Kernel-KAN: Kernel Kolmogorov Arnold Networks

Ayad Alhusseiny

University of Kerbala

ayad.a@uokerbala.edu.iq

Abstract

The precise approximation of complex nonlinear functions poses a significant challenge in numerous scientific and engineering fields. Conventional neural network architectures, such as Multi-Layer Perceptrons (MLPs), frequently encounter difficulties in effectively capturing the intricate patterns and irregularities found in high-dimensional functions. This paper introduces the Kernel Kolmogorov-Arnold Network (Kernel-KAN), a novel neural network architecture inspired by the Kolmogorov-Arnold representation theorem, which integrates the robust approximation capabilities of kernel regression. By employing learnable functions that are parameterized by kernel interpolation formulas on the edges of the network, Kernel-KANs improve flexibility, efficiency, and interpretability in function approximation tasks. We showcase the effectiveness of Kernel-KANs through experiments involving digit classification, synthetic function approximation, and multivariate function approximation, demonstrating their superiority over traditional MLPs regarding parameter efficiency and interpretability.

Keywords: Kernel-KAN, Kernel Kolmogorov-Arnold Networks, Kolmogorov-Arnold Theorem.

Kernel-KAN: Kernel Kolmogorov Arnold Networks

Introduction

Recent developments in artificial intelligence (AI) have resulted in the emergence of highly capable AI systems that make decisions for reasons that remain unclear to us. This situation has sparked concerns regarding the extensive use of unreliable AI systems in both the economy and our everyday lives, introducing various new risks, including the possibility of future AIs misleading humans to achieve undesirable goals [2], [3]. The effort to decipher the black-box behavior of neural networks has garnered significant attention in recent years. The interpretability of neural networks is vital as it affects trust in these systems and aids in addressing ethical issues such as algorithmic discrimination. Furthermore, it is crucial for the application of neural networks in scientific domains like drug discovery and genomics, where comprehending the model's decisions is essential for validation and regulatory compliance [4], [5]. The multilayer feedforward perceptron (MLP) model is one of the most commonly utilized and effective neural network models [6]. Despite their widespread application, MLPs possess significant drawbacks, such as utilizing nearly all non-embedding parameters in transformers and exhibiting lower interpretability compared to attention layers [7], [1], [8].

Model renovation strategies are designed to improve interpretability by integrating more comprehensible elements into a network. These elements may consist of neurons with uniquely crafted activation functions, additional layers serving specific purposes, modular designs, and other similar characteristics [9].

Kolmogorov-Arnold Networks (KANs), which are based on the Kolmogorov-Arnold representation theorem [10], offer numerous benefits, including enhanced interpretability and

precision [1]. KANs feature a univariate learnable activation function on edges, while nodes aggregate these activation functions. The combination of KAN, ensembles of probabilistic trees, and multivariate B-spline representations is discussed in [11].

All the experiments and the implementation can be found in this GitHub repository: https://github.com/ayadalhusseiny/Kernel-KAN.

Kernel Regression

In the field of statistics, kernel regression serves as a non-parametric method for estimating the conditional expectation of a random variable. The aim is to identify a non-linear relationship between two random variables, **X** and **Y**.

In any nonparametric regression context, the conditional expectation of the variable **Y**, in relation to the variable **X**, can be expressed as:

$$E(Y \mid X) = m(X) \quad (1)$$

where m represents an unknown function [26].

Nadaraya–Watson kernel regression.

In 1964, Nadaraya and Watson suggested estimating m through a locally weighted average, employing a kernel as the weighting function.

Let the data be (y_i, X_i) where y_i is real-valued and X_i is a q-vector, and assume that all are continuously distributed with a joint density $f(y, x)$. Let $f(y|x)=f(y, x)/f(x)$ be the conditional density of y_i given X_i where $f(x) = \int f(y, x)dy$ is the marginal density of X_i. The regression function for y_i on X_i is

$$m(x) = E(y_i \mid X_i = x) \quad (2)$$

We aim to estimate this nonparametrically, making minimal assumptions regarding m. If we had a substantial number of observations where X_i precisely equals x, we could calculate the average value of the $y_i's$ for these observations. However, since X_i is continuously distributed, we will not encounter multiple observations that are equal to the same value.

The approach involves examining a neighborhood around x, and it is important to note that if X_i exhibits a positive density at x, we can expect to see a growing number of observations within this neighborhood as the sample size increases. If the regression function $m(x)$ is continuous, it is reasonable to assume that it remains relatively constant within this neighborhood (provided it is sufficiently small), allowing us to calculate the average of the y_i values corresponding to these observations. The key challenge lies in determining the appropriate size of the neighborhood to balance the variation in $m(x)$ across the neighborhood (estimation bias) with the quantity of observations present in that neighborhood.

We will examine a significant quantity of X_i within any specified neighborhood of x_i.

Consider the one-regressor scenario where q = 1.

Define a neighborhood of x as $x \pm h$ for a certain bandwidth $h > 0$. Consequently, a straightforward nonparametric estimator of $m(x)$ is the mean value of the $y_i's$ for the observations i for which X_i falls within this neighborhood, that is,

$$\hat{m}(x) = \frac{\sum_{i=1}^{n} 1(|X_i - x| \leq h) y_i}{\sum_{i=1}^{n} 1(|X_i - x| \leq h)} = \frac{\sum_{i=1}^{n} k(\frac{X_i - x}{h}) y_i}{\sum_{i=1}^{n} k(\frac{X_i - x}{h})} \quad (3)$$

where $K(u)$ is the uniform kernel with a bandwidth h.

This model offers the benefit of being non-parametric, which implies that it does not presuppose a distribution for the random variables and is not required to meet any assumptions. This characteristic enables the model to effectively capture the majority of relationships between variables, even in cases where those relationships are non-linear and no established function correlates them. Among the most commonly utilized kernels for density estimation is the Gaussian Kernel, which is defined as:

$$k(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2} \qquad (4)$$

which is utilized for every interval of size h. When employing Gaussian-like kernels, it has been demonstrated that the Scott rule for estimating the bandwidth yields exceptional results (Wasserman 2006) [23]. This bandwidth is computed as

$$h = 1.06\sigma(x)n^{-\frac{1}{5}} \qquad (5)$$

Where σ is population standard deviation (estimated by the sample standard deviation) and n is the sample size [24] [25]. This bandwidth optimizes the Integrated Mean Square Error (IMSE) and depends on the sample size and the variability of the data. The Scott bandwidth is usually used for normal symmetric and unimodal data.

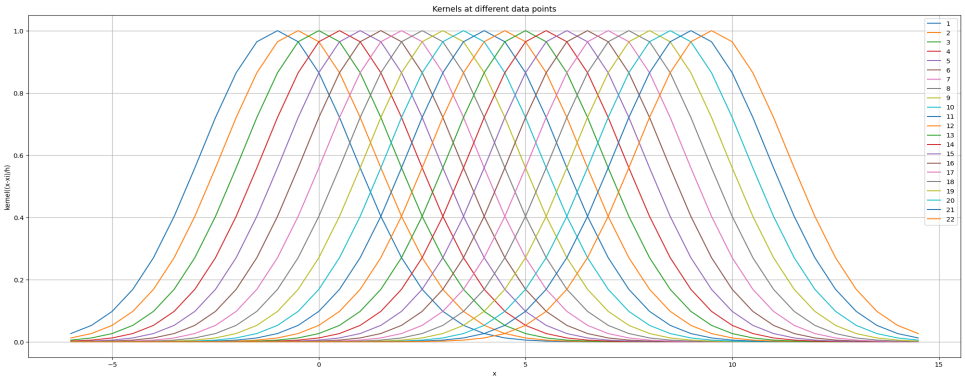

Figure 1: Example of a kernel function plotted for all xi.

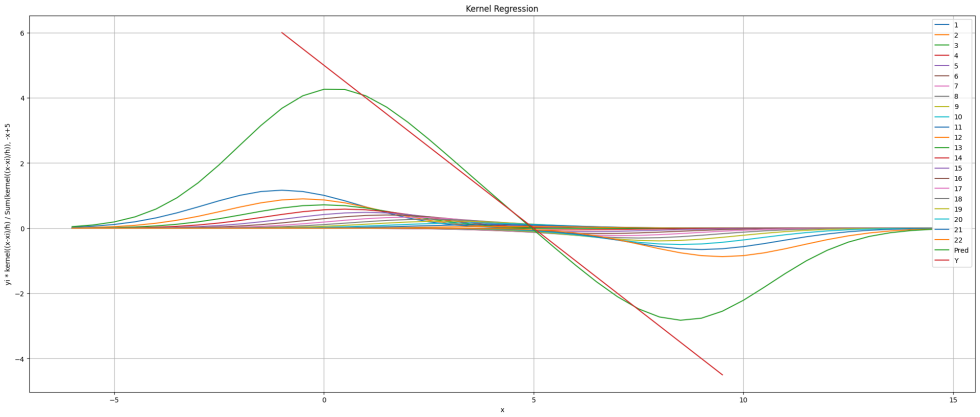

Figure 2: Example of function approximation using kernel regression.

Kolmogorov-Arnold Superposition Theorem

In the fields of real analysis and approximation theory, the Kolmogorov–Arnold representation theorem [10], also known as the superposition theorem, asserts that any multivariate continuous function f : $[0, 1]^d \rightarrow \mathbb{R}$ can be expressed as a superposition of continuous functions of a single variable.The relationship between the Kolmogorov-Arnold theorem (KAT) and neural networks is well-established in the literature [13, 14, 15, 16, 17, 18, 19, 20, 21, 22]. The contributions of Vladimir Arnold and Andrey Kolmogorov demonstrated that if f is a continuous multivariate function, it can be expressed as a finite composition of continuous functions of one variable along with the binary operation of addition. More specifically,

$$f(x) = f(x_1, \ldots, x_d) = \sum_{q=0}^{2d} \Psi_q \left(\sum_{p=1}^{d} \Psi_{q,p}(x_p) \right) \quad (6)$$

where $\Psi_{q,p} : [0, 1] \rightarrow \mathbb{R}$ and $\Psi_q : \mathbb{R} \rightarrow \mathbb{R}$. In essence, they demonstrated that the only genuine multivariate function is addition, as every other function can be expressed through univariate functions and their sum. One might optimistically view this as excellent news for machine

learning: learning a high-dimensional function essentially reduces to learning a polynomial number of one-dimensional functions. However, these one-dimensional functions can exhibit non-smooth and even fractal characteristics, rendering them potentially unlearnable in practice. Due to this pathological behavior, the Kolmogorov-Arnold representation theorem has effectively been deemed obsolete in the realm of machine learning, considered theoretically valid yet practically ineffective.

Nevertheless, we hold a more positive view regarding the applicability of the Kolmogorov-Arnold theorem in the context of machine learning. To begin with, we are not confined to the initial Eq. (6), which is limited to two-layer nonlinearities and a minimal number of terms (2d) within the hidden layer: we intend to extend the network to accommodate arbitrary widths and depths.

Kernel-KAN architecture

Assuming we are dealing with a supervised learning task that comprises input-output pairs $\{x_i, y_i\}$, our objective is to identify a function f such that $y_i \approx f(x_i)$ for every data point. Equation (6) suggests that our task is complete if we can determine suitable univariate functions Ψ_q and $\Psi_{q,p}$. This leads us to create a neural network that explicitly parameterized Equation (6). Given that all functions we aim to learn are univariate, we can represent each one-dimensional function as a kernel interpolation curve, utilizing learnable coefficients from kernel interpolation functions.

There exists a resemblance between MLPs and Kernel-KAN. In MLPs, once a layer is established, we can incorporate further layers to enhance the network's depth. It seems that a

Kernel-KAN layer featuring d_{in}-dimensional inputs and d_{out}-dimensional outputs can be described as a matrix composed of one-dimensional functions.

$$\psi = \{\psi_{q,p}\}, \quad p = 1, 2, \cdots, d_{in}, \quad q = 1, 2 \cdots, d_{out} \qquad (7)$$

where the functions $\psi_{q,p}$ possess trainable parameters, as elaborated below. In the KA theorem in Eq. (6), the initial functions constitute a Kernel-KAN layer with $d_{in} = d$ and $d_{out} = 2d$, while the subsequent functions create a deep Kernel-KAN layer with $d_{in} = 2d$ and $d_{out} = 1$. Thus, the Kernel-KAN representations in Eq. (6) are merely compositions of two Kernel-KAN layers.

The configuration of a Kernel-KAN is denoted by an integer array

$$[d_0, d_1, \cdots, d_L], \qquad (8)$$

where d_i represents the count of nodes in the i^{th} layer of the computational graph. We refer to the ith neuron in the lth layer as (l, i), and the activation value of the (l, i)-neuron is represented by $x_{l,i}$. Between layer l and layer l +1, there exist $d_l\, d_{l+1}$ activation functions: the activation function linking $(l,\ i)$ and $(l\ +\ 1,\ j)$ is denoted by

$$\psi_{l,j,i}, \quad l = 0, \cdots, L - 1, \quad i = 1, \cdots, d_l, \quad j = 1, \cdots, d_{l+1}. \qquad (9)$$

The pre-activation of $\psi_{l,j,i}$ is represented as $x_{l,i}$; the post-activation of $\psi_{l,j,i}$ is indicated by $\tilde{x}_{l,j,i} \equiv \psi_{l,j,i}(x_{l,i})$. The activation value of the (l + 1, j) neuron is merely the multiply of all incoming post-activations:

$$x_{l+1,j} = \prod_{i=1}^{d_l} \tilde{x}_{l,j,i} = \prod_{i=1}^{d_l} \psi_{l,j,i}\left(x_{l,i}\right), \quad j\ =\ 1, \ldots, d_{l+1} \qquad (10)$$

A deep Kernel-KAN consists of L layers: when provided with an input vector $x_0 \in R^{d0}$, the output of Kernel-KAN is

$$\text{Kernel-KAN}(X) = (\psi_{L-1} \circ \psi_{L-2} \circ \cdots \circ \psi_1 \circ \psi_0)x. \quad (11)$$

The aforementioned equation can also be reformulated to align more closely with Eq. (6), under the assumption that the output dimension $d_L = 1$, and define $f(x) \equiv \text{Kernel-KAN}(X)$:

$$f(x) = \sum_{i_{L-1}=1}^{d_{L-1}} \Psi_{L-1,i_L,i_{L-1}} \left(\sum_{i_{L-2}=1}^{d_{L-2}} \cdots \left(\sum_{i_2=1}^{d_2} \Psi_{2,i_3,i_2} \left(\sum_{i_1=1}^{d_1} \Psi_{1,i_2,i_1} \left(\sum_{i_0=1}^{d_0} \Psi_{0,i_1,i_0} \left(x_{i0} \right) \right) \right) \right) \cdots \right) \quad (12)$$

Implementation details.

In this part we need to optimize the Kernel-KAN layer described in Eq. (10).

1. The activation function $\psi(x)$ is the kernel regression of x

$$\psi(x) = Kernel\_Interpolation(x) \quad (13)$$

In the majority of instances, $Kernel\_Interpolation(x)$ is defined as

$$Kernel\_Interpolation(x) = \frac{\sum_{i=1}^{n} k(\frac{X_i - x}{h}) y_i}{\sum_{i=1}^{n} k(\frac{X_i - x}{h})} = \frac{\sum_{i=1}^{n} e^{-\frac{1}{2}(\frac{X_i - x}{h})^2} y_i}{\sum_{i=1}^{n} e^{-\frac{1}{2}(\frac{X_i - x}{h})^2}} \quad (14)$$

Where X_i and y_i are trainable parameters, represented by the grid number.

2. Initialization scales. The input x is normalized to the range $[-1, 1]$ using a hyperbolic tangent function

$$x = tanh(x) \quad (15)$$

So we can initialize X_i and y_i using random values drawn from a normal distribution, where the $mean$ is 0 and the $standard\ deviation$ is 1.

Experiments and Results

In this section, we provide an experimental assessment of the Kernel Kolmogorov-Arnold Network (Kernel-KAN) across multiple tasks, such as function approximation, generation of fractal functions and the digit classification using the MNIST dataset.

Function Approximation.

We created input-output pairs by uniformly sampling the input from a specified range and computing the corresponding output using the target function. We trained the Kernel-KAN model to minimize the mean squared error (MSE) between the predicted outputs and the actual outputs.

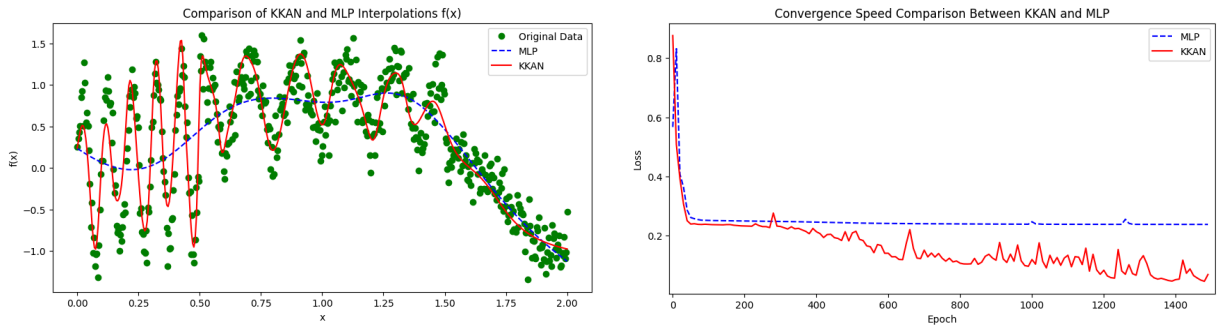

Figure 3: Function approximation example by using MLP and Kernel-KAN.

The Kernel-KAN model demonstrated superior approximation accuracy compared to traditional MLPs. The Kernel-KAN model achieved an MSE of 0.0676 while MLP 0.2371.

In the second part of this experiment, we examined how changing the number of grids affects the model's loss. The grid influences the complexity of the activation functions utilized in the Kernel-KAN layer. We conducted experiments with grids of sizes (5, 30, 100) and recorded the corresponding loss on the function.

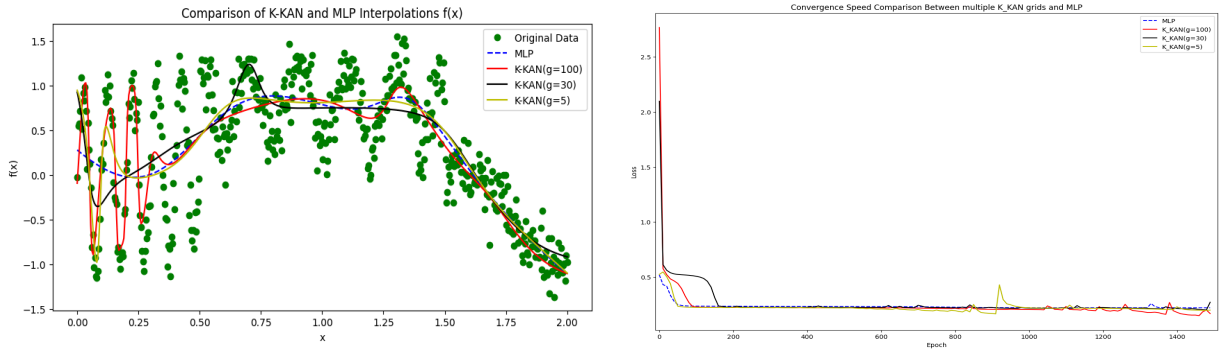

Figure 4: Function approximation example with different grid size by using MLP and Kernel-KAN.

The results are presented in Table 1. The performance of the model is greatly influenced by the selection of grid size.

Grid size	Loss
5	0.2147
30	0.2237
100	0.2204

Table 1: Degree grid loss

Test on Multivariable Function.

We generated input-output pairs by sampling the input uniformly from a specified range and computing the corresponding output using the multivariable target function. We trained the Kernel-KAN model to minimize the mean squared error (MSE) between the predicted and actual outputs.

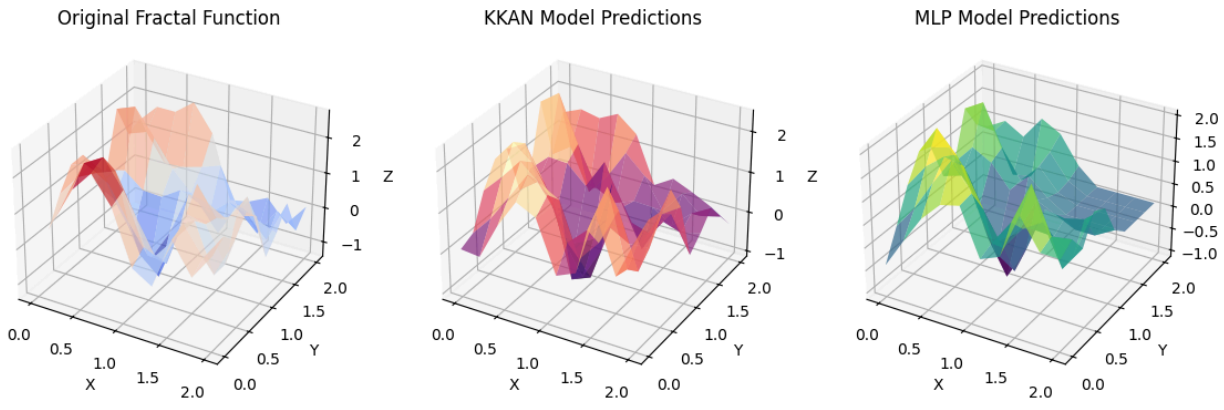

Figure 5: Multivariable function approximation example by using MLP and Kernel-KAN.

The Kernel-KAN model exhibited enhanced approximation accuracy in comparison to conventional MLPs. The Kernel-KAN model attained a mean squared error (MSE) of 0.1589, whereas the MLP recorded an MSE of 0.2221.

Digit Classification on MNIST.

The MNIST dataset serves as a standard benchmark for assessing image classification algorithms, comprising 60,000 training images and 10,000 test images of handwritten digits ranging from 0 to 9. Each image is presented as a 28x28 pixel grayscale image.

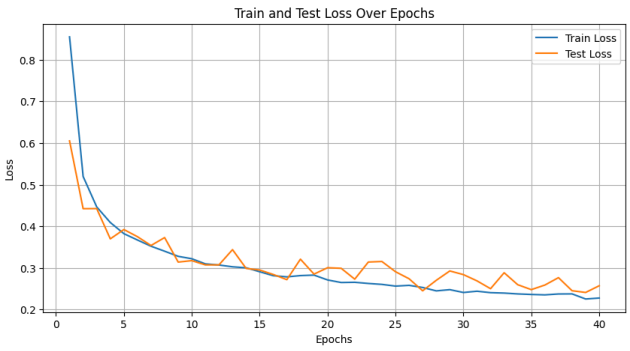

Figure 6: Train and test loss of the Digit Classification on MNIST dataset.

We implemented a Kernel-KAN model for digit classification, consisting of several layers of Kernel-KAN layers followed by fully connected layers. The input images were flattened and normalized to the range [−1, 1] using the hyperbolic tangent function.

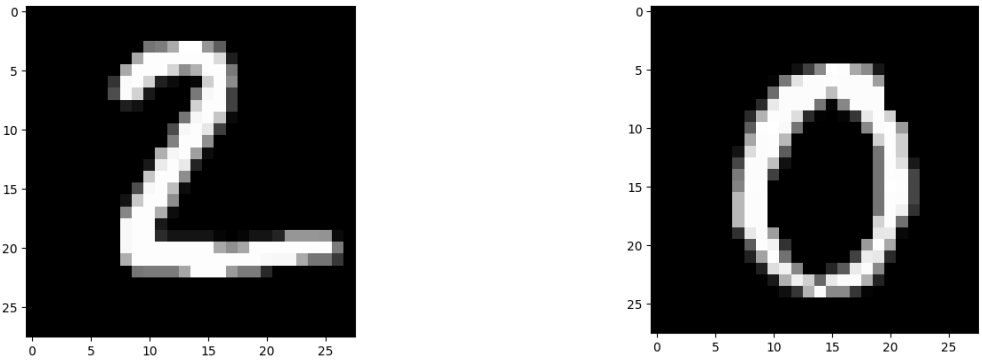

Figure 7: Some examples on MNIST dataset.

The use of a learnable coefficient in the Kernel-KAN significantly improved the classification performance by better capturing complex patterns in the digit images. The Kernel-KAN model achieved a test accuracy of 92%.

Conclusion

The Kernel Interpolation Kolmogorov-Arnold Network (Kernel-KAN) signifies a notable progress in nonlinear function approximation, providing a strong and effective substitute for conventional Multi-Layer Perceptrons (MLPs). By merging the Kolmogorov-Arnold theorem with the impressive approximation capabilities of kernel interpolation, Kernel-KANs attain enhanced performance regarding parameter efficiency, dynamic activation functions, and interpretability.

The experimental findings confirm the efficacy of Kernel-KANs across multiple tasks. Function approximation tasks further highlighted the model's ability to accurately capture complex

nonlinear relationships. In digit classification using the MNIST dataset, the Kernel-KAN model exhibited high accuracy while utilizing fewer parameters than conventional MLPs. Moreover, the capacity to approximate complex functions emphasizes the promise of Kernel-KANs in managing intricate, high-dimensional data.

In summary, the Kernel-KAN architecture significantly improves the flexibility and adaptability of neural networks while also offering a clear and interpretable framework for comprehending and modeling intricate functions. This groundbreaking method shows potential for various applications in scientific and engineering fields, setting the stage for further progress in function approximation and neural network development.

References

[1] Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljaciˇ c, T. Y. Hou, and M. Tegmark, “Kan: Kolmogorov-arnold networks,” ´ arXiv preprint arXiv:2404.19756, 2024.

[2] D. Hendrycks, M. Mazeika, and T. Woodside, “An overview of catastrophic ai risks,” arXiv preprint arXiv:2306.12001, 2023.

[3] R. Ngo, L. Chan, and S. Mindermann, “The alignment problem from a deep learning perspective,” arXiv preprint arXiv:2209.00626, 2022.

[4] Y. Zhang, P. Tino, A. Leonardis, and K. Tang, “A survey on neural network interpretability,” ˇ IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 5, no. 5, pp. 726–742, 2021.

[5] F. Doshi-Velez and B. Kim, “Towards a rigorous science of interpretable machine learning,” arXiv preprint arXiv:1702.08608, 2017.

[6] A. Pinkus, “Approximation theory of the mlp model in neural networks,” Acta numerica, vol. 8, pp. 143–195, 1999.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” Advances in neural information processing systems, vol. 30, 2017.

[8] H. Cunningham, A. Ewart, L. Riggs, R. Huben, and L. Sharkey, “Sparse autoencoders find highly interpretable features in language models,” arXiv preprint arXiv:2309.08600, 2023.

[9] F.-L. Fan, J. Xiong, M. Li, and G. Wang, “On interpretability of artificial neural networks: A survey,” IEEE Transactions on Radiation and Plasma Medical Sciences, vol. 5, no. 6, pp. 741–760, 2021.

[10] A. N. Kolmogorov, On the representation of continuous functions of several variables by superpositions of continuous functions of a smaller number of variables. American Mathematical Society, 1961.

[11] D. Fakhoury, E. Fakhoury, and H. Speleers, “Exsplinet: An interpretable and expressive spline-based neural network,” Neural Networks, vol. 152, pp. 332–346, 2022.

[12] Carl De Boor. A practical guide to splines, volume 27. springer-verlag New York, 1978.

[13] David A Sprecher and Sorin Draghici. Space-filling curves and kolmogorov superposition-based neural networks. Neural Networks, 15(1):57–67, 2002.

[14] Mario Köppen. On the training of a kolmogorov network. In Artificial Neural Networks—ICANN 2002: International Conference Madrid, Spain, August 28–30, 2002 Proceedings 12, pages 474–479. Springer, 2002.

[15] Ji-Nan Lin and Rolf Unbehauen. On the realization of a kolmogorov network. Neural Computation, 5(1):18–20, 1993.

[16] Ming-Jun Lai and Zhaiming Shen. The kolmogorov superposition theorem can break the curse of dimensionality when approximating high dimensional functions. arXiv preprint arXiv:2112.09963, 2021.

[17] Pierre-Emmanuel Leni, Yohan D Fougerolle, and Frédéric Truchetet. The kolmogorov spline network for image processing. In Image Processing: Concepts, Methodologies, Tools, and Applications, pages 54–78. IGI Global, 2013.

[18] Daniele Fakhoury, Emanuele Fakhoury, and Hendrik Speleers. Exsplinet: An interpretable and expressive spline-based neural network. Neural Networks, 152:332–346, 2022.

[19] Tomaso Poggio. How deep sparse networks avoid the curse of dimensionality: Efficiently computable functions are compositionally sparse. CBMM Memo, 10:2022, 2022.

[20] Johannes Schmidt-Hieber. The kolmogorov–arnold representation theorem revisited. Neural networks, 137:119–126, 2021.

[21] Aysu Ismayilova and Vugar E Ismailov. On the kolmogorov neural networks. Neural Networks, page 106333, 2024.

[22] Michael Poluektov and Andrew Polar. A new iterative method for construction of the kolmogorov-arnold representation. arXiv preprint arXiv:2305.08194, 2023.

[23] Chen., S. Optimal Bandwidth Selection for Kernel Density Functional Estimation. Journal of Probability and Statistics, 2015:1-21.

[24] Arai, Y. & Ichimura, H. Optimal Bandwidth Selection for Differences of Nonparametric Estimators with an Application to Sharp Regression Discontinuity Design. GRIPS discussion paper. National Graduate Institute for Policy Studies, Tokyo Japan. 2013.

[25] Hansen, B.E. Bandwidth Selection for Nonparametric Distribution Estimation. Research supported by the National Science Foundation. Department of Economics University of Wisconsin, Madison. 2004.

[26] Silverman, B.W. Density Estimation for Statistics and Data Analysis. Chapman and Hall, London. 1986.