frac{1}{\sqrt{\zeta}}$ as a normalizing constant for $W_s = \frac{1}{\sqrt{\zeta}} \sum_\alpha r_\alpha$ such that $W^T W = I$, then W_s is not guaranteed to be the optimal solution for the HSIC objective.*

Proof. We start with the Lagrangian

$$\mathcal{L} = -\sum_{i,j} \Gamma_{i,j} e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma^2}} - \mathrm{Tr}(\Lambda(W^T W - I)). \tag{108}$$

If we now take the derivative with respect to the Lagrange, we get

$$\nabla \mathcal{L} = \frac{1}{\sigma^2} \sum_{i,j} \Gamma_{i,j} e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma^2}} (r_i - r_j)(r_i - r_j)^T W - 2W\Lambda. \tag{109}$$

By setting the gradient to 0, we have

$$\left[\frac{1}{2\sigma^2} \sum_{i,j} \Gamma_{i,j} e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma^2}} (r_i - r_j)(r_i - r_j)^T \right] W = W\Lambda. \tag{110}$$

$$\mathcal{Q}_l W = W\Lambda. \tag{111}$$

From Eq. (111), we see that W is only the optimal solution when W is the eigenvector of Q_l. Therefore, by setting W to $W_s = \frac{1}{\sqrt{\zeta}} \sum_\alpha r_\alpha$, it is not guaranteed to yield an optimal.

□

Appendix B Proof for Theorem 2

Theorem 2: *As $l \to \infty$ and $\mathcal{H}_l \to \mathcal{H}^*$, the following properties are satisfied:*

I the scatter ratio approaches 0 where

$$\lim_{l \to \infty} \frac{\mathrm{Tr}(S_w^l)}{\mathrm{Tr}(S_b^l)} = 0 \tag{112}$$

II the *Kernel Sequence* converges to the following kernel:

$$\lim_{l \to \infty} \mathcal{K}(x_i, x_j)^l = \mathcal{K}^* = \begin{cases} 0 & \forall i, j \in \mathcal{S}^c \\ 1 & \forall i, j \in \mathcal{S} \end{cases}. \tag{113}$$

Proof. We start by proving condition II starting from the $\mathcal{H}$ objective using a GK

$$\max_W \sum_{i,j \in \mathcal{S}} \Gamma_{i,j} \mathcal{K}_W(r_i, r_j) - \sum_{i,j \in \mathcal{S}^c} |\Gamma_{i,j}| \mathcal{K}_W(r_i, r_j) \tag{114}$$

$$\max_W \sum_{i,j \in \mathcal{S}} \Gamma_{i,j} e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma^2}} - \sum_{i,j \in \mathcal{S}^c} |\Gamma_{i,j}| e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma^2}} \tag{115}$$

Given that $\mathcal{H}_l \to \mathcal{H}^*$, and the fact that $0 \leq \mathcal{K}_W \leq 1$, this implies that the following condition must be true:

$$\mathcal{H}^* = \sum_{i,j \in \mathcal{S}} \Gamma_{i,j} = \sum_{i,j \in \mathcal{S}} \Gamma_{i,j}(1) - \sum_{i,j \in \mathcal{S}^c} |\Gamma_{i,j}|(0). \tag{116}$$

Based on Eq. (82), our construction at each layer ensures to satisfy

$$\mathcal{H}^* - \mathcal{H}_l \leq \delta_l. \tag{117}$$

Substituting the definition of $\mathcal{H}^*$ and $\mathcal{H}_l$, we have

$$\sum_{i,j \in \mathcal{S}} \Gamma_{i,j}(1) - \left[\sum_{i,j \in \mathcal{S}} \Gamma_{i,j} \mathcal{K}_W(r_i, r_j) - \sum_{i,j \in \mathcal{S}^c} |\Gamma_{i,j}| \mathcal{K}_W(r_i, r_j) \right] \leq \delta_l \tag{118}$$

$$\sum_{i,j \in \mathcal{S}} \Gamma_{i,j}(1 - \mathcal{K}_W(r_i, r_j)) + \sum_{i,j \in \mathcal{S}^c} |\Gamma_{i,j}| \mathcal{K}_W(r_i, r_j) \leq \delta_l. \tag{119}$$

Since every term within the summation in Eq. (119) is positive, this implies

$$1 - \mathcal{K}_W(r_i, r_j) \leq \delta_l \quad i, j \in \mathcal{S} \tag{120}$$

$$\mathcal{K}_W(r_i, r_j) \leq \delta_l \quad i, j \in \mathcal{S}^c. \tag{121}$$

So as $l \to \infty$ and $\delta_l \to 0$, every component getting closer to limit Kernel, i.e, taking the limit from both sides and using the fact that is proven is theorem 1 $\lim_{l\to\infty} \delta_l = 0$ leads to

$$\lim_{l\to\infty} 1 \leq \mathcal{K}_W(r_i, r_j) \quad i, j \in \mathcal{S} \tag{122}$$

$$\lim_{l\to\infty} \mathcal{K}_W(r_i, r_j) \leq 0 \quad i, j \in \mathcal{S}^c \tag{123}$$

both terms must instead be strictly equality. Therefore, we see that at the limit point $\mathcal{K}_W$ would have the form

$$\mathcal{K}^* = \begin{cases} 0 & \forall i, j \in \mathcal{S}^c \\ 1 & \forall i, j \in \mathcal{S} \end{cases} . \tag{124}$$

First Property:

Using Eq. (120) and Eq. (121) we have:

$$1 - \delta_l \leq e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma^2}} \quad i, j \in \mathcal{S} \tag{125}$$

$$e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma^2}} \leq \delta_l \quad i, j \in \mathcal{S}^c. \tag{126}$$

As $\lim_{l\to\infty} \delta_l = 0$, taking the limit from both side leads to:

$$\begin{cases} e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma^2}} = 1 & \forall i, j \in \mathcal{S} \\ e^{-\frac{(r_i - r_j)^T W W^T (r_i - r_j)}{2\sigma^2}} = 0 & \forall i, j \in \mathcal{S}^c \end{cases} . \tag{127}$$

If we take the log of the conditions, we get

$$\begin{cases} \frac{1}{2\sigma^2}(r_i - r_j)^T W W^T (r_i - r_j) = 0 & \forall i, j \in \mathcal{S} \\ \frac{1}{2\sigma^2}(r_i - r_j)^T W W^T (r_i - r_j) = \infty & \forall i, j \in \mathcal{S}^c \end{cases} . \tag{128}$$

This implies that as $l \to \infty$ we have

$$\lim_{l\to\infty} \sum_{i,j\in\mathcal{S}} \frac{1}{2\sigma^2}(r_i - r_j)^T W W^T (r_i - r_j) = \lim_{l\to\infty} \mathrm{Tr}(S_w) = 0. \tag{129}$$

$$\lim_{l\to\infty} \sum_{i,j\in\mathcal{S}^c} \frac{1}{2\sigma^2}(r_i - r_j)^T W W^T (r_i - r_j) = \lim_{l\to\infty} \mathrm{Tr}(S_b) = \infty, \tag{130}$$

This yields the ratio

$$\lim_{\mathcal{H}_l \to \mathcal{H}^*} \frac{\mathrm{Tr}(S_w)}{\mathrm{Tr}(S_b)} = \frac{0}{\infty} = 0. \tag{131}$$

□

Appendix C Proof for Corollary 1 and 2

Corollary 1: *Given $\mathcal{H}_l \to \mathcal{H}^*$, the network output in IDS solves MSE via a translation of labels.*

Proof.

As $\mathcal{H}_l \to \mathcal{H}^*$, Thm. 2 shows that sample of the same class are mapped into the same point. Assuming that ϕ has mapped the sample into c points $\alpha = [\alpha_1, ..., \alpha_c]$ that's different from the truth label $\xi = [\xi_1, ..., \xi_c]$. Then the MSE objective is minimized by translating the ϕ output by

$$\xi - \alpha. \tag{132}$$

□

Corollary 2: *Given $\mathcal{H}_l \to \mathcal{H}^*$, the network output in RKHS solves CE via a change of bases.*

Assumptions, and Notations.

1. n is the number of samples.
2. τ is the number of classes.
3. $y_i \in \mathbb{R}^\tau$ is the ground truth label for the i^{th} sample. It is one-hot encoded where only the j^{th} element is 1 if x_i belongs to the j^{th} class, all other elements would be 0.
4. We denote ϕ as the network, and $\hat{y}_i \in \mathbb{R}^\tau$ as the network output where $\hat{y}_i = \phi(x_i)$. We also assume that $\hat{y}_i$ is constrained on a probability simplex where $1 = \hat{y}_i^T \mathbf{1}_n$.
5. We denote the j^{th} element of y_i, and $\hat{y}_i$ as $y_{i,j}$ and $\hat{y}_{i,j}$ respectively.
6. We define

   Orthogonality Condition: A set of samples $\{\hat{y}_1, ..., \hat{y}_n\}$ satisfies the orthogonality condition if

   $$\begin{cases} \langle \hat{y_i}, \hat{y_j} \rangle = 1 & \forall \quad i, j \text{ same class} \\ \langle \hat{y_i}, \hat{y_j} \rangle = 0 & \forall \quad i, j \text{ not in the same class} \end{cases}. \tag{133}$$

7. We define the Cross-Entropy objective as

$$\arg\min_{\phi} - \sum_{i=1}^{n} \sum_{j=1}^{\tau} y_{i,j} \log(\phi(x_i)_{i,j}). \tag{134}$$

Proof.

From Thm. 2, we know that the network ϕ output, $\{\hat{y}_1, \hat{y}_2, ..., \hat{y}_n\}$, satisfy the orthogonality condition at $\mathcal{H}^*$. Then there exists a set of orthogonal bases represented by $\Xi = [\xi_1, \xi_2, ..., \xi_c]$ that maps $\{\hat{y}_1, \hat{y}_2, ..., \hat{y}_n\}$ to simulate the output of a softmax layer. Let $\xi_i = \hat{y}_j, j \in \mathcal{S}^i$, i.e., for the i_{th} class we arbitrary choose one of the samples from this class and assigns ξ_i of that class to be equal to the sample's output. Realize in our problem we have $< \hat{y}_i, \hat{y}_i >= 1$, so if $< \hat{y}_i, \hat{y}_j >= 1$, then subtracting these two would lead to $< \hat{y}_i, \hat{y}_i - \hat{y}_j >= 0$, which is the same as $\hat{y}_i = \hat{y}_j$. So this representation is well-defined and its independent of choices of the sample from each group if they satisfy orthogonality condition. Now we define transformed labels, Y as:

$$Y = \hat{Y}\Xi. \tag{135}$$

Note that $Y = [y_1, y_2, ..., y_n]^T$ which each y_i is a one hot vector representing the class membership of i sample in c classes. Since given Ξ as the change of basis, we can match $\hat{Y}$ to Y exactly, CE is minimized.

□

Appendix D Dataset Details

No samples were excludes from any of the dataset.

Wine. This dataset has 13 features, 178 samples, and 3 classes. The features are continuous and heavily unbalanced in magnitude. The dataset can be downloaded at `https://archive.ics.uci.edu/ml/datasets/wine`.

Divorce. This dataset has 54 features, 170 samples, and 2 classes. The features are discrete and balanced in magnitude. The dataset can be downloaded at `https://archive.ics.uci.edu/ml/datasets/Divorce+Predictors+data+set`.

Car. This dataset has 6 features, 1728 samples and 2 classes. The features are discrete and balanced in magnitude. The dataset can be downloaded at `https://archive.ics.uci.edu/ml/datasets/Car+Evaluation`.

Cancer. This dataset has 9 features, 683 samples, and 2 classes. The features are discrete and unbalanced in magnitude. The dataset can be downloaded at `https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)`.

Face. This dataset consists of images of 20 people in various poses. The 624 images are vectorized into 960 features. The dataset can be downloaded at `https://archive.ics.uci.edu/ml/datasets/CMU+Face+Images`.

Random. This dataset has 2 features, 80 samples and 2 classes. It is generate with a gaussian distribution where half of the samples are randomly labeled as 1 or 0.

Adversarial. This dataset has 2 features, 80 samples and 2 classes. It is generate with the following code:

```
#!/usr/bin/env python

n = 40
X1 = np.random.rand(n,2)
X2 = X1 + 0.01*np.random.randn(n,2)

X = np.vstack((X1,X2))
Y = np.vstack(( np.zeros((n,1)), np.ones((n,1)) ))
```

Appendix E W_l Dimensions for each 10 Fold of each Dataset

We report the input and output dimensions of each W_l for every layer of each dataset in the form of (α, β); the corresponding dimension becomes $W_l \in \mathbb{R}^{\alpha \times \beta}$. Since each dataset consists of 10-folds, the network structure for each fold is reported. We note that the input of the 1st layer is the dimension of the original data. However, after the first layer, the width of the RFF becomes the output of each layer; here we use 300.

The β value is chosen during the ISM algorithm. By keeping only the most dominant eigenvector of the Φ matrix, the output dimension of each layer corresponds with the rank of Φ. It can be seen from each dataset that the first layer significantly expands the rank. The expansion is generally followed by a compression of fewer and fewer eigenvalues. These results conform with the observations made by Montavon et al. [22] and Ansuini et al. [36].

Data	Layer 1	Layer 2	Layer 3	Layer 4
adversarial 1	(2, 2)	(300, 61)	(300, 35)	
adversarial 2	(2, 2)	(300, 61)	(300, 35)	
adversarial 3	(2, 2)	(300, 61)	(300, 8)	(300, 4)
adversarial 4	(2, 2)	(300, 61)	(300, 29)	
adversarial 5	(2, 2)	(300, 61)	(300, 29)	
adversarial 6	(2, 2)	(300, 61)	(300, 7)	(300, 4)
adversarial 7	(2, 2)	(300, 61)	(300, 34)	
adversarial 8	(2, 2)	(300, 12)	(300, 61)	(300, 30)
adversarial 9	(2, 2)	(300, 61)	(300, 33)	
adversarial 10	(2, 2)	(300, 61)	(300, 33)	

Data	Layer 1	Layer 2	Layer 3
Random 1	(3, 3)	(300, 47)	(300, 25)
Random 2	(3, 3)	(300, 46)	(300, 25)
Random 3	(3, 3)	(300, 46)	(300, 25)
Random 4	(3, 3)	(300, 47)	(300, 4)
Random 5	(3, 3)	(300, 47)	(300, 25)
Random 6	(3, 3)	(300, 45)	(300, 23)
Random 7	(3, 3)	(300, 45)	(300, 25)
Random 8	(3, 3)	(300, 45)	(300, 21)
Random 9	(3, 3)	(300, 45)	(300, 26)
Random 10	(3, 3)	(300, 47)	(300, 25)

Data	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
spiral 1	(2, 2)	(300, 15)	(300, 6)	(300, 7)	(300, 6)	
spiral 2	(2, 2)	(300, 13)	(300, 6)	(300, 7)	(300, 6)	(300, 6)
spiral 3	(2, 2)	(300, 12)	(300, 6)	(300, 7)	(300, 6)	(300, 6)
spiral 4	(2, 2)	(300, 13)	(300, 6)	(300, 7)	(300, 6)	(300, 6)
spiral 5	(2, 2)	(300, 13)	(300, 6)	(300, 7)	(300, 6)	
spiral 6	(2, 2)	(300, 14)	(300, 6)	(300, 7)	(300, 6)	
spiral 7	(2, 2)	(300, 14)	(300, 6)	(300, 7)	(300, 6)	
spiral 8	(2, 2)	(300, 14)	(300, 6)	(300, 7)	(300, 6)	(300, 6)
spiral 9	(2, 2)	(300, 13)	(300, 6)	(300, 7)	(300, 6)	
spiral 10	(2, 2)	(300, 14)	(300, 6)	(300, 7)	(300, 6)	

Data	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
wine 1	(13, 11)	(300, 76)	(300, 6)	(300, 7)	(300, 6)	(300, 6)
wine 2	(13, 11)	(300, 76)	(300, 6)	(300, 6)	(300, 6)	(300, 6)
wine 3	(13, 11)	(300, 75)	(300, 6)	(300, 7)	(300, 6)	(300, 6)
wine 4	(13, 11)	(300, 76)	(300, 6)	(300, 6)	(300, 6)	(300, 6)
wine 5	(13, 11)	(300, 74)	(300, 6)	(300, 7)	(300, 6)	(300, 6)
wine 6	(13, 11)	(300, 74)	(300, 6)	(300, 6)	(300, 6)	(300, 6)
wine 7	(13, 11)	(300, 74)	(300, 6)	(300, 6)	(300, 6)	(300, 6)
wine 8	(13, 11)	(300, 75)	(300, 6)	(300, 7)	(300, 6)	(300, 6)
wine 9	(13, 11)	(300, 75)	(300, 6)	(300, 8)	(300, 6)	(300, 6)
wine 10	(13, 11)	(300, 76)	(300, 6)	(300, 7)	(300, 6)	(300, 6)

Data	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
car 1	(6, 6)	(300, 96)	(300, 6)	(300, 8)	(300, 6)	
car 2	(6, 6)	(300, 96)	(300, 6)	(300, 8)	(300, 6)	
car 3	(6, 6)	(300, 91)	(300, 6)	(300, 8)	(300, 6)	
car 4	(6, 6)	(300, 88)	(300, 6)	(300, 8)	(300, 6)	(300, 6)
car 5	(6, 6)	(300, 94)	(300, 6)	(300, 8)	(300, 6)	
car 6	(6, 6)	(300, 93)	(300, 6)	(300, 7)		
car 7	(6, 6)	(300, 92)	(300, 6)	(300, 8)	(300, 6)	
car 8	(6, 6)	(300, 95)	(300, 6)	(300, 7)	(300, 6)	
car 9	(6, 6)	(300, 96)	(300, 6)	(300, 9)	(300, 6)	
car 10	(6, 6)	(300, 99)	(300, 6)	(300, 8)	(300, 6)	

Data	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5
divorce 1	(54, 35)	(300, 44)	(300, 5)	(300, 5)	
divorce 2	(54, 35)	(300, 45)	(300, 4)	(300, 4)	
divorce 3	(54, 36)	(300, 49)	(300, 6)	(300, 6)	
divorce 4	(54, 36)	(300, 47)	(300, 7)	(300, 6)	
divorce 5	(54, 35)	(300, 45)	(300, 6)	(300, 6)	
divorce 6	(54, 36)	(300, 47)	(300, 6)	(300, 6)	
divorce 7	(54, 35)	(300, 45)	(300, 6)	(300, 6)	(300, 4)
divorce 8	(54, 36)	(300, 47)	(300, 6)	(300, 7)	(300, 4)
divorce 9	(54, 36)	(300, 47)	(300, 5)	(300, 5)	
divorce 10	(54, 36)	(300, 47)	(300, 6)	(300, 6)	

Data	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7	Layer 8	Layer 9	Layer 10
cancer 1	(9, 8)	(300, 90)	(300, 5)	(300, 6)	(300, 6)	(300, 5)	(300, 4)	(300, 5)	(300, 6)	(300, 6)
cancer 2	(9, 8)	(300, 90)	(300, 6)	(300, 7)	(300, 8)	(300, 11)	(300, 8)	(300, 4)		
cancer 3	(9, 8)	(300, 88)	(300, 5)	(300, 6)	(300, 7)	(300, 7)	(300, 6)	(300, 4)		
cancer 4	(9, 8)	(300, 93)	(300, 6)	(300, 7)	(300, 9)	(300, 11)	(300, 8)			
cancer 5	(9, 8)	(300, 93)	(300, 9)	(300, 10)	(300, 10)	(300, 11)	(300, 9)	(300, 7)		
cancer 6	(9, 8)	(300, 92)	(300, 7)	(300, 8)	(300, 8)	(300, 7)	(300, 7)			
cancer 7	(9, 8)	(300, 90)	(300, 4)	(300, 4)	(300, 5)	(300, 6)	(300, 6)	(300, 6)	(300, 6)	
cancer 8	(9, 8)	(300, 88)	(300, 5)	(300, 6)	(300, 7)	(300, 8)	(300, 7)	(300, 6)		
cancer 9	(9, 8)	(300, 88)	(300, 5)	(300, 7)	(300, 7)	(300, 7)	(300, 7)			
cancer 10	(9, 8)	(300, 97)	(300, 9)	(300, 11)	(300, 12)	(300, 13)	(300, 6)			

Data	Layer 1	Layer 2	Layer 3	Layer 4
face 1	(960, 233)	(300, 74)	(300, 73)	(300, 46)
face 2	(960, 231)	(300, 75)	(300, 73)	(300, 43)
face 3	(960, 231)	(300, 76)	(300, 73)	(300, 44)
face 4	(960, 232)	(300, 76)	(300, 74)	(300, 44)
face 5	(960, 231)	(300, 77)	(300, 73)	(300, 43)
face 6	(960, 232)	(300, 74)	(300, 72)	(300, 47)
face 7	(960, 232)	(300, 76)	(300, 73)	(300, 45)
face 8	(960, 230)	(300, 74)	(300, 74)	(300, 44)
face 9	(960, 233)	(300, 76)	(300, 76)	(300, 45)
face 10	(960, 231)	(300, 76)	(300, 70)	(300, 43)

Appendix F Sigma Values used for Random and Adversarial Simulation

The simulation of Thm. 1 as shown in Fig. 1 spread the improvement across multiple layers. The σ_l and $\mathcal{H}_l$ values are recorded here. We note that σ_l are reasonably large and not approaching 0 and the improvement of $\mathcal{H}_l$ is monotonic.

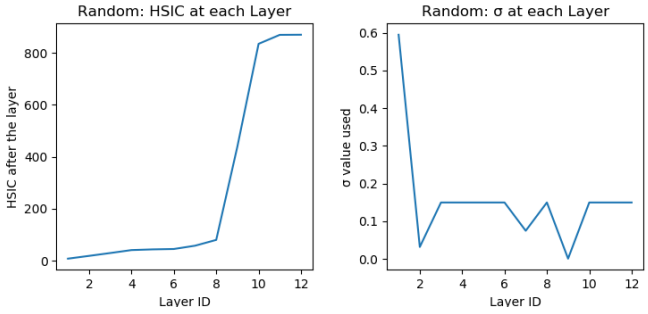

Figure 6

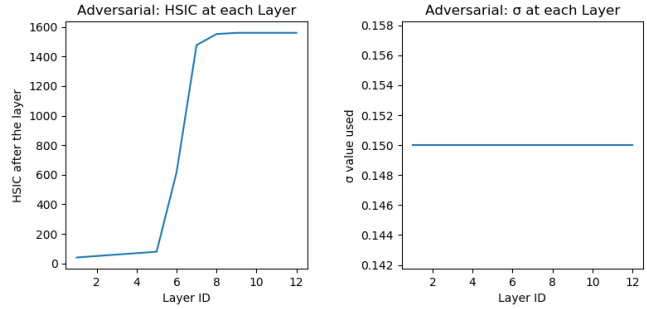

Figure 7

Given a sufficiently small σ_0 and σ_1, Thm. 1 claims that it can come arbitrarily close to the global optimal using a minimum of 2 layers. We here simulate 2 layers using a relatively small σ values ($\sigma_0 = 10^{-5}$) on the Random (left) and Adversarial (right) data and display the results of the 2 layers below. Notice that given 2 layer, it generated a clearly separable clusters that are pushed far apart.

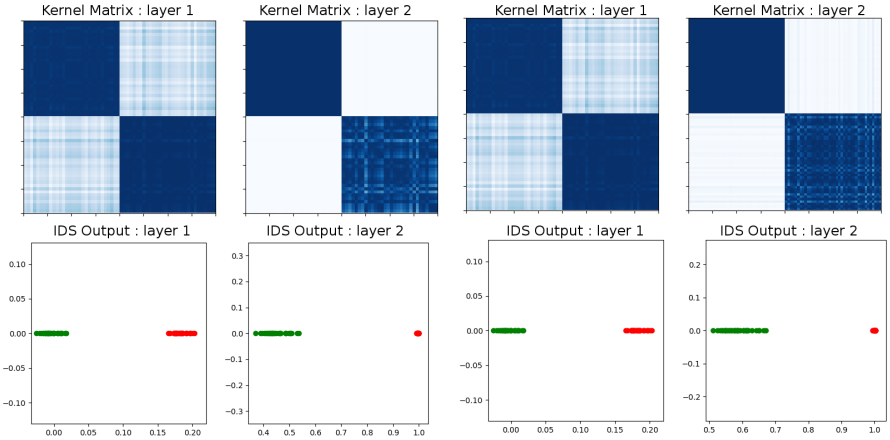

Figure 8: Random Dataset with 2 layers and $\sigma = 10^{-5}$

Figure 9: Adversarial Dataset with 2 layers and $\sigma = 10^{-5}$

Appendix G Graphs of Kernel Sequences

A representation of the *Kernel Sequence* are displayed in the figures below for each dataset. The samples of the kernel matrix are previously organized to form a block structure by placing samples of the same class adjacent to each other. Since the Gaussian kernel is restricted to values between 0 and 1, we let white and dark blue be 0 and 1 respectively where the gradients reflect values in between. Our theorems predict that the *Kernel Sequence* will evolve from an uninformative kernel into a highly discriminating kernel of perfect block structures.

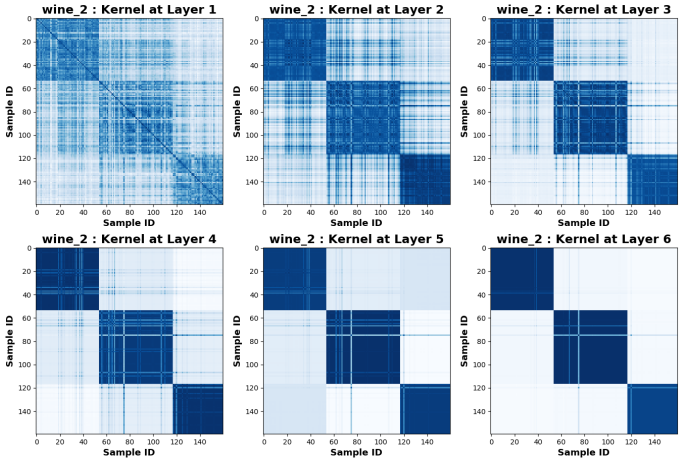

Figure 10: The kernel sequence for the wine dataset.

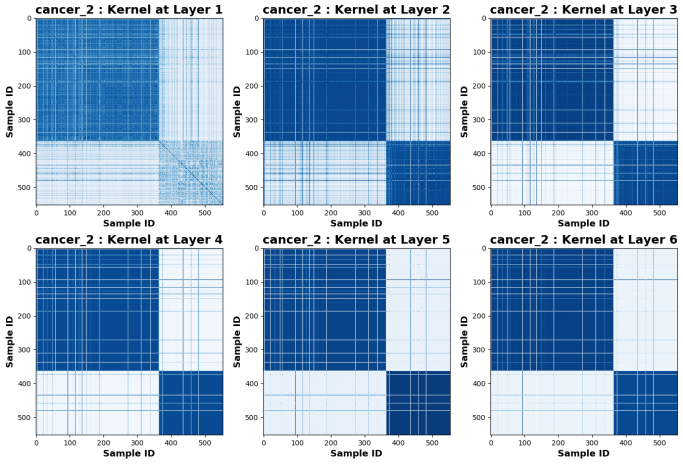

Figure 11: The kernel sequence for the cancer dataset.

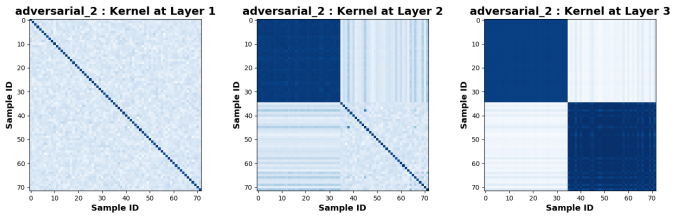

Figure 12: The kernel sequence for the Adversarial dataset.

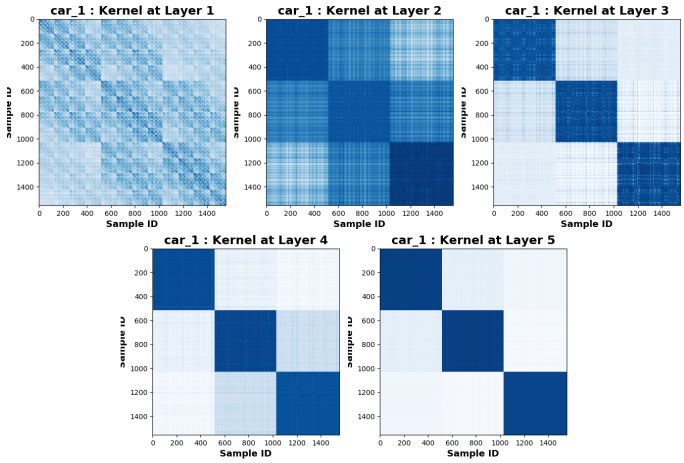

Figure 13: The kernel sequence for the car dataset.

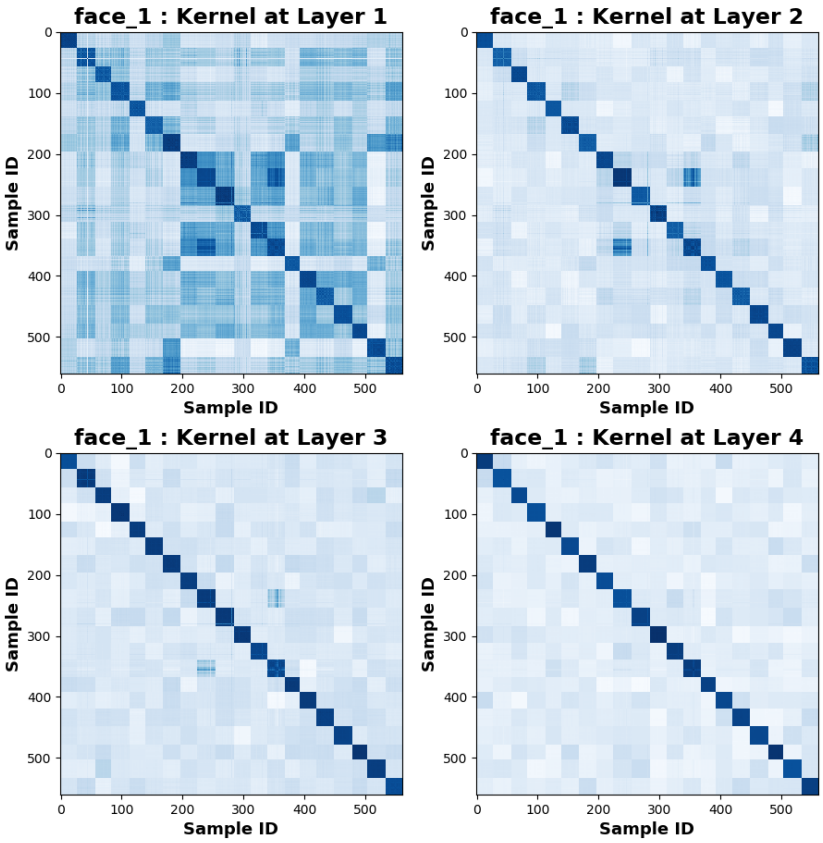

Figure 14: The kernel sequence for the face dataset.

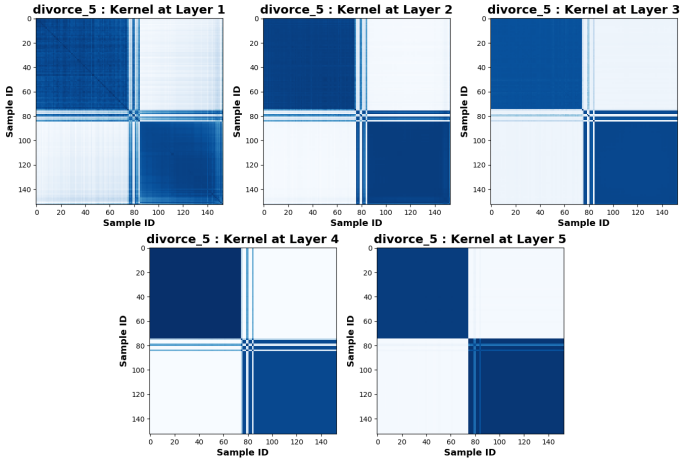

Figure 15: The kernel sequence for the divorce dataset.

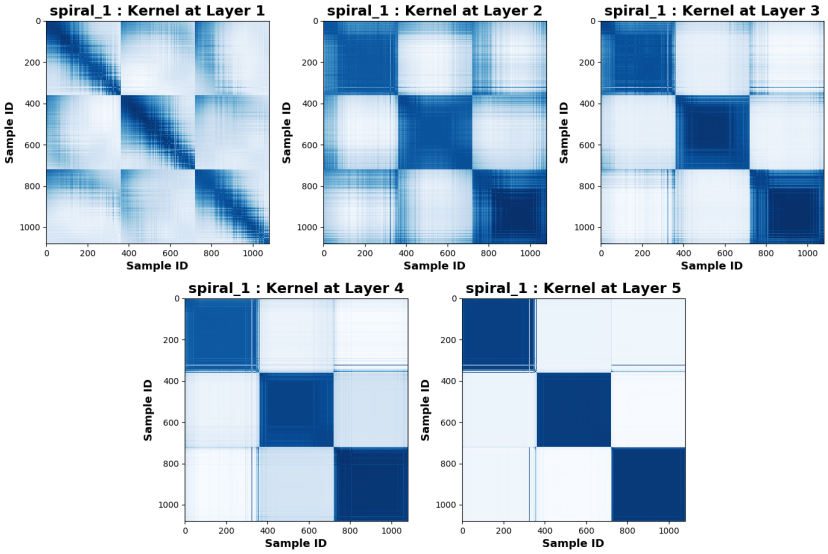

Figure 16: The kernel sequence for the spiral dataset.

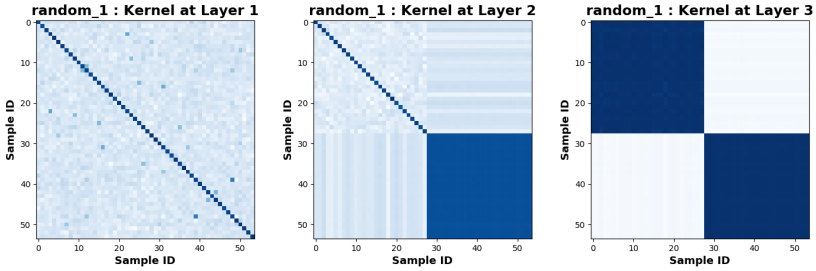

Figure 17: The kernel sequence for the Random dataset.

Appendix H Evaluation Metrics Graphs

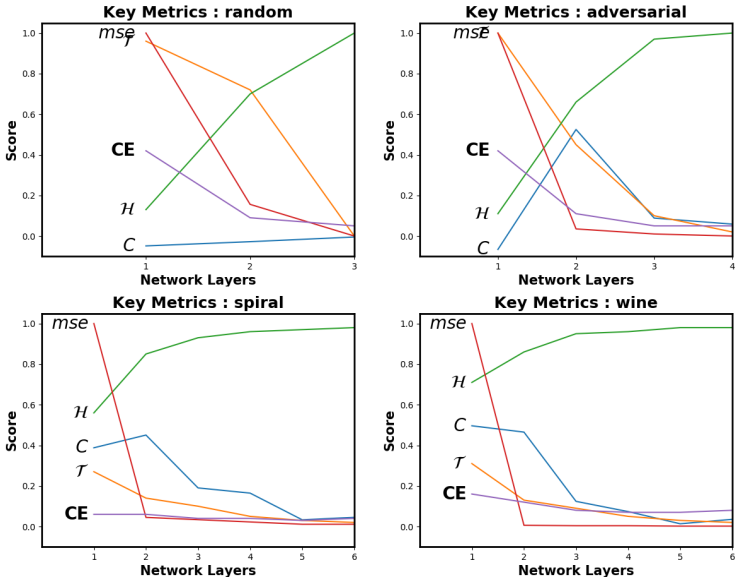

Figure 18

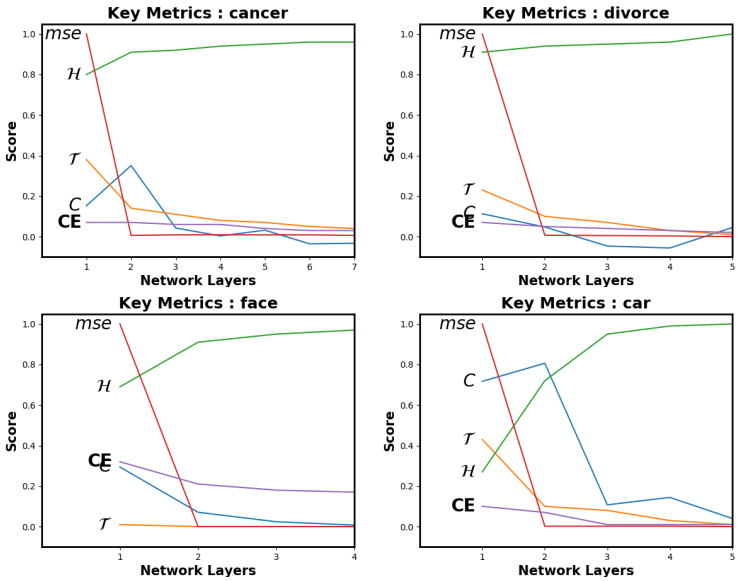

Figure 19: Figures of key metrics for all datasets as samples progress through the network. It is important to notice the uniformly and monotonically increasing $\mathcal{H}$-*Sequence* for each plot since this guarantees a converging kernel/risk sequence. As the $\mathcal{T}$ approach 0, samples of the same/difference classes in IDS are being pulled into a single point or pushed maximally apart respectively. As C approach 0, samples of the same/difference classes in RKHS are being pulled into 0 or $\frac{\pi}{2}$ cosine similarity respectively.

Appendix I Optimal Gaussian σ for Maximum Kernel Separation

Although the Gaussian kernel is the most common kernel choice for kernel methods, its σ value is a hyperparameter that must be tuned for each dataset. This work proposes to set the σ value based on the maximum kernel separation. The source code is made publicly available on `https://github.com/anonamous`.

Let $X \in \mathbb{R}^{n \times d}$ be a dataset of n samples with d features and let $Y \in \mathbb{R}^{n \times \tau}$ be the corresponding one-hot encoded labels where τ denotes the number of classes. Given $\kappa_X(\cdot, \cdot)$ and $\kappa_Y(\cdot, \cdot)$ as two kernel functions that applies respectively to X and Y to construct kernel matrices $K_X \in \mathbb{R}^{n \times n}$ and $K_Y \in \mathbb{R}^{n \times n}$. Given a set $\mathcal{S}$, we denote $|\mathcal{S}|$ as the number of elements within the set. Also let $\mathcal{S}$ and $\mathcal{S}^c$ be sets of all pairs of samples of (x_i, x_j) from a dataset X that belongs to the same and different classes respectively, then the average kernel value for all (x_i, x_j) pairs with the same class is

$$d_{\mathcal{S}} = \frac{1}{|\mathcal{S}|} \sum_{i,j \in \mathcal{S}} e^{-\frac{||x_i - x_j||^2}{2\sigma^2}} \tag{136}$$

and the average kernel value for all (x_i, x_j) pairs between different classes is

$$d_{\mathcal{S}^c} = \frac{1}{|\mathcal{S}^c|} \sum_{i,j \in \mathcal{S}^c} e^{-\frac{||x_i - x_j||^2}{2\sigma^2}}. \tag{137}$$

We propose to find the σ that maximizes the difference between $d_{\mathcal{S}}$ and $d_{\mathcal{S}^c}$ or

$$\max_{\sigma} \quad \frac{1}{|\mathcal{S}|} \sum_{i,j \in \mathcal{S}} e^{-\frac{||x_i - x_j||^2}{2\sigma^2}} - \frac{1}{|\mathcal{S}^c|} \sum_{i,j \in \mathcal{S}^c} e^{-\frac{||x_i - x_j||^2}{2\sigma^2}}. \tag{138}$$

It turns out that is expression can be computed efficiently. Let $g = \frac{1}{|\mathcal{S}|}$ and $\bar{g} = \frac{1}{|\mathcal{S}^c|}$, and let $\mathbf{1}_{n \times n} \in \mathbb{R}^{n \times n}$ be a matrix of 1s, then we can define Q as

$$Q = -gK_Y + \bar{g}(\mathbf{1}_{n \times n} - K_Y). \tag{139}$$

Or Q can be written more compactly as

$$Q = \bar{g}\mathbf{1}_{n \times n} - (g + \bar{g})K_Y. \tag{140}$$

Given Q, Eq. (138) becomes

$$\min_{\sigma} \quad \mathrm{Tr}(K_X Q). \tag{141}$$

This objective can be efficiently solved with BFGS.

Below in Fig. 20, we plot out the average within cluster kernel and the between cluster kernel values as we vary σ. From the plot, we can see that the maximum separation is discovered via BFGS.

Relation to HSIC. From Eq. (141), we can see that the σ that causes maximum kernel separation is directly related to HSIC. Given that the HSIC objective is normally written as

$$\min_{\sigma} \quad \mathrm{Tr}(K_X H K_Y H), \tag{142}$$

by setting $Q = HK_YH$, we can see how the two formulations are related. While the maximum kernel separation places the weight of each sample pair equally, HSIC weights the pair differently. We also notice that the $Q_{i,j}$ element is positive/negative for (x_i, x_j) pairs that are with/between classes respectively. Therefore, the argument for the global optimum should be relatively close for both objectives. Below in Figure 21, we show a figure of HSIC values as we vary σ. Notice how the optimal σ is almost equivalent to the solution from maximum kernel separation. For the purpose of *KChain*, we use σ that maximizes the HSIC value.

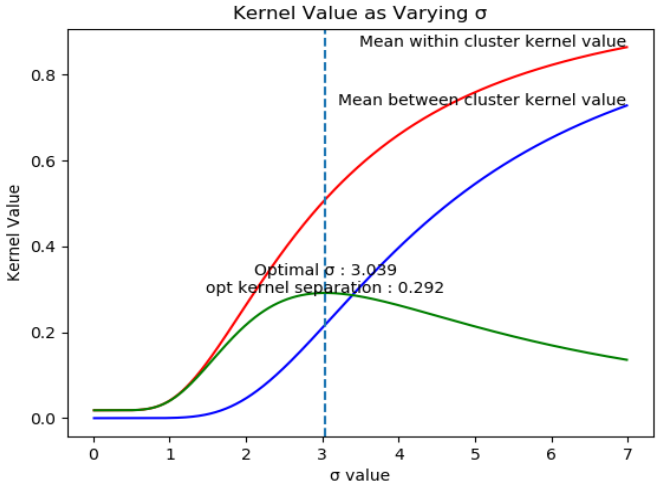

Figure 20: Maximum Kernel separation.

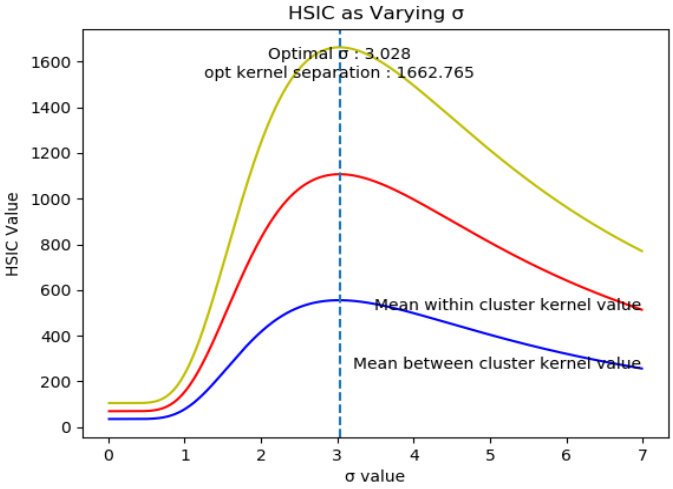

Figure 21: Maximal HSIC.