Survey of KAN

Jinheung Kim

Kan: Kolmogorov-arnold networks

[PDF] arxiv.org

Z Liu, Y Wang, S Vaidya, F Ruehle, J Halverson, M Soljačić, TY Hou, M Tegmark

arXiv preprint arXiv:2404.19756, 2024 • arxiv.org

> Inspired by the Kolmogorov-Arnold representation theorem, we propose Kolmogorov-Arnold Networks (KANs) as promising alternatives to Multi-Layer Perceptrons (MLPs). While MLPs have fixed activation functions on nodes ("neurons"), KANs have learnable activation functions on edges ("weights"). KANs have no linear weights at all -- every weight parameter is replaced by a univariate function parametrized as a spline. We show that this seemingly simple change makes KANs outperform MLPs in terms of accuracy

SHOW MORE

☆ Save Cite Cited by 58 Related articles All 3 versions »

2024.04.30~2024.07.013

KAN

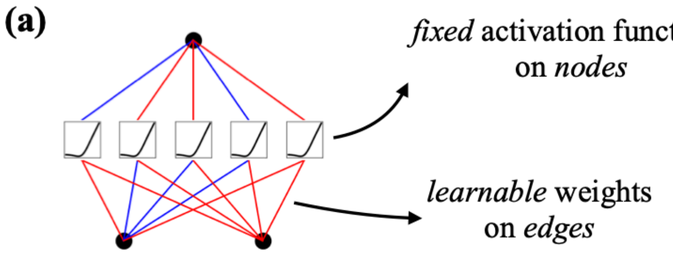
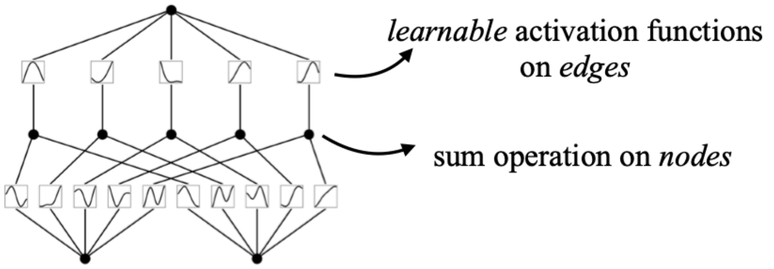
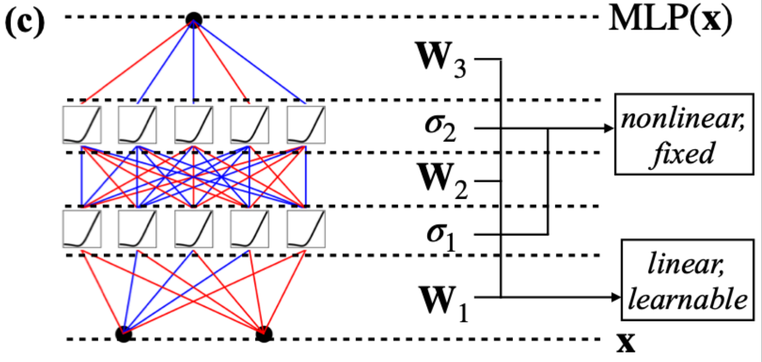
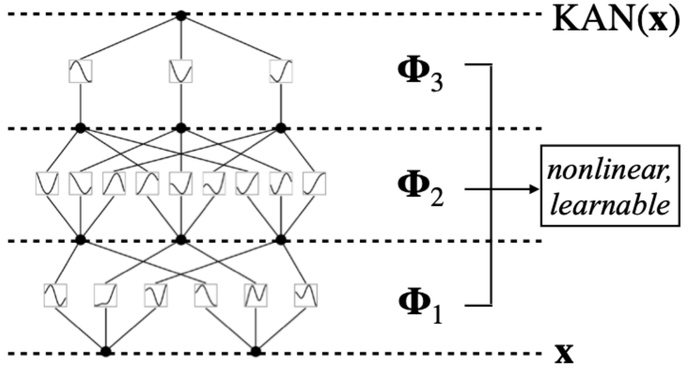
Model	Multi-Layer Perceptron (MLP)	Kolmogorov-Arnold Network (KAN)
Theorem	**Universal Approximation Theorem**	**Kolmogorov-Arnold Representation Theorem**
Formula (Shallow)	$f(\mathbf{x}) \approx \sum_{i=1}^{N(\epsilon)} a_i \sigma(\mathbf{w}_i \cdot \mathbf{x} + b_i)$	$f(\mathbf{x}) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right)$
Model (Shallow)		
Formula (Deep)	$\mathrm{MLP}(\mathbf{x}) = (\mathbf{W}_3 \circ \sigma_2 \circ \mathbf{W}_2 \circ \sigma_1 \circ \mathbf{W}_1)(\mathbf{x})$	$\mathrm{KAN}(\mathbf{x}) = (\mathbf{\Phi}_3 \circ \mathbf{\Phi}_2 \circ \mathbf{\Phi}_1)(\mathbf{x})$
Model (Deep)		

Figure 0.1: Multi-Layer Perceptrons (MLPs) vs. Kolmogorov-Arnold Networks (KANs)

[PDF] KAN: Kolmogorov–Arnold Networks: A review [PDF] vikasdhiman.info

V Dhiman - 2024 - vikasdhiman.info

61 days ago - Why review this? On Apr 30, 2024,[Liu et al., 2024] appears on ArXiV and by May 7th, I have heard about this paper from multiple students, from whom I do not hear …

☆ Save 99 Cite Cited by 1 Related articles »

Optimizing Hand Region Detection in MediaPipe Holistic Full-Body Pose Estimation to Improve Accuracy and Avoid Downstream Errors [PDF] arxiv.org

A Moryossef - arXiv preprint arXiv:2405.03545, 2024 - arxiv.org

62 days ago - This paper addresses a critical flaw in MediaPipe Holistic's hand Region of Interest (ROI) prediction, which struggles with non-ideal hand orientations, affecting sign …

☆ Save 99 Cite Cited by 2 Related articles All 2 versions »

SeNMo: A self-normalizing deep learning model for enhanced multi-omics data analysis in oncology [PDF] arxiv.org

A Waqas, A Tripathi, S Ahmed, A Mukund… - arXiv preprint arXiv …, 2024 - arxiv.org

107 days ago - Multi-omics research has enhanced our understanding of cancer heterogeneity and progression. Investigating molecular data through multi-omics …

☆ Save 99 Cite Related articles All 3 versions »

LCA-on-the-Line: Benchmarking Out of Distribution Generalization with Class Taxonomies [PDF] openreview.net

J Shi, GR Gare, J Tian, S Chai, Z Lin… - Forty-first International … - openreview.net

258 days ago - We tackle the challenge of predicting models' Out-of-Distribution (OOD) performance using in-distribution (ID) measurements without requiring OOD data. Existing …

☆ Save 99 Cite Related articles All 2 versions »

Biology-inspired joint distribution neurons based on Hierarchical Correlation Reconstruction allowing for multidirectional neural networks

[PDF] arxiv.org

J Duda - arXiv preprint arXiv:2405.05097, 2024 - arxiv.org

60 days ago - Popular artificial neural networks (ANN) optimize parameters for unidirectional value propagation, assuming some guessed parametrization type like Multi-Layer …

☆ Save 99 Cite Related articles All 3 versions »

- Introduce the HCR
- Compair MLP and KAN.

ISR: Invertible Symbolic Regression

[PDF] arxiv.org

T Tohme, MJ Khojasteh, M Sadr, F Meyer… - arXiv preprint arXiv …, 2024 - arxiv.org

59 days ago - We introduce an Invertible Symbolic Regression (ISR) method. It is a machine learning technique that generates analytical relationships between inputs and outputs of a …

☆ Save 99 Cite Related articles All 2 versions »

FastKAN

Kolmogorov-arnold networks are radial basis function networks [PDF] arxiv.org

Z Li - arXiv preprint arXiv:2405.06721, 2024 - arxiv.org

59 days ago - This short paper is a fast proof-of-concept that the 3-order B-splines used in Kolmogorov-Arnold Networks (KANs) can be well approximated by Gaussian radial basis …

☆ Save 99 Cite Cited by 6 Related articles All 2 versions »

This short paper is a fast proof-of-concept that the 3-order B-splines used in Kolmogorov-Arnold Networks (KANs) can be well approximated by Gaussian radial basis functions. Doing so leads to FastKAN, a much faster implementation of KAN which is also a radial basis function (RBF) network. Code available at `github.com/ZiyaoLi/fast-kan` .

$$f(x) = \sum_{i=1}^{N} w_i \phi(|x - c_i|)$$

$$\phi(r) = \exp\left(-\frac{r^2}{2h^2}\right)$$

Implementation	Fwd. (μs)	Fwd. acc.	Fwd. + Bwd. (μs)	Fwd. + Bwd. acc.
`efficient_kan`	742±186	1.00	1160±18.8	1.00
FastKAN	223±19	3.33	925±13.6	1.25

Table 1: Forward and backward time.

MNIST

TKAN

Tkan: Temporal kolmogorov-arnold networks [PDF] arxiv.org

R Genet, H Inzirillo - arXiv preprint arXiv:2405.07344, 2024 - arxiv.org

57 days ago - Recurrent Neural Networks (RNNs) have revolutionized many areas of machine learning, particularly in natural language and data sequence processing. Long …

☆ Save 99 Cite Cited by 10 Related articles All 3 versions »

***Abstract*—Recurrent Neural Networks (RNNs) have revolutionized many areas of machine learning, particularly in natural language and data sequence processing. Long Short-Term Memory (LSTM) has demonstrated its ability to capture long-term dependencies in sequential data. Inspired by the Kolmogorov-Arnold Networks (KANs) a promising alternatives to Multi-Layer Perceptrons (MLPs), we proposed a new neural networks architecture inspired by KAN and the LSTM, the Temporal Kolomogorov-Arnold Networks (TKANs). TKANs combined the strenght of both networks, it is composed of Recurring Kolmogorov-Arnold Networks (RKANs) Layers embedding memory management. This innovation enables us to perform multi-step time series forecasting with enhanced accuracy and efficiency. By addressing the limitations of traditional models in handling complex sequential patterns, the TKAN architecture offers significant potential for advancements in fields requiring more than one step ahead forecasting.**

TKAN

Tkan: Temporal kolmogorov-arnold networks [PDF] arxiv.org

R Genet, H Inzirillo - arXiv preprint arXiv:2405.07344, 2024 - arxiv.org

57 days ago - Recurrent Neural Networks (RNNs) have revolutionized many areas of machine learning, particularly in natural language and data sequence processing. Long …

☆ Save 99 Cite Cited by 10 Related articles All 3 versions »

RKAN (Recurrent KAN)

$$h_t = f(W_{hh}h_{t-1} + W_{hx}x_t + b_h)$$

$$x_{l+1,j}(t) = \sum_{i=1}^{n_l} \tilde{x}_{l,j,i}(t) = \sum_{i=1}^{n_l} \phi_{l,j,i,t}(x_{l,i}(t), h_{l,i}(t))$$

$$h_{l,i}(t) = W_{hh}h_{l,i}(t-1) + W_{hz}x_{l,i}(t)$$

Node i in l th layer $\quad j = 1, \ldots, n_{l+1}$

$$\mathrm{KAN}(x,t) = (\Phi_{L-1,t} \circ \Phi_{L-2,t} \circ \ldots \Phi_{1,t} \circ \Phi_{0,t})(x,t)$$

TKAN

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$ Forget gate

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$ Input gate

$$o_t = \sigma(\text{KAN}(\vec{x}, t))$$ Output gate

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$ Cell state

$$\tilde{c}_t = \sigma(W_c x_t + U_c h_{t-1} + b_c)$$

$$h_t = o_t \odot \tanh(c_t)$$ Output

Our dataset consists of the notional amounts traded each hour on several assets: BTC, ETH, ADA, XMR, EOS, MATIC, TRX, FTM, BNB, XLM, ENJ, CHZ, BUSD, ATOM, LINK, ETC, XRP, BCH and LTC, which are to be used to predict just one of them, BTC. The data period runs from January 1, 2020 to December 31, 2022.

$$R^2 = 1 - \frac{\sum_i^N (\hat{X}_{t+1}^{(i)} - X_{t+1}^{(i)})^2}{\sum_i^N (X_{t+1}^{(i)} - \bar{X}_{t+1}^{(i)})^2}$$

TABLE I: Average (R^2) obtained over 5 run

Time	TKAN:5 B-Spline	GRU:default	LSTM:default	Last Value
1	0.350845	0.365136	0.355532	0.292171
3	0.198842	0.200674	0.061220	-0.062813
6	0.140543	0.082504	-0.225838	-0.331346
9	0.117477	0.087164	-0.290584	-0.457718
12	0.105111	0.017864	-0.473220	-0.518252
15	0.086077	0.033423	-0.404432	-0.555633

TABLE II: Standard Deviation of the (R^2) obtained over 5 run

Time	TKAN:5 B-Spline	GRU:default	LSTM:default
1	0.014208	0.008336	0.011163
3	0.007385	0.004848	0.080200
6	0.007238	0.023637	0.062710
9	0.014551	0.014833	0.052729
12	0.003919	0.086386	0.085746
15	0.024656	0.024078	0.092729

Predictive Modeling of Flexible EHD Pumps using Kolmogorov-Arnold Networks

[PDF] arxiv.org

Y Peng, M He, F Hu, Z Mao, X Huang, J Ding - arXiv preprint arXiv …, 2024 - arxiv.org

55 days ago - We present a novel approach to predicting the pressure and flow rate of flexible electrohydrodynamic pumps using the Kolmogorov-Arnold Network. Inspired by the ..

☆ Save 99 Cite Cited by 2 Related articles All 3 versions »

TABLE I
MSE FOR PRESSURE AND FLOW RATE PREDICTIONS BY DIFFERENT MODELS.

Model	Pressure MSE	Flow Rate MSE
KAN	**12.186**	0.012
Random Forest	1750.017	0.040
MLP	78.329	**0.002**

Kolmogorov-arnold networks (kans) for time series analysis

CJ Vaca-Rubio, L Blanco, R Pereira… - arXiv preprint arXiv …, 2024 - arxiv.org

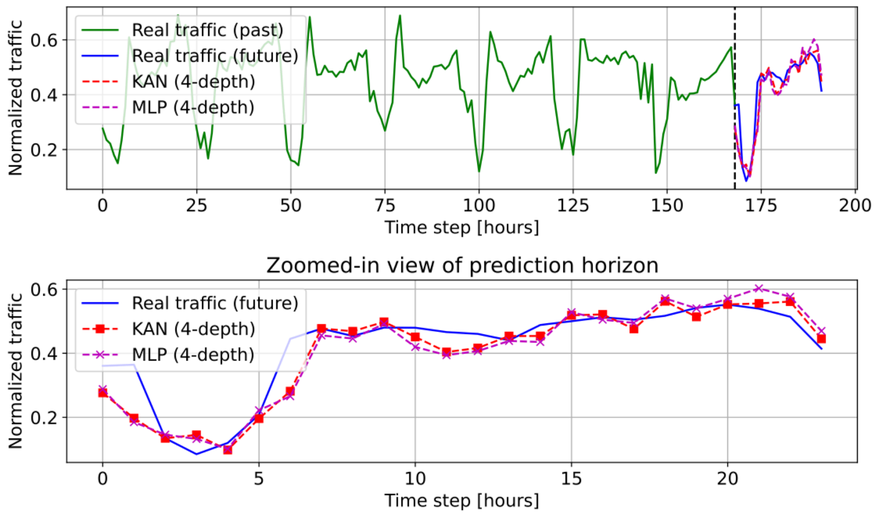
54 days ago - This paper introduces a novel application of Kolmogorov-Arnold Ne (KANs) to time series forecasting, leveraging their adaptive activation functions fo

☆ Save 99 Cite Cited by 10 Related articles All 2 versions »

Table 2: Results summary

Model	MSE ($\times 10^{-3}$)	RMSE ($\times 10^{-2}$)	MAE ($\times 10^{-2}$)	MAPE	Parameters
MLP (3-depth)	6.34	7.96	5.41	0.64	238k
MLP (4-depth)	6.12	7.82	5.55	1.05	329k
KAN (3-depth)	5.99	7.73	5.51	0.62	**93k**
KAN (4-depth)	**5.08**	**7.12**	**5.06**	**0.52**	109k

Satellite traffic forecasting



(a) Forecast over beam 1.

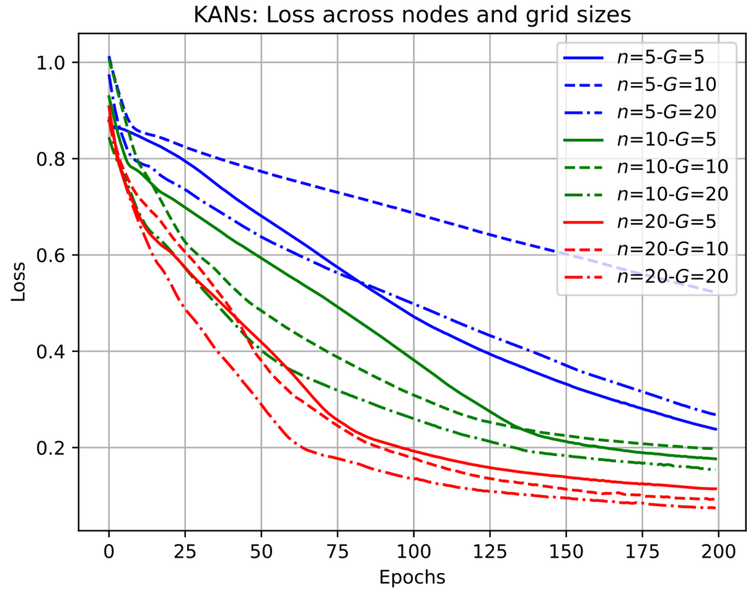


Fig. 4: Ablation comparison of KAN-specific parameters.

Smooth Kolmogorov Arnold networks enabling structural knowledge representation

[PDF] arxiv.org

ME Samadi, Y Müller, A Schuppert - arXiv preprint arXiv:2405.11318, 2024 - arxiv.org

51 days ago - Kolmogorov-Arnold Networks (KANs) offer an efficient and interpretable alternative to traditional multi-layer perceptron (MLP) architectures due to their finite network …

☆ Save 99 Cite Cited by 5 Related articles All 2 versions »

WAV-KAN

Wav-kan: Wavelet kolmogorov-arnold networks

[PDF] arxiv.org

Z Bozorgasl, H Chen - arXiv preprint arXiv:2405.12832, 2024 - arxiv.org

47 days ago - In this paper, we introduce Wav-KAN, an innovative neural network architecture that leverages the Wavelet Kolmogorov-Arnold Networks (Wav-KAN) framework …

☆ Save 99 Cite Cited by 12 Related articles All 3 versions »

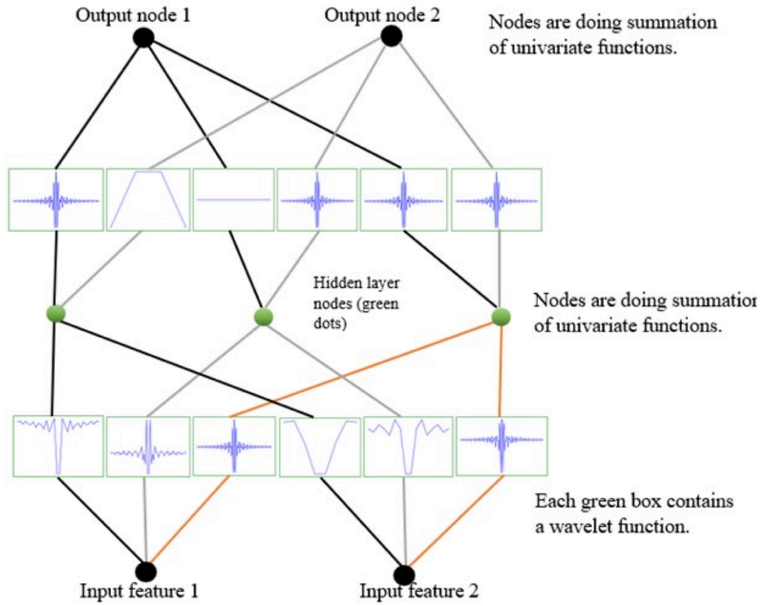


Fig. 1: Wav-KAN with arbitrary number of layers (here is Wav-KAN[2,3,2])

WAV-KAN

Wav-kan: Wavelet kolmogorov-arnold networks [PDF] arxiv.org

Z Bozorgasl, H Chen - arXiv preprint arXiv:2405.12832, 2024 - arxiv.org

47 days ago - In this paper, we introduce Wav-KAN, an innovative neural network architecture that leverages the Wavelet Kolmogorov-Arnold Networks (Wav-KAN) framework …

☆ Save 99 Cite Cited by 12 Related articles All 3 versions »

B-Spline → Wavelet

$\psi \in L^2(\mathbb{R})$: mother wavelet

Zero Mean: $\int_{-\infty}^{\infty} \psi(t)dt = 0$

Admissibility Condition: $C_\psi = \int_0^{\infty} \frac{|\psi(w)|^2 dw}{w} < \infty$

$$C(s,\tau) = \int_{-\infty}^{\infty} g(t) \frac{1}{\sqrt{s}} \psi\left(\frac{t-\tau}{s} dt\right)$$

$$g(t) = \frac{1}{C_\psi} \int_{-\infty}^{\infty} \int_0^{\infty} C(s,\tau) \frac{1}{\sqrt{s}} \psi\left(\frac{t-\tau}{s}\right) \frac{ds d\tau}{s^2}$$

s: Scale factor

τ: shift factor

WAV-KAN

Wav-kan: Wavelet kolmogorov-arnold networks [PDF] arxiv.org

Z Bozorgasl, H Chen - arXiv preprint arXiv:2405.12832, 2024 - arxiv.org

47 days ago - In this paper, we introduce Wav-KAN, an innovative neural network architecture that leverages the Wavelet Kolmogorov-Arnold Networks (Wav-KAN) framework …

☆ Save 99 Cite Cited by 12 Related articles All 3 versions »

$\phi \in L^2(\mathbb{R})$: scaling function

$$a_j(k) = \sum_n g(n)\phi_{j,k}(n)$$

$$d_j(k) = \sum_n g(n)\psi_{j,k}(n)$$

$$g(n) = \sum_k a_j(k)\phi_{j,k}(n) + \sum_{j=1}^{J}\sum_k d_j(k)\psi_{j,k}(n)$$

WAV-KAN

respect to MLPs exist for Wav-KAN. More importantly Wav-KAN has solved the major disadvantage of Spl-KANs which was slow training speed. In terms of number of parameter, we compared Wav-KAN with Spl-KAN and MLPs for a hypothetical neural network which has N input nodes and N output node, with L layers. As we see in Table I and by considering the value of G, Wav-KAN has less number of

TABLE I: Comparison of MLPs, Spl-KAN and Wav-KAN

Neural network With L layers, each layer has N nodes		
Neural Network Structure	Order	Parameters
MLPs	$O(N^2L)$ or $O(N^2L + NL)$	weights and biases
Spl-KAN	$O(N^2L(G + k + 1)) \sim O(N^2LG)$	weights
Wav-KAN	$O(3N^2L)$	weight, translation, scaling

Moreover, Spl-KAN heavily depends on the grid spaces, and for better performance, it requires increasing the number of grids [10], though, it brings two disadvantages. First it needs curvefitting which is a cumbersome and computational expensive operation, and also while we increase the number of grids, loss has some jumps [11]. Fortunately, wavelet is safe from such computations and deficiencies and if one wants to capture more details, DWT efficiently does that without recalculation of the previous steps.

RoPINN: Region Optimized Physics-Informed Neural Networks [PDF] arxiv.org

H Wu, H Luo, Y Ma, J Wang, M Long - arXiv preprint arXiv:2405.14369, 2024 - arxiv.org

45 days ago - Physics-informed neural networks (PINNs) have been widely applied to solve partial differential equations (PDEs) by enforcing outputs and gradients of deep models to …

☆ Save 99 Cite Related articles All 2 versions »

Endowing Interpretability for Neural Cognitive Diagnosis by Efficient Kolmogorov-Arnold Networks [PDF] arxiv.org

S Yang, L Qin, X Yu - arXiv preprint arXiv:2405.14399, 2024 - arxiv.org

46 days ago - In the realm of intelligent education, cognitive diagnosis plays a crucial role in subsequent recommendation tasks attributed to the revealed students' proficiency in …

☆ Save 99 Cite Cited by 2 Related articles All 2 versions »

Table 2: Performance comparison on four datasets. The best, second-best, and third-best results are in bold, underlined, and underwaved: KA2NCD-native's results are not involved.

Dataset	Assistments		SLP		JunYi		FrcSub	
Method	AUC↑	ACC↑	AUC↑	ACC↑	AUC↑	ACC↑	AUC↑	ACC↑
IRT	72.02%	70.25%	80.91%	74.29%	74.80%	72.74%	80.63%	57.14%
MIRT	65.84%	63.90%	72.78%	71.90%	69.59%	69.50%	81.93%	69.12%
DINA	72.15%	68.06%	77.24%	71.43%	75.81%	68.18%	80.66%	83.12%
MF	70.55%	68.26%	79.22%	72.80%	79.48%	74.15%	84.10%	84.10%
NCD	74.84%	72.15%	84.76%	80.72%	80.70%	76.73%	90.12%	70.15%
NCD+	75.71%	71.91%	84.72%	80.38%	80.38%	76.12%	90.66%	74.30%
KaNCD	76.44%	**73.33%**	85.21%	81.61%	80.80%	76.15%	90.11%	76.68%
KaNCD+	76.99%	73.54%	85.25%	81.91%	82.06%	77.23%	91.44%	78.36%
RCD	75.91%	72.99%	85.57%	79.37%	**83.25%**	**78.67%**	89.39%	73.83%
RCD+	77.10%	73.78%	86.38%	80.12%	83.33%	78.76%	89.46%	74.53%
KSCD	76.55%	73.04%	85.90%	81.02%	82.17%	77.83%	90.49%	80.27%
KSCD+	76.72%	73.01%	86.06%	80.87%	83.43%	78.67%	90.66%	82.93%
KA2NCD-e	**76.64%**	72.96%	**86.08%**	**82.66%**	83.18%	78.39%	**91.27%**	**84.58%**
KA2NCD-kan	72.89%	71.55%	85.91%	81.91%	83.14%	78.41%	90.38%	83.30%

DeepOKAN

Deepokan: Deep operator network based on kolmogorov arnold networks for mechanics problems [PDF] arxiv.org

DW Abueidda, P Pantidis, ME Mobasher - arXiv preprint arXiv:2405.19143, 2024 - arxiv.org

40 days ago - The modern digital engineering design often requires costly repeated simulations for different scenarios. The prediction capability of neural networks (NNs) makes …

☆ Save 99 Cite Cited by 7 Related articles All 2 versions »

Gaussian radial basis functions

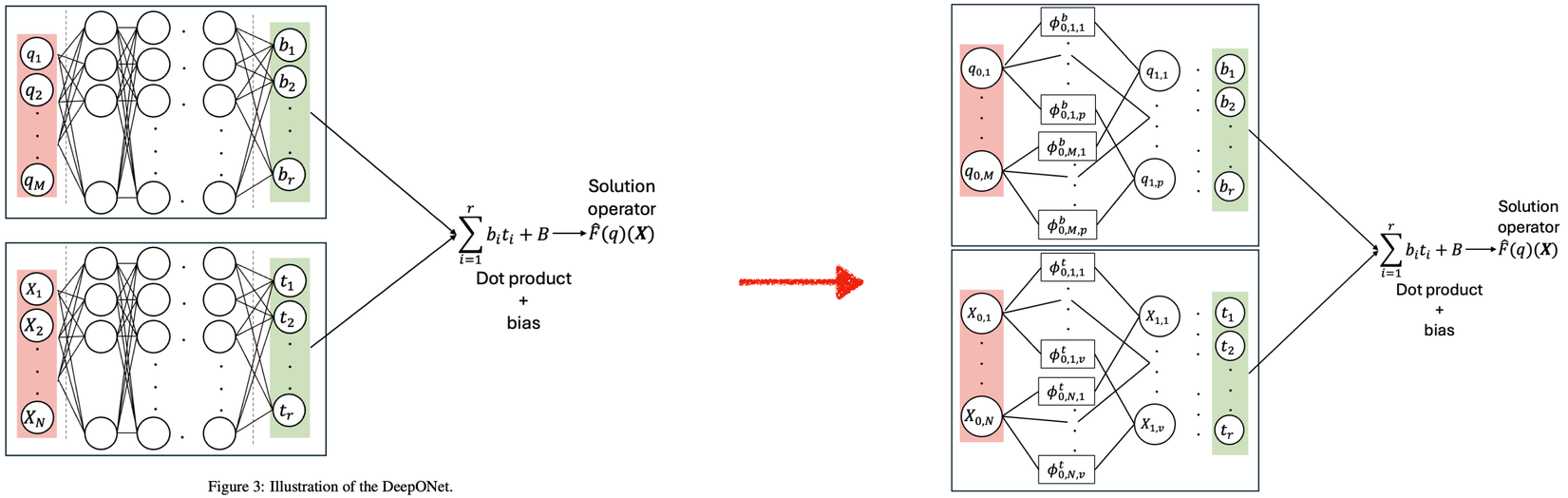


Figure 3: Illustration of the DeepONet.

DeepOKAN

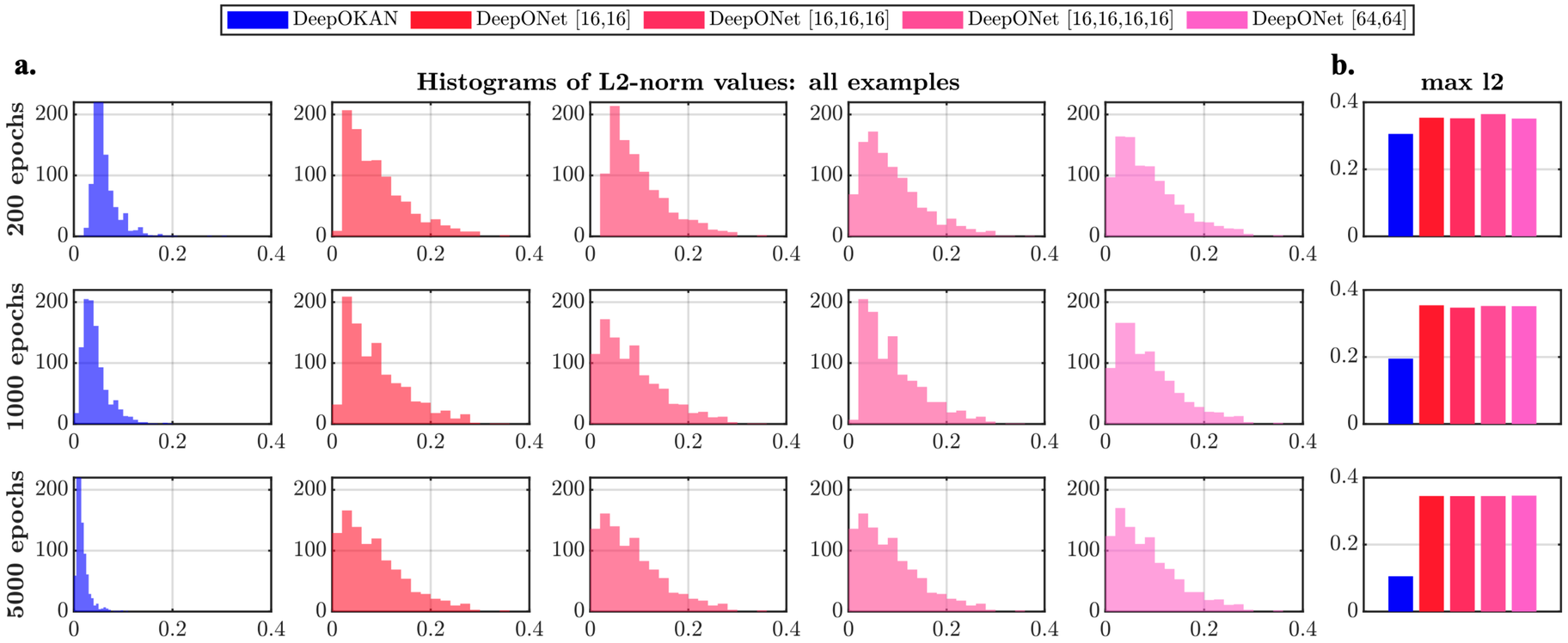


Figure 7: **a.** Histogram of the L2-norm for all the test examples. **b.** Maximum L2-norm value for each case of training epochs.

Avoiding Catastrophic Forgetting Via Neuronal Decay

RO Malashin, MA Mikhalkova - 2024 Wave Electronics and its …, 2024 - ieeexplore.ieee.org

36 days ago - In continual learning settings neural network is taught different tasks sequentially and the network is prone to catastrophic forgetting. We investigate the role of …

☆ Save Cite Related articles

Kolmogorov-Arnold Network for Satellite Image Classification in Remote Sensing

[PDF] arxiv.org

M Cheon - arXiv preprint arXiv:2406.00600, 2024 - arxiv.org

37 days ago - In this research, we propose the first approach for integrating the Kolmogorov-Arnold Network (KAN) with various pre-trained Convolutional Neural Network (CNN) models …

☆ Save Cite Cited by 7 Related articles All 3 versions »

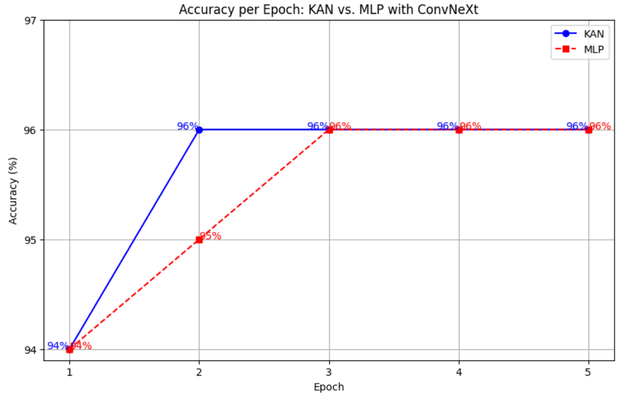


Figure 5: Accuracy per Epoch: KAN vs. MLP with ConvNeXt

Brain-Inspired Learning, Perception, and Cognition: A Comprehensive Review

L Jiao, M Ma, P He, X Geng, X Liu, F Liu… - … on Neural Networks …, 2024 - ieeexplore.ieee.org

40 days ago - The progress of brain cognition and learning mechanisms has provided new inspiration for the next generation of artificial intelligence (AI) and provided the biological …

☆ Save Cite Related articles All 3 versions

FourierKAN-GCF

FourierKAN-GCF: Fourier Kolmogorov-Arnold Network--An Effective and Efficient Feature Transformation for Graph Collaborative Filtering [PDF] arxiv.org

J Xu, Z Chen, J Li, S Yang, W Wang, X Hu… - arXiv preprint arXiv …, 2024 - a

35 days ago - Graph Collaborative Filtering (GCF) has achieved state-of-the-a for recommendation tasks. However, most GCF structures simplify the feature

☆ Save 99 Cite Cited by 6 Related articles All 2 versions »

ABSTRACT

Graph Collaborative Filtering (GCF) has achieved state-of-the-art performance for recommendation tasks. However, most GCF structures simplify the feature transformation and nonlinear operation during message passing in the graph convolution network (GCN). We revisit these two components and discover that a part of feature transformation and nonlinear operation during message passing in GCN can improve the representation of GCF, but increase the difficulty of training.

In this work, we propose a simple and effective graph-based recommendation model called FourierKAN-GCF. Specifically, it utilizes a novel Fourier Kolmogorov-Arnold Network (KAN) to replace the multilayer perceptron (MLP) as a part of the feature transformation during message passing in GCN, which improves the representation power of GCF and is easy to train. We further employ message dropout and node dropout strategies to improve the representation power and robustness of the model. Extensive experiments on two public datasets demonstrate the superiority of FourierKAN-GCF over most state-of-the-art methods. The implementation code is available at https://github.com/Jinfeng-Xu/FKAN-GCF.

FourierKAN-GCF

Neural Graph Collaborative Filtering (NGCF)

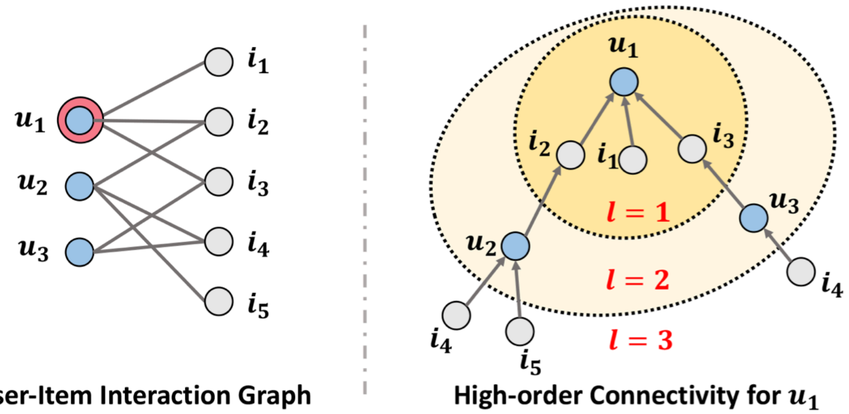


Figure 1: An illustration of the user-item interaction graph and the high-order connectivity. The node u_1 is the target user to provide recommendations for.

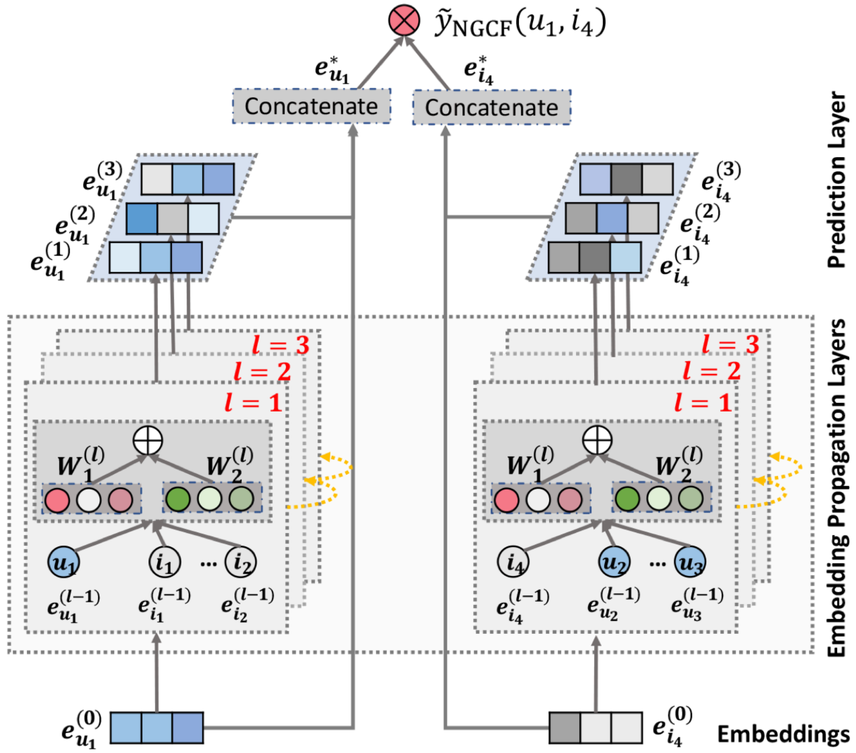


Figure 2: An illustration of NGCF model architecture (the arrowed lines present the flow of information). The representations of user u_1 (left) and item i_4 (right) are refined with multiple embedding propagation layers, whose outputs are concatenated to make the final prediction.

$$\mathbf{e}_u^{(l)} = \text{LeakyReLU}\left(\mathbf{m}_{u\leftarrow u}^{(l)} + \sum_{i\in\mathcal{N}_u} \mathbf{m}_{u\leftarrow i}^{(l)}\right),$$

FourierKAN-GCF

3.2 Fourier KAN

However, KAN is more difficult to train than MLP due to the spline function, which does not satisfy our motivation to provide a more effective and efficient method to replace $\mathbf{W}_2(\mathbf{e}_i^{(l)} \odot \mathbf{e}_u^{(l)})$ part. The essence of KAN is to form an arbitrary function by superposition of multiple nonlinear functions. Our goal can be converted into finding the split from a complex function into multiple relatively simple nonlinear functions. Naturally, the Fourier Coefficients [12] is an interesting choice. Therefore, we propose the following equation:

$$\phi_F(\mathbf{x}) = \sum_{i=1}^{d} \sum_{k=1}^{g} \left(\cos(k\mathbf{x}_i) \cdot a_{ik} + \sin(k\mathbf{x}_i) \cdot b_{ik}\right), \tag{8}$$

where d is the dimension number of features. Fourier coefficients a_{ik} and b_{ik} are trainable. Hyper-parameter g is the gridsize, which plays a critical role in controlling the number of terms (frequencies) used in the Fourier series expansion. Specifically, g determines how many different sine and cosine terms are included in the Fourier Coefficients corresponding to each input dimension. The Fourier Coefficients has a significant advantage in computational efficiency and solves the training difficulty caused by the spline function.

3.3 FourierKAN-GCF

The message passing in our FourierKAN-GCF is defined as:

$$\begin{aligned} \mathbf{e}_u^{(l+1)} &= \sigma\left(\mathbf{e}_u^{(l)} + \sum_{i \in \mathcal{N}_u} \frac{\mathbf{e}_i^{(l)} + \phi_F(\mathbf{e}_i^{(l)} \odot \mathbf{e}_u^{(l)})}{\sqrt{|\mathcal{N}_u||\mathcal{N}_i|}}\right), \\ \mathbf{e}_i^{(l+1)} &= \sigma\left(\mathbf{e}_i^{(l)} + \sum_{u \in \mathcal{N}_i} \frac{\mathbf{e}_u^{(l)} + \phi_F(\mathbf{e}_u^{(l)} \odot \mathbf{e}_i^{(l)})}{\sqrt{|\mathcal{N}_u||\mathcal{N}_i|}}\right), \end{aligned} \tag{9}$$

where $\phi_F(\cdot)$ is simplified single layer Fourier KAN function. We remove the unnecessary transform matrix $\mathbf{W}_1$ in NGCF and utilize our Fourier KAN function to replace the transform matrix $\mathbf{W}_2$. The final user embedding and item embedding are calculated as:

$$\hat{\mathbf{e}}_u = \mathbf{e}_u^{(1)}||\mathbf{e}_u^{(2)}||...||\mathbf{e}_u^{(L)}, \quad \hat{\mathbf{e}}_i = \mathbf{e}_i^{(1)}||\mathbf{e}_i^{(2)}||...||\mathbf{e}_i^{(L)}, \tag{10}$$

where $||$ is the concatenation operation.

FourierKAN-GCF

Table 2: Performance comparison of baselines and FourierKAN-GCF in terms of R@K and N@K. The superscript * indicates th improvement is statistically significant where the p-value is less than 0.05.

Datasets	[1]MOOC						[2]Games					
Metrics	R@10	R@20	R@50	N@10	N@20	N@50	R@10	R@20	R@50	N@10	N@20	N@50
BPR-MF	0.2401	0.3353	0.4813	0.1571	0.1898	0.2261	0.0210	0.0369	0.0699	0.0135	0.0183	0.0265
BUIR	0.2354	0.3196	0.4801	0.1589	0.1835	0.2224	0.0227	0.0384	0.0749	0.0143	0.0192	0.0282
NGCF	0.2453	0.3361	0.4799	0.1722	0.1894	0.2349	0.0231	0.0379	0.0782	0.0142	0.0196	0.0274
LR-GCCF	0.2466	0.3336	0.4809	0.1678	0.1938	0.2294	0.0253	0.0440	0.0815	0.0171	0.0224	0.0317
LightGCN	0.2402	0.3307	0.4773	0.1581	0.1811	0.2217	0.0271	0.0447	0.0844	0.0174	0.0227	0.0326
UltraGCN	0.2372	0.3194	0.4701	0.1737	0.1962	0.2307	0.0277	0.0459	0.0844	0.0173	0.0230	0.0331
IMP-GCN	0.2170	0.2788	0.4183	0.1531	0.1717	0.2057	0.0273	0.0461	0.0839	0.0171	0.0232	0.0323
w/o MD	0.2518	0.3523	0.4912	0.1814	0.2116	0.2449	0.0267	0.0452	0.0825	0.0165	0.0221	0.0314
w/o ND	**0.2603**	0.3527	0.4839	0.1800	0.2071	0.2439	0.0257	0.0452	0.0809	0.0161	0.0219	0.0309
KAN-GCF	0.2536	0.3417	0.4984	0.1758	0.2024	0.2396	0.0272	0.0451	0.0837	0.0172	0.0229	0.0325
FourierKAN-GCF	0.2595	**0.3564**	**0.5065**	**0.1866**	**0.2147**	**0.2462**	**0.0287**	**0.0473**	**0.0856**	**0.0176**	**0.0252**	**0.0342**

[1]Dataset can be accessed at http://moocdata.cn. [2]Dataset can be accessed at http://jmcauley.ucsd.edu/data/amazon/links.html.

[HTML] The enabling technologies for digitalization in the chemical process industry [HTML] sciencedirect.com

M Pietrasik, A Wilbik, P Grefen - Digital Chemical Engineering, 2024 - Elsevier

34 days ago - In this paper, we provide an overview of the technologies that enable digitalization in the chemical process industry and describe their applications to solve …

☆ Save 99 Cite Related articles All 3 versions

Kolmogorov-Arnold Networks for Time Series: Bridging Predictive Power and Interpretability [PDF] arxiv.org

K Xu, L Chen, S Wang - arXiv preprint arXiv:2406.02496, 2024 - arxiv.org

34 days ago - Kolmogorov-Arnold Networks (KAN) is a groundbreaking model recently proposed by the MIT team, representing a revolutionary approach with the potential to be a …

☆ Save 99 Cite Cited by 6 Related articles All 2 versions »

TKAN-2

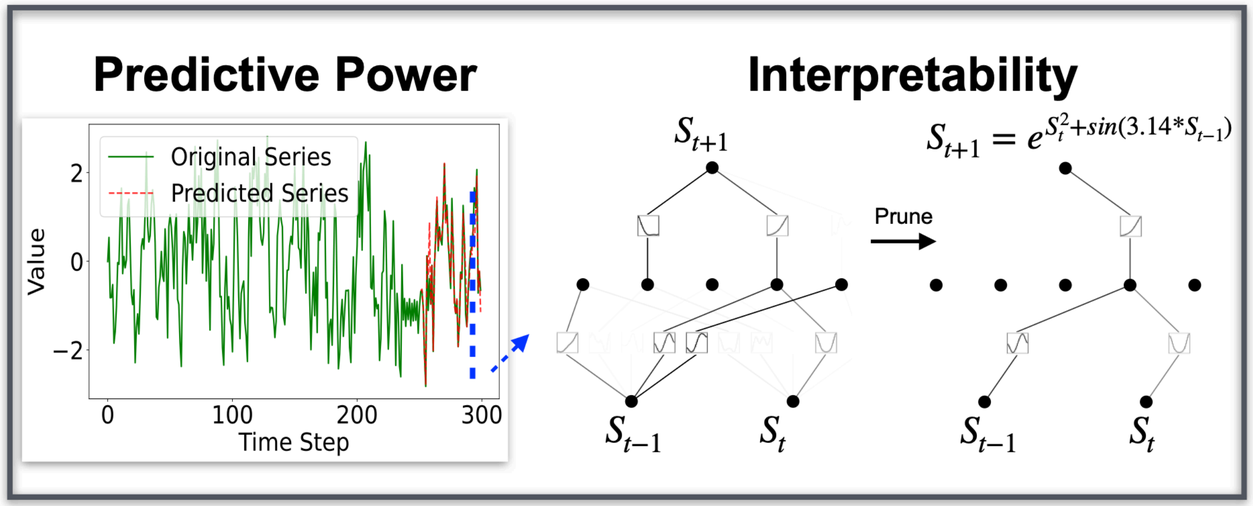


Figure 1: Predictive and Interpretable Capabilities of KAN in Time Series.

TKAT

A Temporal Kolmogorov-Arnold Transformer for Time Series Forecasting

[PDF] arxiv.org

R Genet, H Inzirillo - arXiv preprint arXiv:2406.02486, 2024 - arxiv.org

35 days ago - Capturing complex temporal patterns and relationships within multivariate data streams is a difficult task. We propose the Temporal Kolmogorov-Arnold Transformer (TKAT) …

☆ Save 99 Cite Cited by 3 Related articles All 2 versions »

***Abstract*—Capturing complex temporal patterns and relationships within multivariate data streams is a difficult task. We propose the Temporal Kolmogorov-Arnold Transformer (TKAT), a novel attention-based architecture designed to address this task using Temporal Kolmogorov-Arnold Networks (TKANs). Inspired by the Temporal Fusion Transformer (TFT), TKAT emerges as a powerful encoder-decoder model tailored to handle tasks in which the observed part of the features is more important than the a priori known part. This new architecture combined the theoretical foundation of the Kolmogorov-Arnold representation with the power of transformers. TKAT aims to simplify the complex dependencies inherent in time series, making them more "interpretable". The use of transformer architecture in this framework allows us to capture long-range dependencies through self-attention mechanisms.**

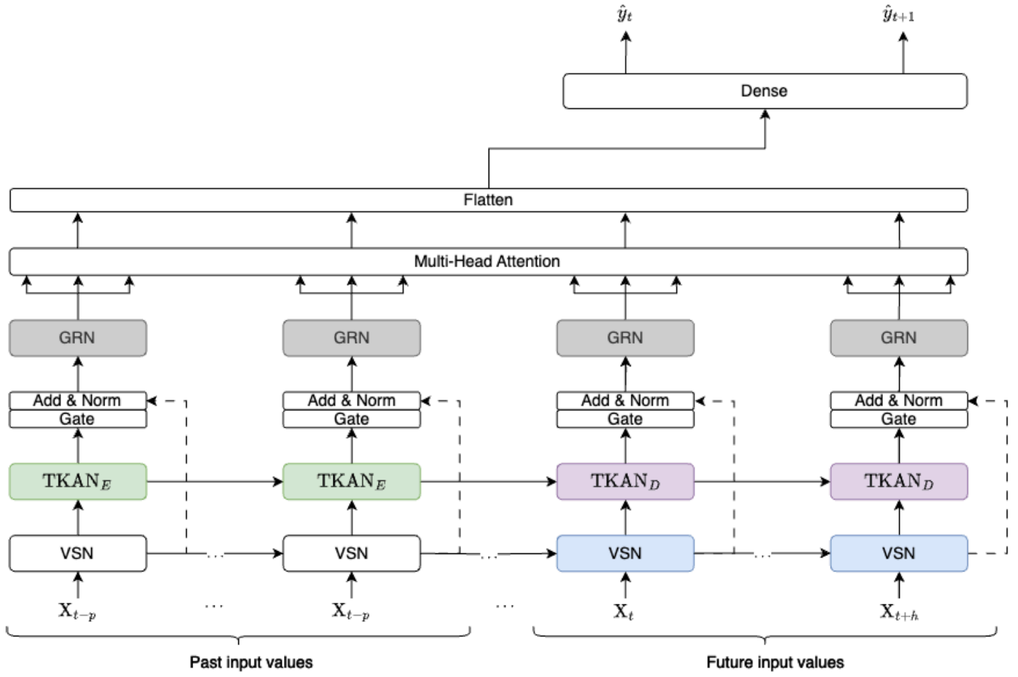


Fig. 1: Temporal Kolmogorov-Arnold Transformer (TKAT)

TKAT

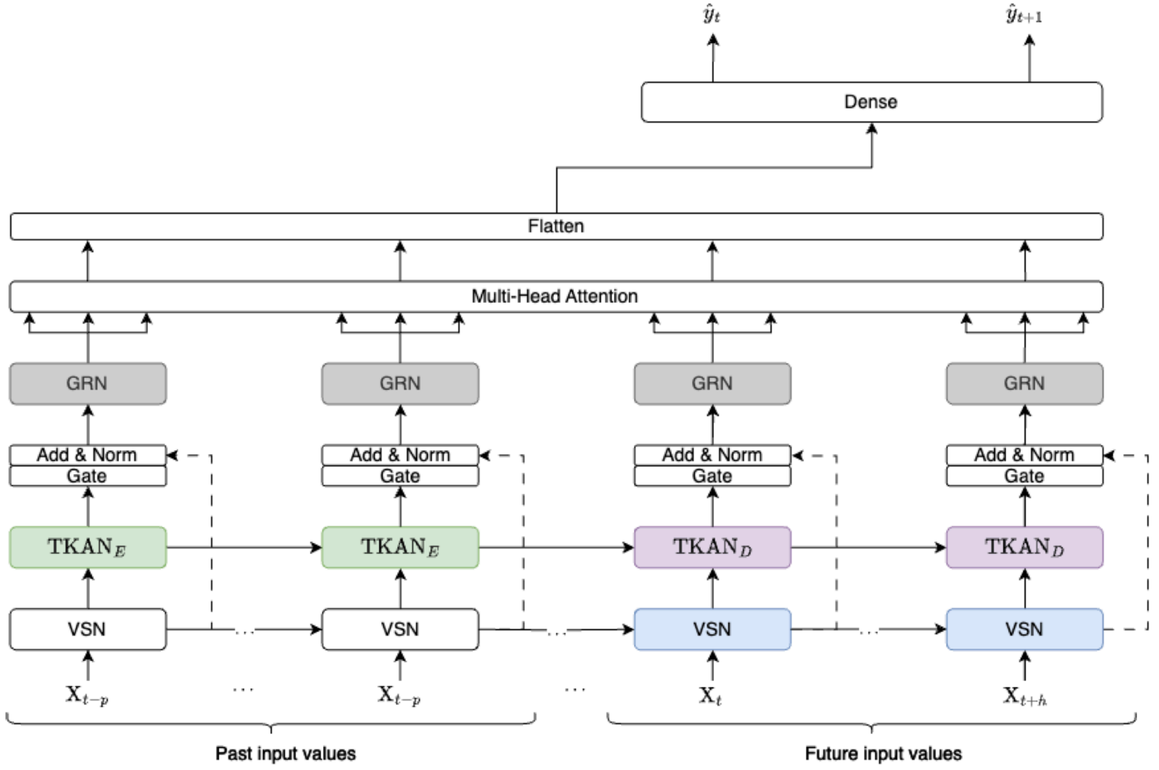


Fig. 1: Temporal Kolmogorov-Arnold Transformer (TKAT)

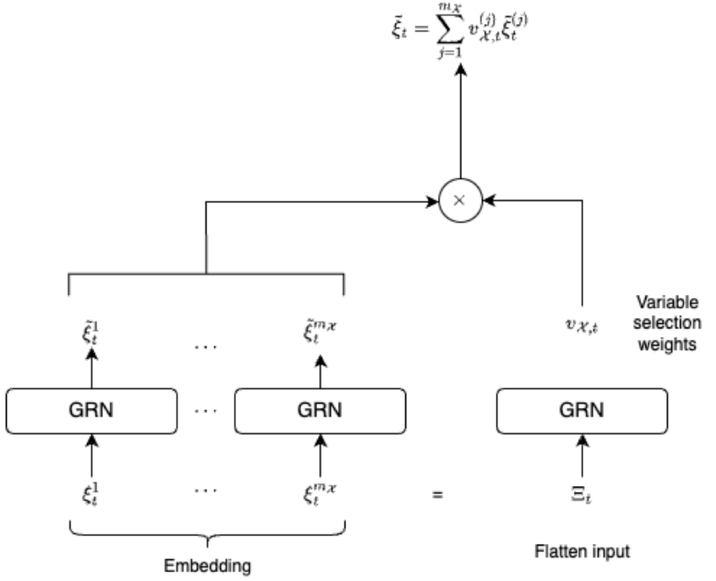


Fig. 4: Variable Selection Networks (VSN)

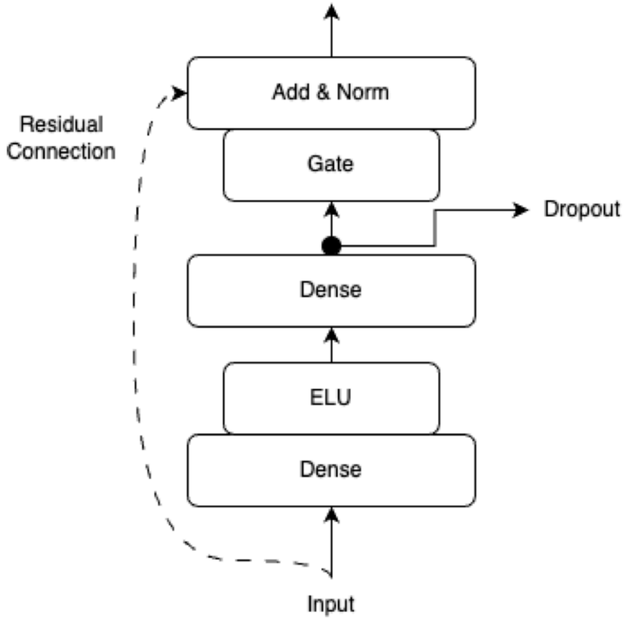


Fig. 3: Gated Residual Networks (GRN) [20]

TKAT

TABLE I: R^2 Average: TKAT vs Benchmark

Time	TKAT	TKAT non-KAN	TKAN	GRU	LSTM
1	0.30519	0.29834	0.35084	0.36513	0.35553
3	0.21801	0.22146	0.19884	0.20067	0.06122
6	0.17955	0.17584	0.14054	0.08250	-0.22583
9	0.16476	0.15378	0.11747	0.08716	-0.29058
12	0.14908	0.15179	0.10511	0.01786	-0.47322
15	0.14504	0.12658	0.08607	0.03342	-0.40443

TABLE II: R^2 Standard Deviation: TKAT vs Benchmark

Time	TKAT	TKAT non-KAN	TKAN	GRU	LSTM
1	0.01886	0.01610	0.01420	0.00833	0.01116
3	0.00906	0.00485	0.00738	0.00484	0.08020
6	0.00654	0.00352	0.00723	0.02363	0.06271
9	0.00896	0.00578	0.01455	0.01483	0.05272
12	0.00477	0.00507	0.00391	0.08638	0.08574
15	0.01014	0.01106	0.02465	0.02407	0.09272

TABLE VI: Trainable Parameters Count

Model	Number of Trainable Parameters	R^2
TKAT	1,561,704	0.127743
TKAN	102,355	0.068004
MLP	298,230	-0.036020
GRU	99,930	0.055263
LSTM	131,430	-0.008245

The results were obtained after a single run for this graph.

U-KAN Makes Strong Backbone for Medical Image Segmentation and Generation

[PDF] arxiv.org

C Li, X Liu, W Li, C Wang, H Liu, Y Yuan - arXiv preprint arXiv:2406.02918, 2024 - arxiv.org

34 days ago - U-Net has become a cornerstone in various visual applications such as image segmentation and diffusion probability models. While numerous innovative designs and …

☆ Save 99 Cite Cited by 6 Related articles All 2 versions »

Leveraging KANs For Enhanced Deep Koopman Operator Discovery

[PDF] arxiv.org

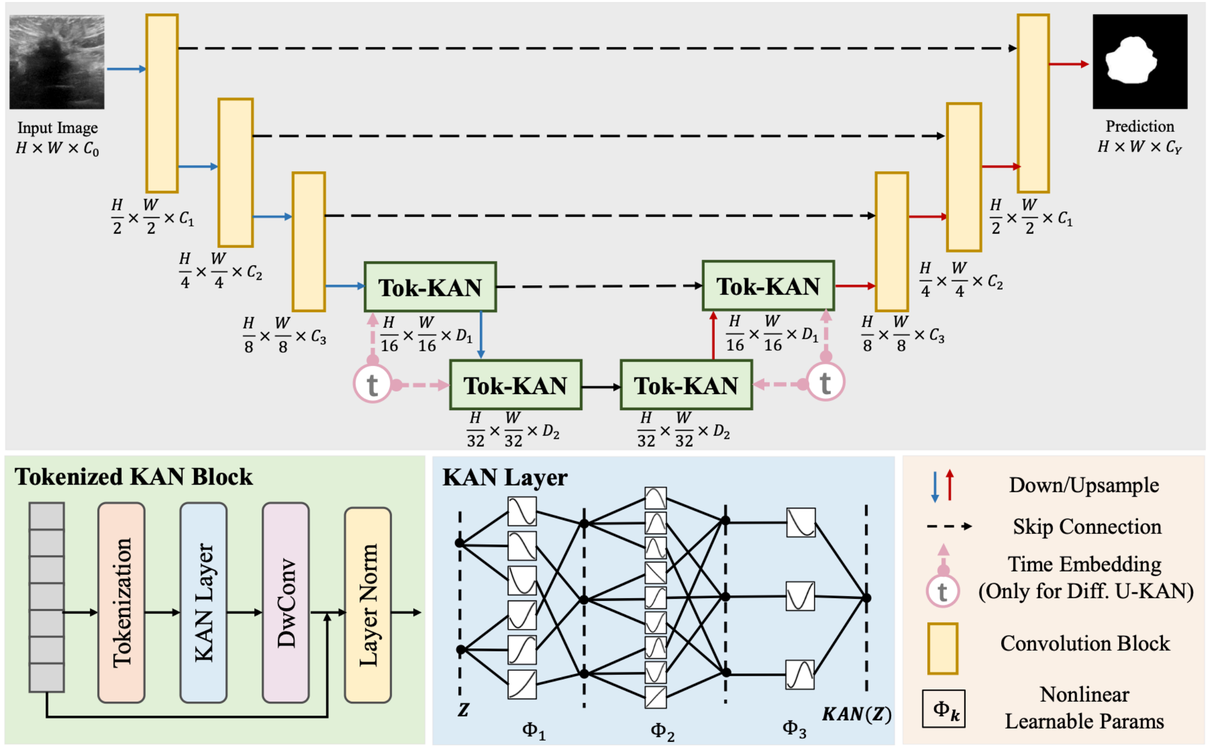
G Nehma, M Tiwari - arXiv preprint arXiv:2406.02875, 2024 - arxiv.org

34 days ago - Multi-layer perceptrons (MLP's) have been extensively utili: Deep Koopman operators for linearizing nonlinear dynamics. With the em

☆ Save 99 Cite Cited by 1 Related articles All 2 versions »

Table 2: Overall comparison with state-of-the-art segmentation models w.r.t. efficiency and seg tation metrics.

Methods	Average Seg.		Efficiency	
	IoU↑	F1↑	Gflops	Params (M)
U-Net [71]	75.89±2.14	85.25±1.52	524.2	34.53
Att-Unet [63]	75.51±1.77	84.85±1.26	533.1	34.9
U-Net++ [94]	76.36±2.33	85.53±1.74	1109	36.6
U-NeXt [81]	72.80±0.54	83.33±0.57	4.58	1.47
Rolling-UNet [54]	76.76±1.01	85.92±0.89	16.82	1.78
U-Mamba [56]	77.87±1.47	86.73±1.25	2087	86.3
Seg. U-KAN (Ours)	**78.69±1.27**	**87.22±1.15**	14.02	6.35

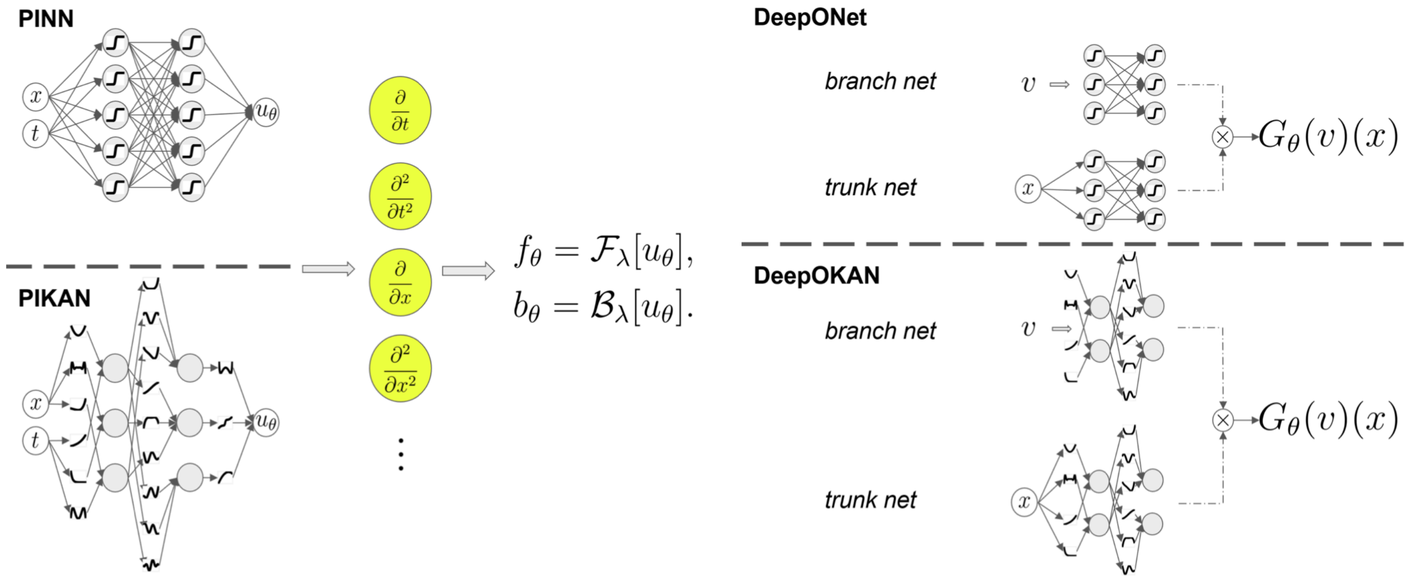


A comprehensive and FAIR comparison between MLP and KAN representations for differential equations and operator networks

[PDF] arxiv.org

K Shukla, JD Toscano, Z Wang, Z Zou… - arXiv preprint arXiv …, 2024 - arxiv.org

34 days ago - Kolmogorov-Arnold Networks (KANs) were recently introduced as an alternative representation model to MLP. Herein, we employ KANs to construct physics …



(a) Physics-informed networks. (b) Operator networks.

Figure 1: An illustration of MLP and KAN for **(a)** differential equations and **(b)** operator networks. We choose DeepONet [34] as the representation model for operator learning. Here activation function for MLP in PINNs and DeepONets is chosen as the hyperbolic tangent only for the demonstration.

Re	Method	polynomial order	Relative L^2 error (u, v, p) %	Time (ms/it)	Num. of parameters
400	PINN	-	0.25, 0.37, 2.25	32	44, 283
	Chebyshev PIKAN	3	1.13, 1.38, 2.34	61	4, 736
		5	0.20, 0.26, 1.63	81	7, 104
		8	0.18, 0.21, 1.8	112	10, 656
	Legendre PIKAN	3	2.0, 2.3, 3.2	82	4, 736
		8	0.21,0.29,1.77	241	10, 656
	Jacobi PIKAN	3	0.31,0.43, 1.96	120	4, 736
		8	0.18,0.26, 1.67	235	10, 656
	Hermite PIKAN	3	4.1,4.9,9.7	84	10, 656
		8	143,101,151	220	10, 656
	PINN+RBA	-	0.24, 0.33, 1.83	34	44, 283
	Chebyshev PIKAN+RBA	3	0.18, 0.20, 1.78	92	4, 736
2000	PINN	-	18.9, 18.8, 22.8	63	44, 283
	Chebyshev PIKAN	5	109.2, 113.8, 132.05	85	7, 104
		8	105.2, 107.8, 110.23	127	10, 656
	PINN+EVM	-	6.4, 5.4, 6.9	64	49, 365
	Chebyshev PIKAN+EVM	8	4.2, 4.5, 8.4	135	20, 736

Table 4: Comparison on performance between PINNs and PIKANs based on different polynomials in solving 2D steady cavity flow at $Re = 400$ and $Re = 2000$. The PINNs, PIKANs, RBA and EVM are implemented on our NSFnet [71]. 10^4 training points, 9×10^5 training epochs for the case of $Re = 400$, 2×10^4 training points, 4×10^5 training epochs for the case of $Re = 2000$. The Adams optimizer [67] is used and the training is performed on an Nvidia's RTX 4090 GPU. Note that the coefficients of the Jacobi polynomial $P_n^{\alpha,\beta}(x)$ used in this section are: $\alpha = 1$, $\beta = 1$.

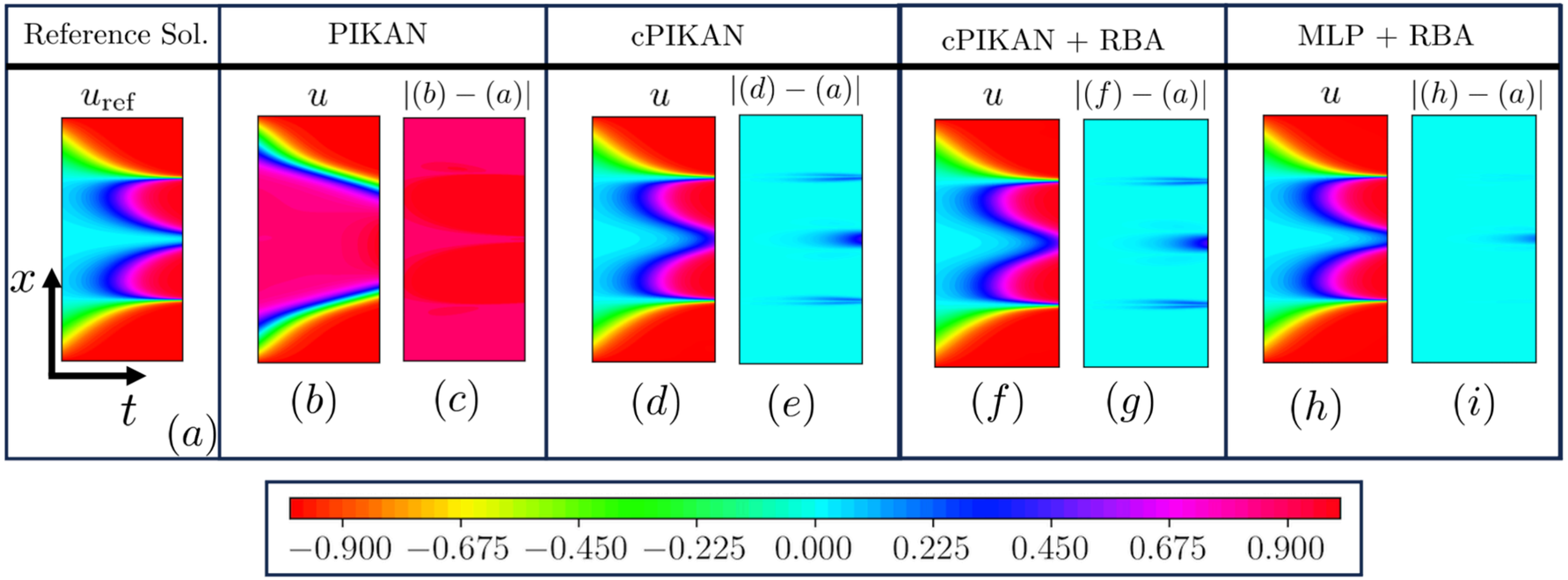


Figure 14: The solution of the Allen-Cahn equation (27) in the spatio-temporal domain of ($x \in [-1, 1]$, $t \in$

$$\partial_t u + u\partial_x u = \nu \partial_{xx} u$$

Methods	Rel. l_2–error	No. of parameters	Time: ms/iter
DeepONet	$5.83\% \pm 0.19\%$	63900	1.9
DeepOKAN 1	$2.71\% \pm 0.08\%$	252800	3.9
DeepOKAN 2	$3.02\% \pm 0.13\%$	76400	3.9

Table 7: Rel. l_2–error for learning the solution operator of the 1D Burgers' equation with viscosity $\nu = \frac{1}{100\pi}$. The architectures of the branch and trunk nets are $[128, 100, 100, 100, 100]$ and $[4, 100, 100, 100]$, respectively, for DeepONet and DeepOKAN 1, and are $[128, 50, 50, 50, 50]$ and $[4, 50, 50, 50]$ for DeepOKAN 2. Here Chebyshev KAN [26] is employed with degree three for DeepOKANs and hyperbolic tangent is used as the activation function for the DeepONet. The time per iteration is measured on an Nvidia's GeForce RTX-3090 GPU with 24 GB of memory.

Methods	1% noise	5% noise	10% noise
DeepONet	$5.83\% \pm 0.19\%$	$5.93\% \pm 0.18\%$	$6.29\% \pm 0.19\%$
DeepOKAN 1	$2.72\% \pm 0.08\%$	$2.94\% \pm 0.09\%$	$3.57\% \pm 0.05\%$
DeepOKAN 2	$3.02\% \pm 0.13\%$	$3.24\% \pm 0.12\%$	$3.91\% \pm 0.13\%$

Table 8: Rel. l_2–error for learning the solution operator of the 1D Burgers' equation when trained with clean data but tested with noisy input data. Here the noise is additive Gaussian noise with mean zero and different levels of standard deviation, and the noise level is defined as the percentage of the absolute value of the input function evaluated at the grid.

$$\nabla \cdot (\lambda(x, y) \nabla u(x, y)) = f$$

Methods	L^2 relative error	No. of parameters	Time: ms/iter
DeepONet	$1.62\% \pm 0.15\%$	147000	2.3
DeepOKAN	$2.18\% \pm 0.02\%$	585200	8.8

Table 9: Rel. l_2–error for learning the solution operator of the Darcy problem. The architectures of the branch and trunk nets are $[961, 100, 100, 100, 100]$ and $[2, 100, 100, 100]$, respectively, for both operator networks. Chebyshev KAN [26] is employed with degree three for the DeepOKAN and hyperbolic tangent is used as the activation function for the DeepONet. The time per iteration is measured on an Nvidia's GeForce RTX-3090 GPU with 24 GB of memory.

Methods	1% noise	5% noise	10% noise
DeepONet	$1.66\% \pm 0.15\%$	$2.25\% \pm 0.12\%$	$3.47\% \pm 0.12\%$
DeepOKAN	$2.18\% \pm 0.02\%$	$2.20\% \pm 0.02\%$	$2.30\% \pm 0.03\%$

Table 10: Rel. l_2–error for learning the solution operator of the Darcy problem when trained with clean data but tested with noisy input data. The noise is additive Gaussian noise with mean zero and different levels of standard deviation, and the noise level is defined as the percentage of the absolute value of the input function evaluated at the grid.

Adaptive Multi-Scale Decomposition Framework for Time Series Forecasting

[PDF] arxiv.org

Y Hu, P Liu, P Zhu, D Cheng, T Dai - arXiv preprint arXiv:2406.03751, 2024 - arxiv.org

33 days ago - Transformer-based and MLP-based methods have emerged as leading approaches in time series forecasting (TSF). While Transformer-based methods excel in …

☆ Save Cite Related articles All 2 versions »

KAN-EEG: Towards Replacing Backbone-MLP for an Effective Seizure Detection System

[PDF] medrxiv.org

LF Herbozo Contreras, J Cui, L Yu, Z Huang, A Nikpour… - medRxiv, 2024 - medrxiv.org

31 days ago - The landscape of artificial intelligence (AI) research is witnessing a transformative shift with the emergence of the Kolmogorov-Arnold Network (KAN) …

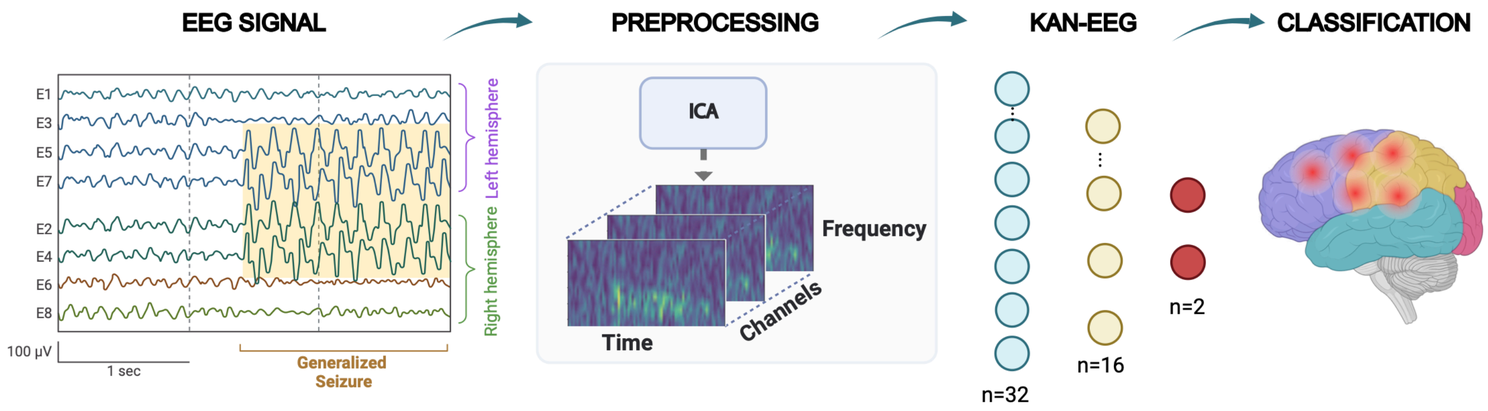


Fig. 3. KAN-EEG Seizure System. We analyzed different kinds of seizures utilizing the TUH dataset for training. We then preprocessed our model utilizing Independent Component Analysis (ICA) and Short-Time Fourier Transform (STFT). Subsequently, we incorporate a shallow KAN algorithm that leads to efficient results.

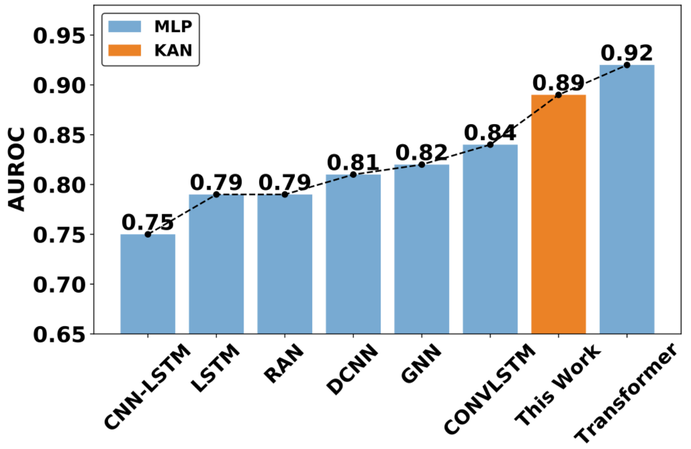


Fig. 4. Comparison of KAN-EEG (◇Architecture **I-764-256-O** [see Table II]) training on TUH dataset with other MLP based models [31], [37], [46]–[48]. Our model was trained on 400 hours of data, which is significantly less than the amount of data used to train other models, such as ConvLSTM and Transformer, with 752 hours and 910 hours, respectively.

Seizure: 발작

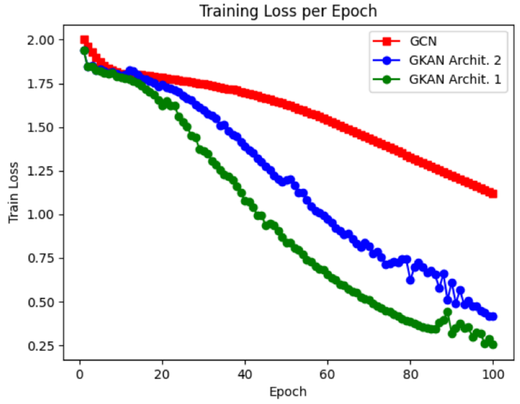
GKAN

GKAN: Graph Kolmogorov-Arnold Networks

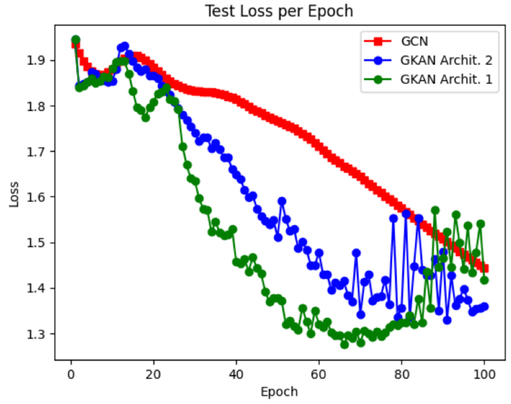
M Kiamari, M Kiamari, B Krishnamachari - arXiv preprint arXiv:2406.06470, 2024 -

29 days ago - We introduce Graph Kolmogorov-Arnold Networks (GKAN), an innov
neural network architecture that extends the principles of the recently proposed Ko

☆ Save 99 Cite Cited by 3 Related articles All 2 versions »

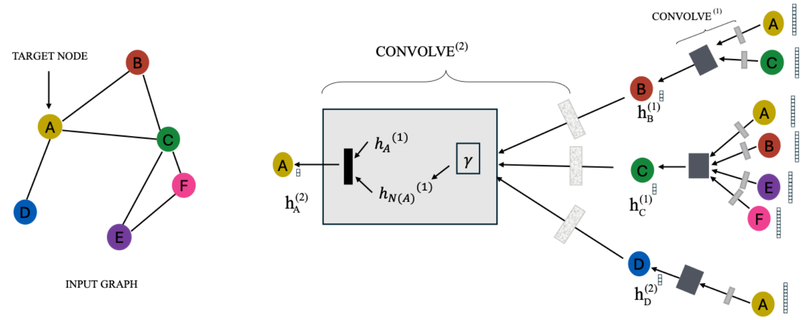
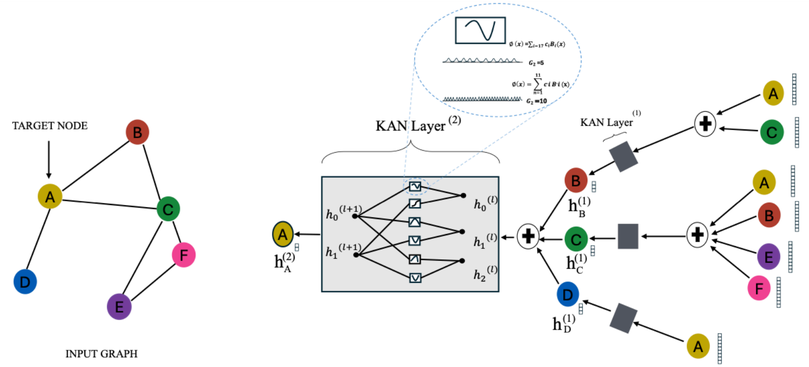
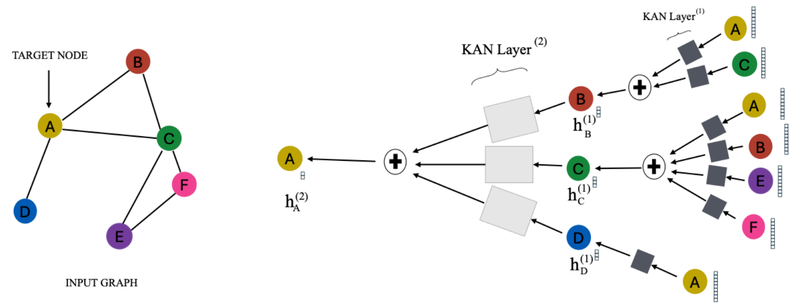


(a) Training loss of different schemes.



(b) Test loss of different schemes.

Figure 3: Loss value comparison of GCN and GKAN architectures for $k = 1$ and $g = 3$.

Scheme	Overview of Model Architecture
GCN Ying et al. (2018)	
GKAN Architecture 1	
GKAN Architecture 2	

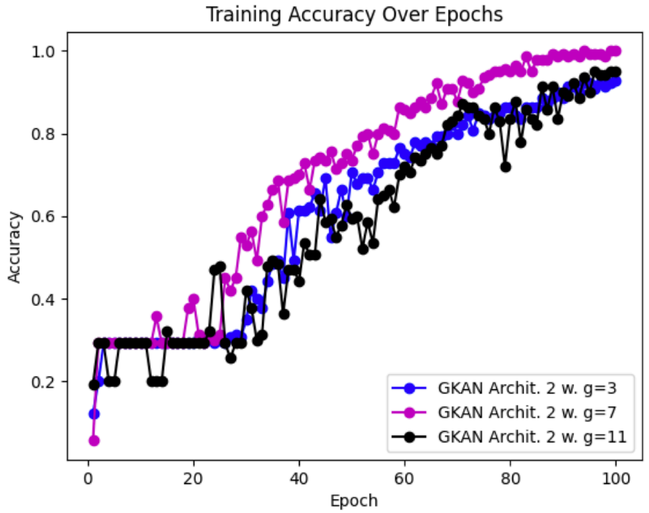
(a) Overview of a two-layer GCN Ying et al. (2018) architecture.

(b) Overview of a two-layer GKAN Architecture 1.

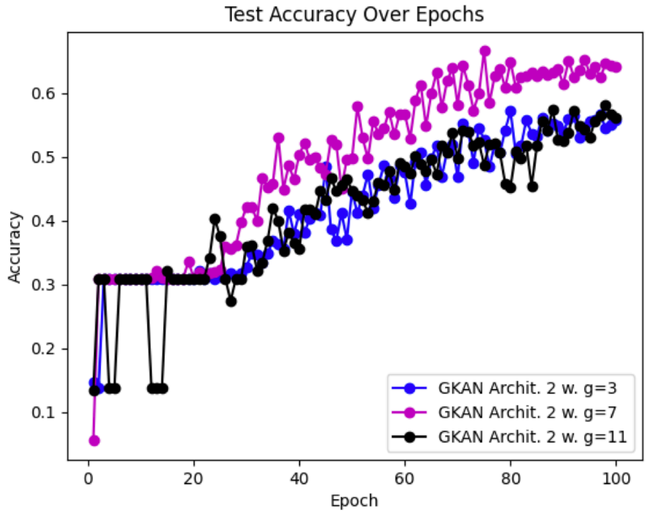
(c) Overview of a two-layer GKAN Architecture 2.

Figure 1: Comparison of different model architectures.

GKAN

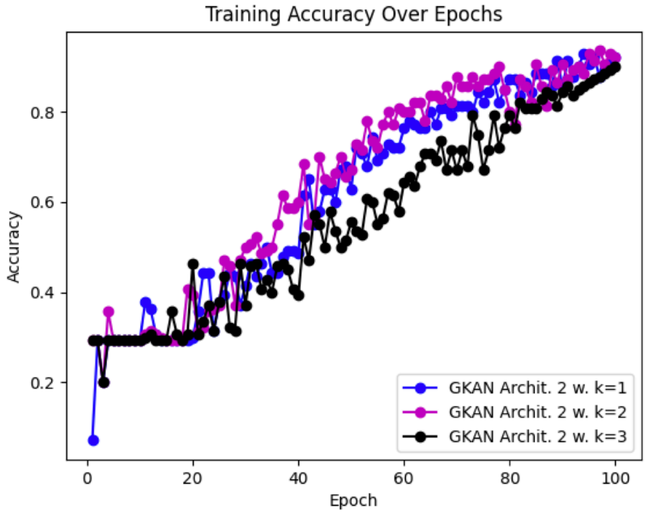


(a) Training accuracy of GKAN Architecture 2 for different parameter g and $k = 1$.

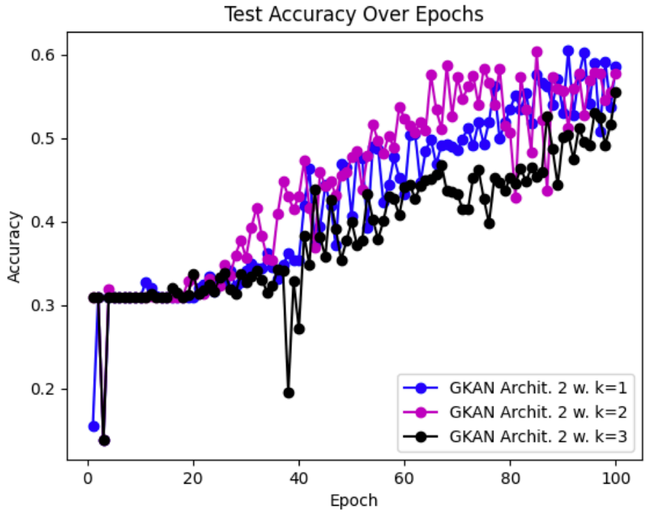


(b) Test accuracy of GKAN Architecture 2 for different parameter g and $k = 1$.

Figure 4: Accuracy comparison of GKAN Architecture 2 for $g \in \{1, 2, 3\}$, $k = 1$, and $h = 16$.



(a) Training accuracy of GKAN Architecture 2 for $k \in \{1, 2, 3\}$ while fixing $g = 3$.



(b) Test accuracy of GKAN Architecture 2 for $k \in \{1, 2, 3\}$ and fixed $g = 3$.

Figure 6: Accuracy of GKAN Architecture 2 for $k \in \{1, 2, 3\}$, $g = 3$, and $h = 16$.

fKAN

fKAN: Fractional Kolmogorov-Arnold Networks with trainable Jacobi basis functions

[PDF] arxiv.org

AA Aghaei - arXiv preprint arXiv:2406.07456, 2024 - arxiv.org

28 days ago - Recent advancements in neural network design have given rise to the development of Kolmogorov-Arnold Networks (KANs), which enhance speed …

☆ Save 99 Cite Cited by 3 Related articles All 2 versions »

Abstract. Recent advancements in neural network design have given rise to the development of Kolmogorov-Arnold Networks (KANs), which enhance speed, interpretability, and precision. This paper presents the Fractional Kolmogorov-Arnold Network (fKAN), a novel neural network architecture that incorporates the distinctive attributes of KANs with a trainable adaptive fractional-orthogonal Jacobi function as its basis function. By leveraging the unique mathematical properties of fractional Jacobi functions, including simple derivative formulas, non-polynomial behavior, and activity for both positive and negative input values, this approach ensures efficient learning and enhanced accuracy. The proposed architecture is evaluated across a range of tasks in deep learning and physics-informed deep learning. Precision is tested on synthetic regression data, image classification, image denoising, and sentiment analysis. Additionally, the performance is measured on various differential equations, including ordinary, partial, and fractional delay differential equations. The results demonstrate that integrating fractional Jacobi functions into KANs significantly improves training speed and performance across diverse fields and applications.

fKAN

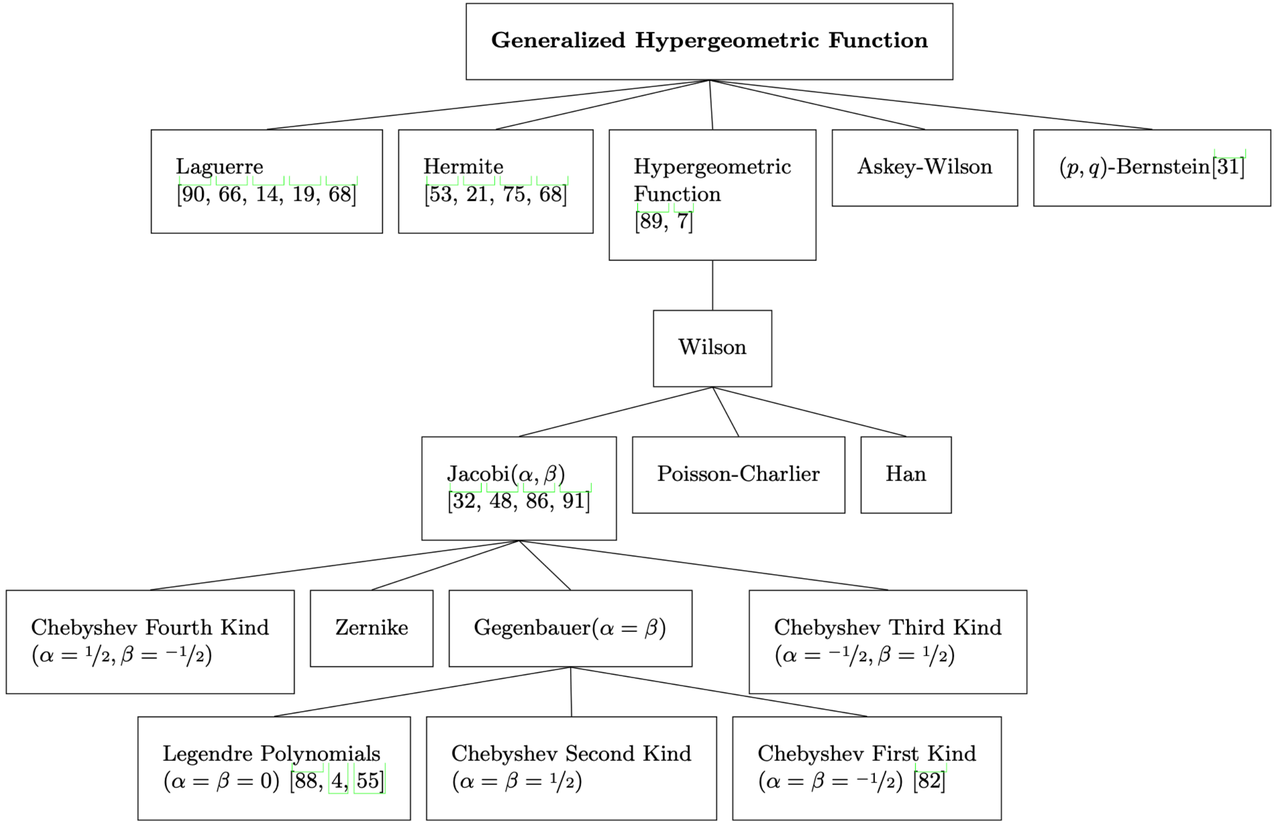


Fig. 1: This figure depicts the hierarchical classification of the generalized hypergeometric function. It shows that Chebyshev and Gegenbauer polynomials are particular cases within the more general class of Jacobi polynomials, which itself is a subclass of the hypergeometric function.

$$(a_i)_n = \frac{\Gamma(a_i + n)}{\Gamma(a_i)} = a_i(a_i + 1)(a_i + 2)\dots(a_i + n - 1), \quad n > 0,$$

fKAN

The Jacobi polynomials $\mathcal{J}_n^{(\alpha,\beta)}(\zeta)$ are a special case of hypergeometric functions. The generalized form of the hypergeometric function ${}_p\mathcal{F}_q : \mathbb{C}^{p+q+1} \to \mathbb{C}$ can be represented as:

$$ {}_p\mathcal{F}_q \left(\begin{matrix} a_1, a_2, \dots, a_p \\ b_1, b_2, \dots, b_q \end{matrix} ; \zeta \right) = \sum_{n=0}^{\infty} \frac{(a_1)_n (a_2)_n \dots (a_p)_n}{(b_1)_n (b_2)_n \dots (b_q)_n} \frac{\zeta^n}{n!}, $$

For the specific choice $p = 2, q = 1, a_1 = -n, a_2 = n + 1 + \alpha + \beta$, and $b_1 = 1 + \alpha$ where $\alpha, \beta > -1$, the Gaussian hypergeometric series reduces to the Jacobi polynomials:

$$ \mathcal{J}_n^{(\alpha,\beta)}(\zeta) = \frac{(\alpha + 1)_n}{n!} \, {}_2\mathcal{F}_1 \left(\begin{matrix} -n, n + 1 + \alpha + \beta \\ 1 + \alpha \end{matrix} ; \tfrac{1}{2}(1 - \zeta) \right). $$

fKAN

$$\phi_{p,q}^{(\gamma)}(\boldsymbol{\zeta}_p) = \mathcal{J}_q^{(\alpha,\beta)}\left(\varphi_{(0,1;\gamma)}(\sigma(\boldsymbol{\zeta}_p))\right).$$

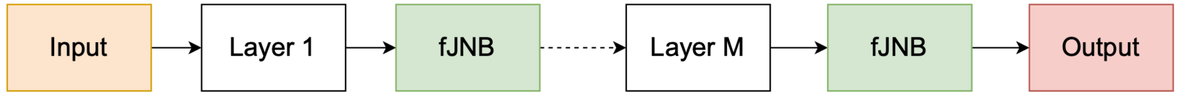
$$\begin{aligned}
\mathcal{J}_0^{(\alpha,\beta)}(\zeta) &= 1, \\
\mathcal{J}_1^{(\alpha,\beta)}(\zeta) &= A_0\zeta + B_0, \\
\mathcal{J}_{n+1}^{(\alpha,\beta)}(\zeta) &= (A_n\zeta + B_n)\mathcal{J}_n^{(\alpha,\beta)}(\zeta) - C_n\mathcal{J}_{n-1}^{(\alpha,\beta)}(\zeta), \quad n \geq 1,
\end{aligned}$$

$$\varphi_{(d_0,d_1,\gamma)}(\zeta) = \frac{2\zeta^\gamma - d_0 - d_1}{d_1 - d_0}.$$

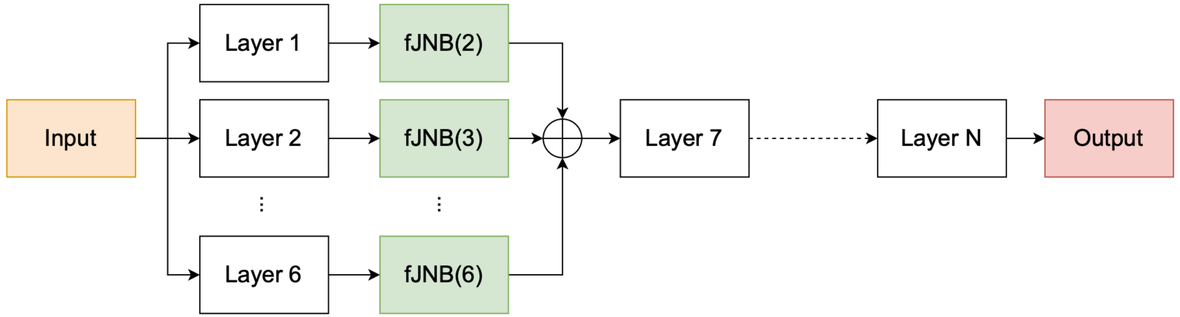
$$\begin{aligned}
A_n &= \frac{(2n+\alpha+\beta+1)(2n+\alpha+\beta+2)}{2(n+1)(n+\alpha+\beta+1)}, \\
B_n &= \frac{(\alpha^2-\beta^2)(2n+\alpha+\beta+1)}{2(n+1)(n+\alpha+\beta+1)(2n+\alpha+\beta)}, \\
C_n &= \frac{(n+\alpha)(n+\beta)(2n+\alpha+\beta+2)}{(n+1)(n+\alpha+\beta+1)(2n+\alpha+\beta)}.
\end{aligned}$$

$$\hat{\chi}(\boldsymbol{\zeta}) = \sum_{q=0}^{Q} \boldsymbol{\Phi}_q^{(\gamma)}\left(\sum_{p=1}^{d} \phi_{q,p}^{(\gamma)}(\boldsymbol{\zeta}_p)\right).$$

fKAN

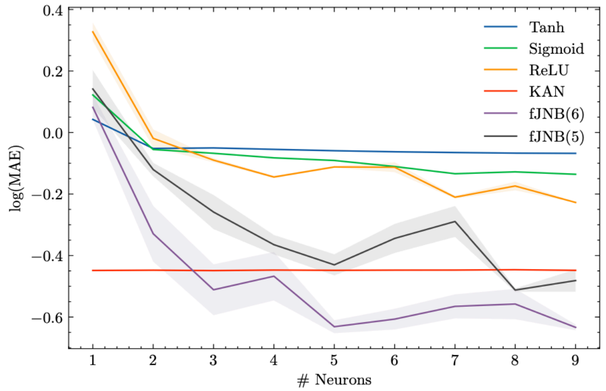


(a) Utilizing fJNB in a sequential architecture, where the value of q may vary for each block. This flexibility allows for the incorporation of different fractional Jacobi basis functions with varying degrees of complexity across successive blocks in the network. By adapting the value of q, the network can tailor its representation capabilities to better capture the intricacies of the underlying data.

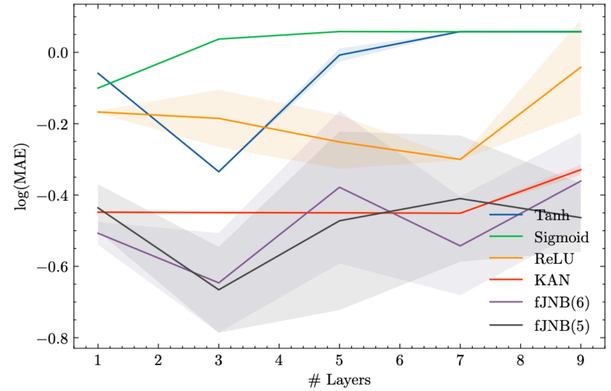


(b) Employing fJNB with varying values of q concurrently, this deep network comprises six subnetworks, each equipped with distinct fJNBs. By incorporating fJNBs with different q values, the network can capture a diverse range of features simultaneously. The outputs of these fJNBs are then fused to produce the final prediction, leveraging the collective information gleaned from multiple representations.

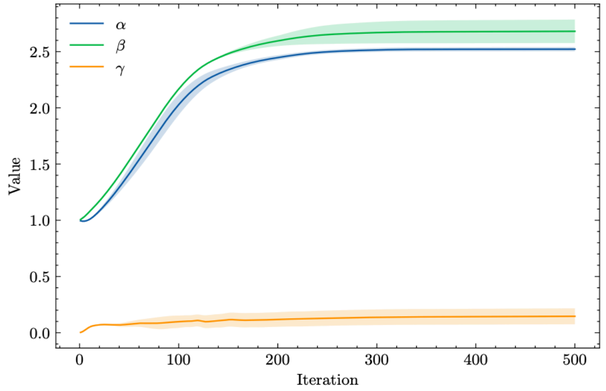
$$\chi(\zeta) := \sin(\pi\zeta) + 10\exp\left(\frac{\zeta}{5}\right) + \frac{\epsilon}{100},$$



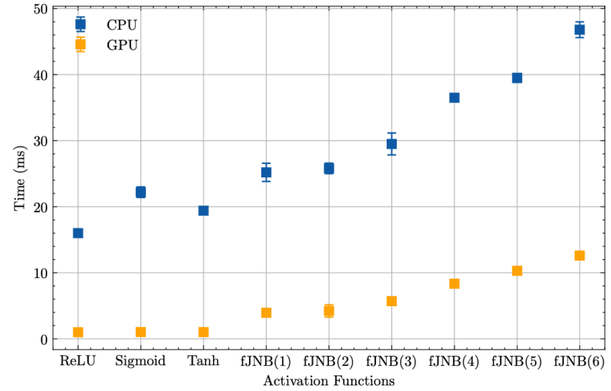
(a) Comparison of prediction accuracy for a single-layer neural network with different activation functions. The proposed activation function demonstrates higher accuracy than well-known functions and KAN.



(b) Accuracy comparison for an MLP with a fixed number of neurons per layer. The proposed network performs better than traditional activation functions and alternatives, though with higher variance.



(c) Evolution of Jacobi polynomial parameters (α, β, and γ) during training. The parameters converge to optimal values after approximately 250 iterations.



(d) Comparison of CPU and GPU time required to compute various activation functions. The times are the mean of 100 different runs.

Fig. 4: The results of simulation of a one-dimensional function regression task for different activation functions. The proposed activation function demonstrates higher accuracy than well-known functions as well as KAN.

fKAN

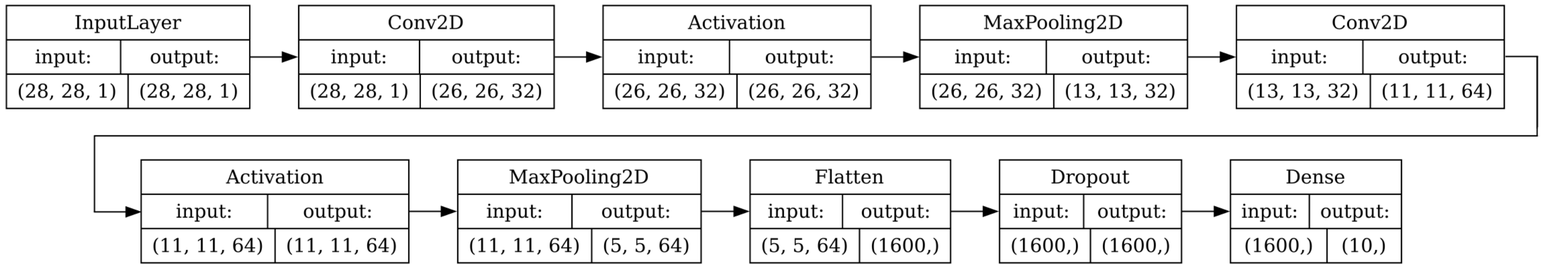


Fig. 5: The architecture of proposed method for MNIST classification data.

Act. Func.	Loss		Accuracy	
	Mean	Std.	Mean	Std.
Sigmoid	0.0611	0.0028	98.092	0.0937
Tanh	0.0322	0.0015	98.904	0.0695
ReLU	0.0256	0.0010	99.140	0.0434
fJNB(2)	0.0252	0.0017	99.134	0.0484
fJNB(3)	0.0224	0.0019	99.200	0.0787
fJNB(4)	**0.0217**	0.0008	**99.228**	0.0515
fJNB(5)	0.0249	0.0009	99.204	0.0467
fJNB(6)	0.0290	0.0028	99.024	0.1198

Table 2: Performance of different activation functions in a CNN for classifying MNIST dataset.

fKAN

Act. Func.	Loss	Accuracy	Precision	Recall	ROC-AUC
Sigmoid	0.604 ± 0.039	85.198 ± 0.468	81.440 ± 1.511	91.266 ± 1.532	91.631 ± 0.422
Tanh	0.610 ± 0.060	85.604 ± 0.818	82.364 ± 2.790	90.912 ± 2.705	91.894 ± 0.505
ReLU	0.808 ± 0.064	83.826 ± 1.141	78.197 ± 2.124	$\mathbf{93.995 \pm 1.358}$	91.080 ± 0.404
JNB(1)	0.724 ± 0.074	84.462 ± 1.156	79.967 ± 3.160	92.387 ± 3.090	91.241 ± 0.501
JNB(2)	0.671 ± 0.044	85.205 ± 0.591	80.559 ± 1.374	92.882 ± 1.126	91.306 ± 0.478
JNB(3)	0.646 ± 0.057	84.828 ± 0.761	80.100 ± 1.787	92.808 ± 1.404	91.576 ± 0.416
JNB(4)	$\mathbf{0.559 \pm 0.062}$	$\mathbf{85.645 \pm 0.762}$	$\mathbf{83.395 \pm 3.549}$	89.498 ± 3.916	$\mathbf{92.497 \pm 0.404}$
JNB(5)	0.691 ± 0.100	84.247 ± 1.479	79.283 ± 3.401	93.190 ± 2.584	91.454 ± 0.825
JNB(6)	0.627 ± 0.107	84.455 ± 1.658	80.079 ± 4.017	92.371 ± 3.215	91.648 ± 0.985

Table 4: Binary classification metrics for the IMDB dataset using different activation functions.

Unveiling the Power of Wavelets: A Wavelet-based Kolmogorov-Arnold Network for Hyperspectral Image Classification [PDF] arxiv.org

ST Seydi - arXiv preprint arXiv:2406.07869, 2024 - arxiv.org

28 days ago - Hyperspectral image classification is a crucial but challenging task due to the high dimensionality and complex spatial-spectral correlations inherent in hyperspectral data …

☆ Save 99 Cite Cited by 1 Related articles »

TABLE I: Results on Different Datasets

Dataset	Model	Overall Accuracy	Kappa
Salinas	Spline-KAN	0.9261	0.9178
	Wav-KAN	**0.9341**	**0.9264**
	MLP	0.8655	0.8499
Pavia	Spline-KAN	0.9863	0.9806
	Wav-KAN	0.9901	0.9860
	MLP	**0.9910**	**0.9873**
Indian Pine	Spline-KAN	0.7731	0.7395
	Wav-KAN	**0.8554**	**0.8348**
	MLP	0.3513	0.2984

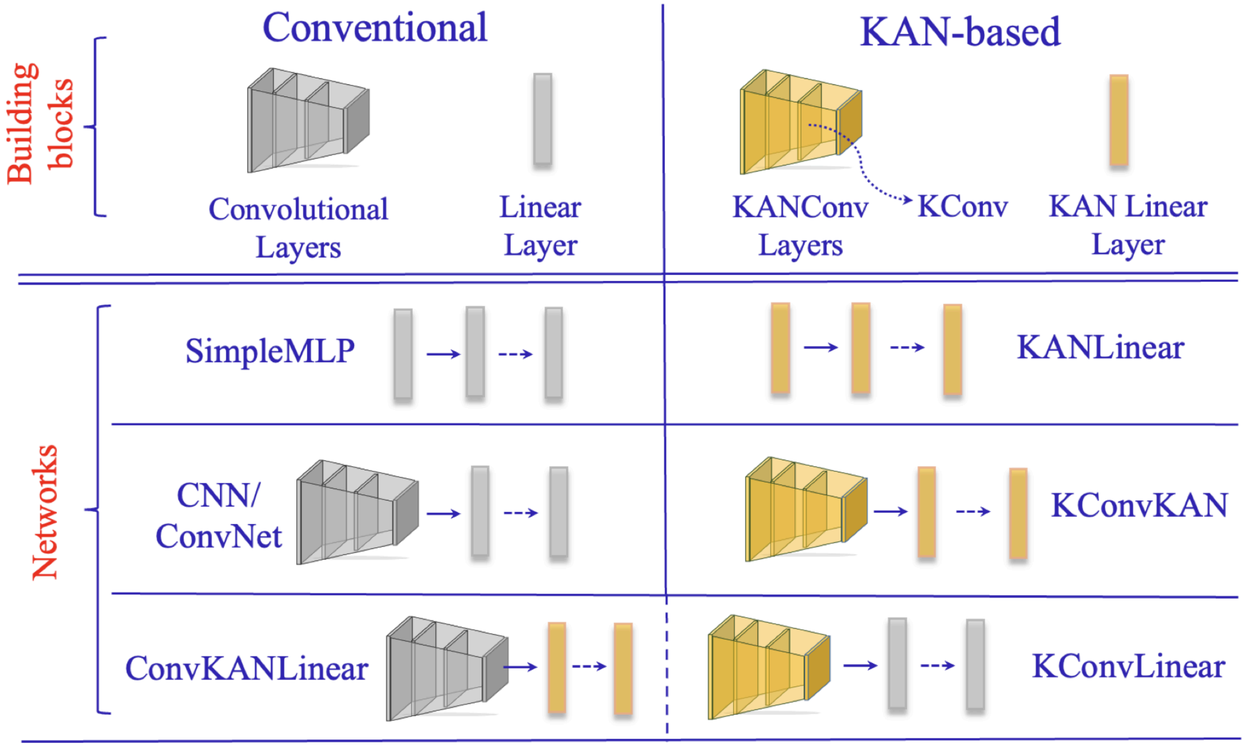
Suitability of KANs for Computer Vision: A preliminary investigation

[PDF] arxiv.org

B Azam, N Akhtar - arXiv preprint arXiv:2406.09087, 2024 - arxiv.org

27 days ago - Kolmogorov-Arnold Networks (KANs) introduce a paradigm of neural modeling that implements learnable functions on the edges of the networks, diverging from the …

☆ Save Cite Cited by 2 Related articles All 2 versions »



$$\text{ConvKAN Kernel} = \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix}$$

$$y_{i,j} = \sum_{m,n} \phi_{m,n}(x_{i+m,j+n}),$$

Fig. 1. Categorization of the types of network architectures used in this work. We employ KAN-based building blocks with conventional layers to construct different types of networks. The same naming conventions are used throughout this work.

TABLE IV
OVERALL PERFORMANCE FOR DIFFERENT MODELS ON MNIST AND CIFAR-10.

Model	MNIST (%)				CIFAR-10 (%)			
	Accuracy	Precision	Recall	F1 Score	Accuracy	Precision	Recall	F1 Score
SimpleMLP	92.4	92.4	92.3	92.3	39.1	39.5	39.4	39.1
ConvNet (Small)	98.4	98.4	92.3	92.3	56.2	55.9	56.2	55.7
ConvNet (Medium)	99.1	99.1	99.1	99.1	64.2	64.4	64.2	64.2
ConvNet (Large)	99.4	99.4	99.4	99.4	71.0	71.3	71.0	70.8
ConvKANLinear	98.5	98.5	98.5	98.5	61.6	61.5	61.6	61.4
KConvLinear	98.3	98.3	98.3	98.3	59.3	59.3	59.3	59.2
KConvKAN (2 Layers)	98.8	98.8	98.8	98.8	62.6	62.4	62.6	62.3
KConvKAN (8 Layers)	99.6	99.6	99.5	99.6	78.8	78.6	78.5	78.6
WavKan (2 Layers)	98.8	98.8	98.7	98.8	64.4	64.4	66.3	66.1
WavKan (8 Layers)	99.6	99.6	99.5	99.6	79.7	79.4	79.5	79.4

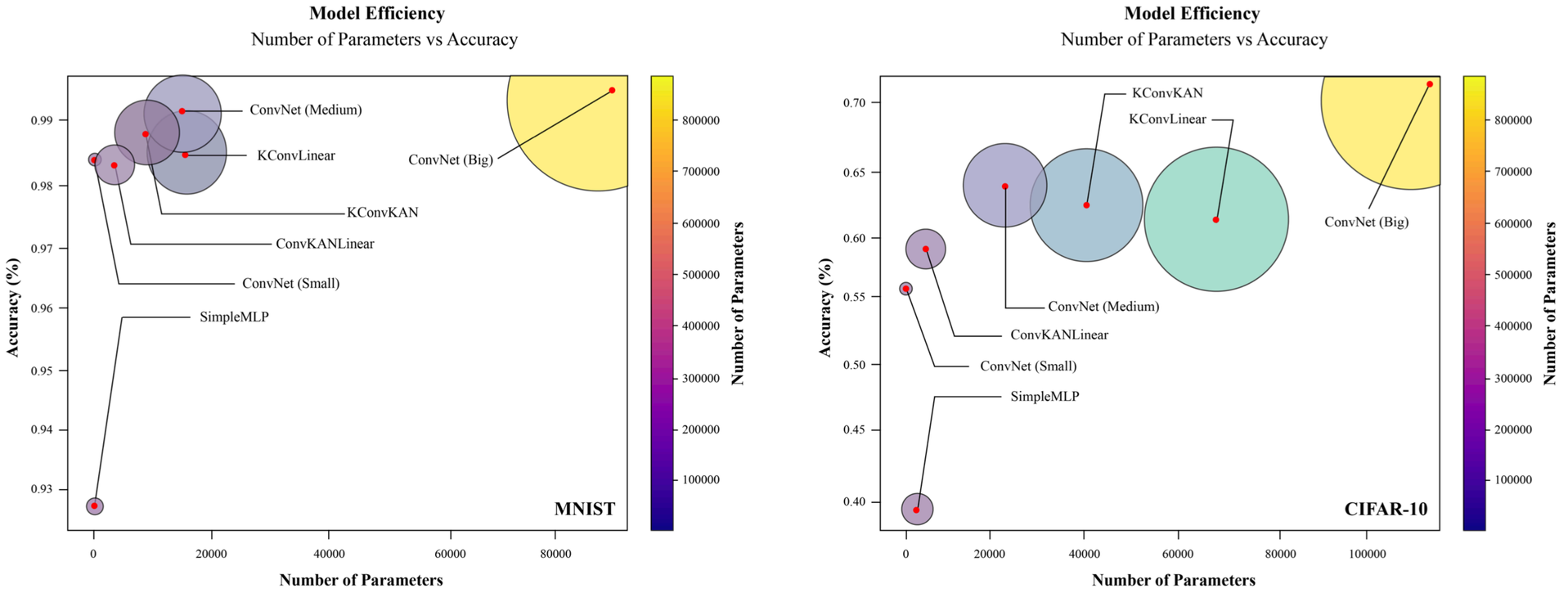


Fig. 4. Comparative performance visualization of different models on MNIST and CIFAR-10 datasets. The sizes and colors of the circles represent the scale of model parameters and performance metrics respectively, showcasing a range of outcomes from simple to complex models.

TABLE V
MODEL PERFORMANCE AND PARAMETERS

Dataset	Model	Accuracy (%)	Parameters	Epochs
MNIST	SimpleMLP	92.4	7,850	10
	ConvNet (Small)	98.4	2,740	
	ConvNet (Medium)	99.1	157,030	
	ConvNet (Large)	99.4	887,530	
	ConvKANLinear	98.5	163,726	
	KConvLinear	98.3	37,030	
	KConvKAN	98.8	94,650	
MNIST	KConvKAN-8	99.6	40,694,018	150
	WavKAN-2	98.8	951,370	
	WavKAN-8	99.6	10731562	
CIFAR-10	SimpleMLP	39.5	30.730	10
	ConvNet (Small)	56.2	3,580	
	ConvNet (Medium)	64.2	224,495	
	ConNet (Large)	71.0	1,134,890	
	ConvKANLinear	61.6	694,926	
	KConvLinear	59.3	48,370	
	KConvKAN	62.6	405,900	
CIFAR-10	KConvKAN-8	78.8	40,695,584	150
	WavKAN-2	64.4	952,650	
	WavKAN-8	79.7	10,732,202	

The variation in epochs across models is intended to investigate the effect of more extensive training durations on performance of KAN based models.

TABLE VI
EVALUATION TIME IN COMPARISON WITH ACCURACIES

Model	Dataset	Accuracy	Train Time	Eval. Time
KConvKAN-8	MNIST	99.6%	1hr-55min	4.75 sec
WavKAN-2		98.8%	3hr-33min	5.72 sec
WavKAN-8		99.6%	8hr-40min	13.76 sec
KConvKAN-8	CIFAR-10	78.8%	2hr-05min	5.20 sec
WavKAN-2		64.4%	3hr-45min	7.81 sec
WavKAN-8		79.7%	8hr-55min	12.92 sec

The batch size used for WavKAN and WavKAN8 is 64, and for KConvKAN-8 it is 128, consistent across both datasets.

SCKansformer: Fine-Grained Classification of Bone Marrow Cells via Kansformer Backbone and Hierarchical Attention Mechanisms

[PDF] arxiv.org

Y Chen, Z Zhu, S Zhu, L Qiu, B Zou, F Jia, Y Zhu… - arXiv preprint arXiv …, 2024 - arxiv.org

28 days ago - The incidence and mortality rates of malignant tumors, such as acute leukemia, have risen significantly. Clinically, hospitals rely on cytological examination of …

☆ Save Cite Related articles »

***Abstract*— The incidence and mortality rates of malignant tumors, such as acute leukemia, have risen significantly. Clinically, hospitals rely on cytological examination of peripheral blood and bone marrow smears to diagnose malignant tumors, with accurate blood cell counting being crucial. Existing automated methods face challenges such as low feature expression capability, poor interpretability, and redundant feature extraction when processing high-dimensional microimage data. We propose a novel fine-grained classification model, SCKansformer, for bone marrow blood cells, which addresses these challenges and enhances classification accuracy and efficiency. The model integrates the Kansformer Encoder, SCConv Encoder, and Global-Local Attention Encoder. The Kansformer Encoder replaces the traditional MLP layer with the KAN, improving nonlinear feature representation and interpretability. The SCConv Encoder, with its Spatial and Channel Reconstruction Units, enhances feature representation and reduces redundancy. The Global-Local Attention Encoder combines Multi-head Self-Attention with a Local Part module to capture both global and local features. We validated our model using the Bone Marrow Blood Cell Fine-Grained Classification Dataset (BMCD-FGCD), comprising over 10,000 samples and nearly 40 classifications, developed with a partner hospital. Comparative experiments on our private dataset, as well as the publicly available PBC and ALL-IDB datasets, demonstrate that SCKansformer outperforms both typical and advanced microcell**

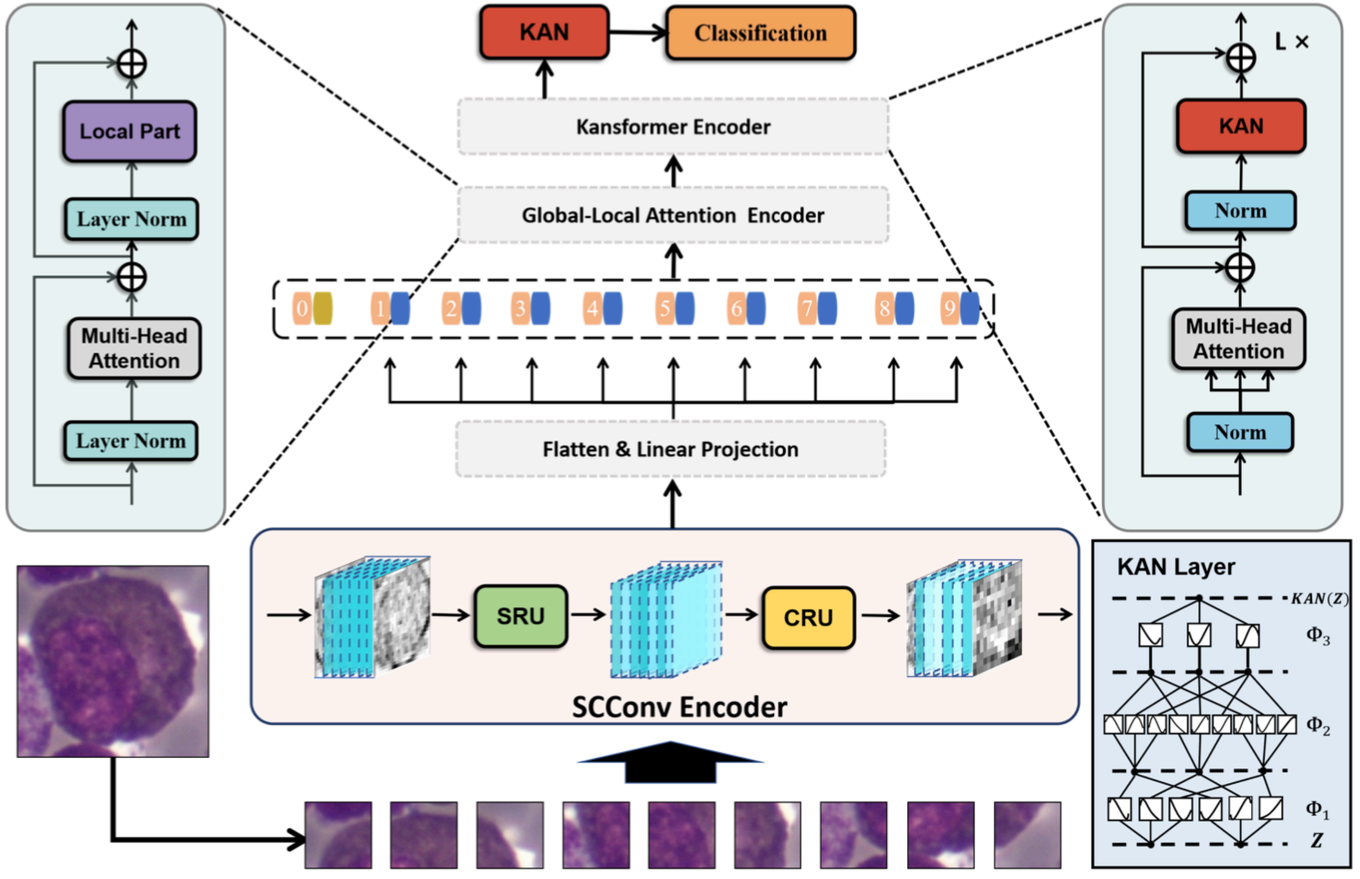


Fig. 1: The overall architecture of our proposed SCKansformer model. The SCKansformer model primarily comprises three parts: Kansformer Encoder, SCConv Encoder and Global-Local Attention Encoder.

TABLE I: Comparison of SCKansformer with other state-of-the-art models on the private BMCD-FGCD dataset.

Method	Precision	Recall	F1-score	Accuracy
VGG16 [23]	74.68	74.43	74.55	74.50
VGG19 [23]	73.53	73.35	73.44	73.26
ResNet-50 [24]	75.62	75.49	75.55	75.48
ResNet-101 [24]	75.43	75.20	75.31	75.21
WBCsNet [27]	73.65	72.04	72.84	72.43
BloodCaps [28]	78.72	76.74	77.94	78.31
ViT [25]	77.65	77.41	77.53	77.42
EfficientV2 [26]	77.37	77.21	77.29	77.20
WBC-GLAformer [30]	79.82	79.38	79.60	79.45
WBC YOLO-ViT [29]	78.92	79.31	79.67	78.47
SCKansformer(Ours)	**85.82**	**84.28**	**84.34**	**83.23**

Featureless point cloud upsampling via patched-based attention

[PDF] lut.fi

ANR Andriamananjara - 2024 - lutpub.lut.fi

27 days ago - Point clouds are a fundamental element of 3D technology, such as self-driving cars, 3D scanners, and many more. Working on it is risky, raw inputs from sensors like …

☆ Save Cite Related articles »

KINN

Kolmogorov Arnold Informed neural network: A physics-informed deep learning framework for solving PDEs based on Kolmogorov Arnold Networks [PDF] arxiv.org

Y Wang, J Sun, J Bai, C Anitescu, MS Eshaghi… - arXiv preprint arXiv …, 2024 - arxiv.org

26 days ago - AI for partial differential equations (PDEs) has garnered significant attention, particularly with the emergence of Physics-informed neural networks (PINNs). The recent …

☆ Save 99 Cite Cited by 4 Related articles »

Abstract

AI for partial differential equations (PDEs) has garnered significant attention, particularly with the emergence of Physics-informed neural networks (PINNs). The recent advent of Kolmogorov-Arnold Network (KAN) indicates that there is potential to revisit and enhance the previously MLP-based PINNs. Compared to MLPs, KANs offer interpretability and require fewer parameters. PDEs can be described in various forms, such as strong form, energy form, and inverse form. While mathematically equivalent, these forms are not computationally equivalent, making the exploration of different PDE formulations significant in computational physics. Thus, we propose different PDE forms based on KAN instead of MLP, termed Kolmogorov-Arnold-Informed Neural Network (KINN). We systematically compare MLP and KAN in various numerical examples of PDEs, including multi-scale, singularity, stress concentration, nonlinear hyperelasticity, heterogeneous, and complex geometry problems. Our results demonstrate that KINN significantly outperforms MLP in terms of accuracy and convergence speed for numerous PDEs in computational solid mechanics, except for the complex geometry problem. This highlights KINN's potential for more efficient and accurate PDE solutions in AI for PDEs.

Keywords: PINNs, Kolmogorov–Arnold Networks, Computational mechanics, AI for PDEs, AI for science

KINN

3.2. KINN

Fig. 1 illustrates the core idea of KINN [1], which uses KAN instead of MLP in the different forms of PDEs (strong, energy, and inverse form). Notably, as the output range can easily exceed the grid range in B-splines after a single KAN layer (we must set the grid range in advance), we apply a tanh activation function to the output of the KAN layer to constrain it within $[-1, 1]$ and better utilize the capabilities of the B-splines in this range. Thus, the output of each KAN layer undergoes an additional tanh activation:

$$\boldsymbol{Y}^{\text{new}} = \tanh(\boldsymbol{Y}) = \tanh\left\{[\sum_{\text{column}} \boldsymbol{\phi}(\boldsymbol{X}) \odot \boldsymbol{S}] + \boldsymbol{W} \cdot \sigma(\boldsymbol{X})\right\}. \tag{30}$$

However, we do not apply tanh to the final layer since the output does not necessarily lie within $[-1, 1]$. If the simulation domain is outside the $[-1, 1]$ grid range, we perform a scale normalization for input $\boldsymbol{X}$. For instance, we can find the smallest bounding box enclosing the geometry to be simulated, denoted as $[L, W, X_c, Y_c]$ as shown in Fig. 1, and then normalize the input as follows:

$$x^s = \frac{x - X_c}{L/2}; \quad y^s = \frac{y - Y_c}{W/2}. \tag{31}$$

KINN

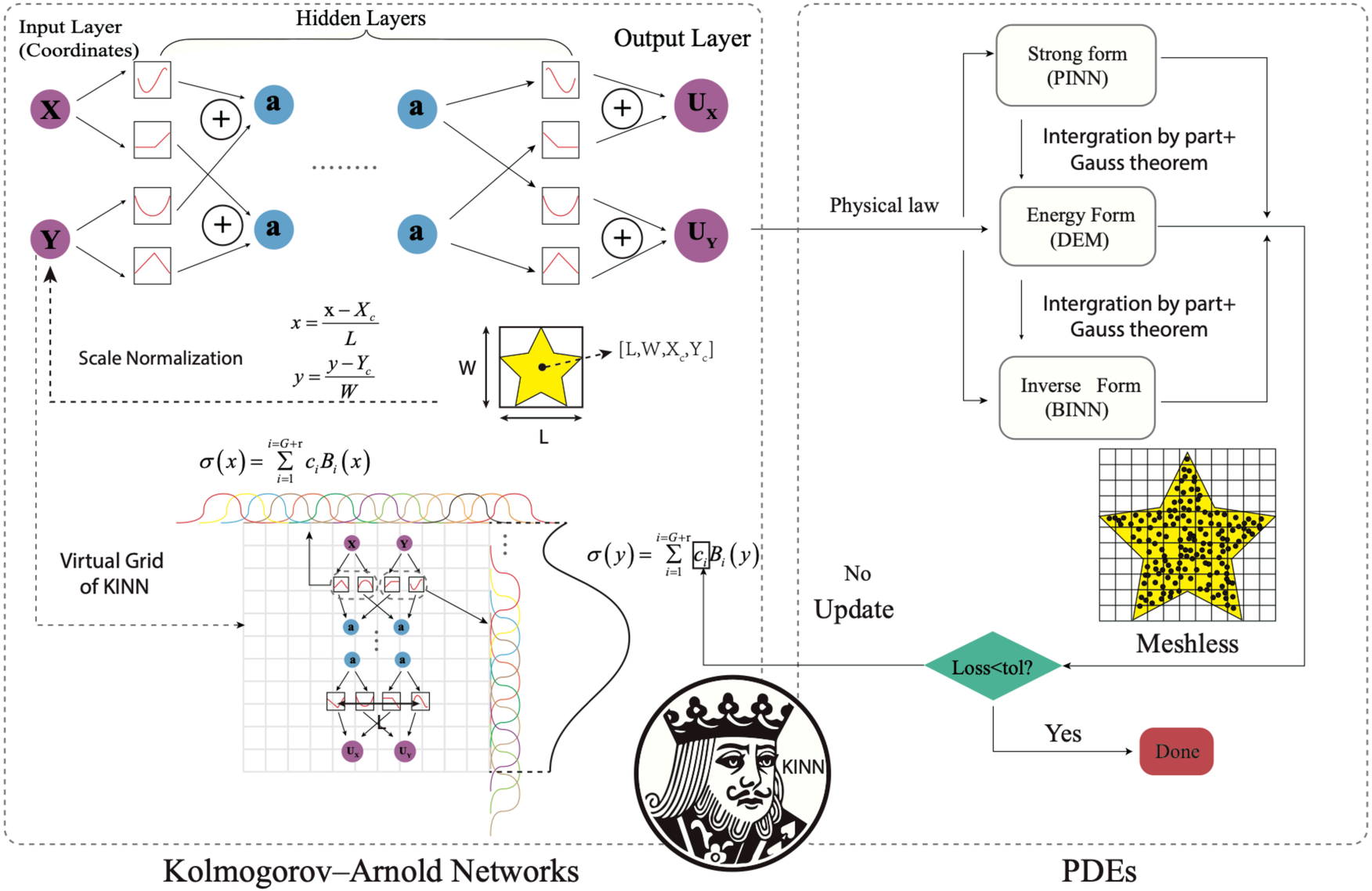
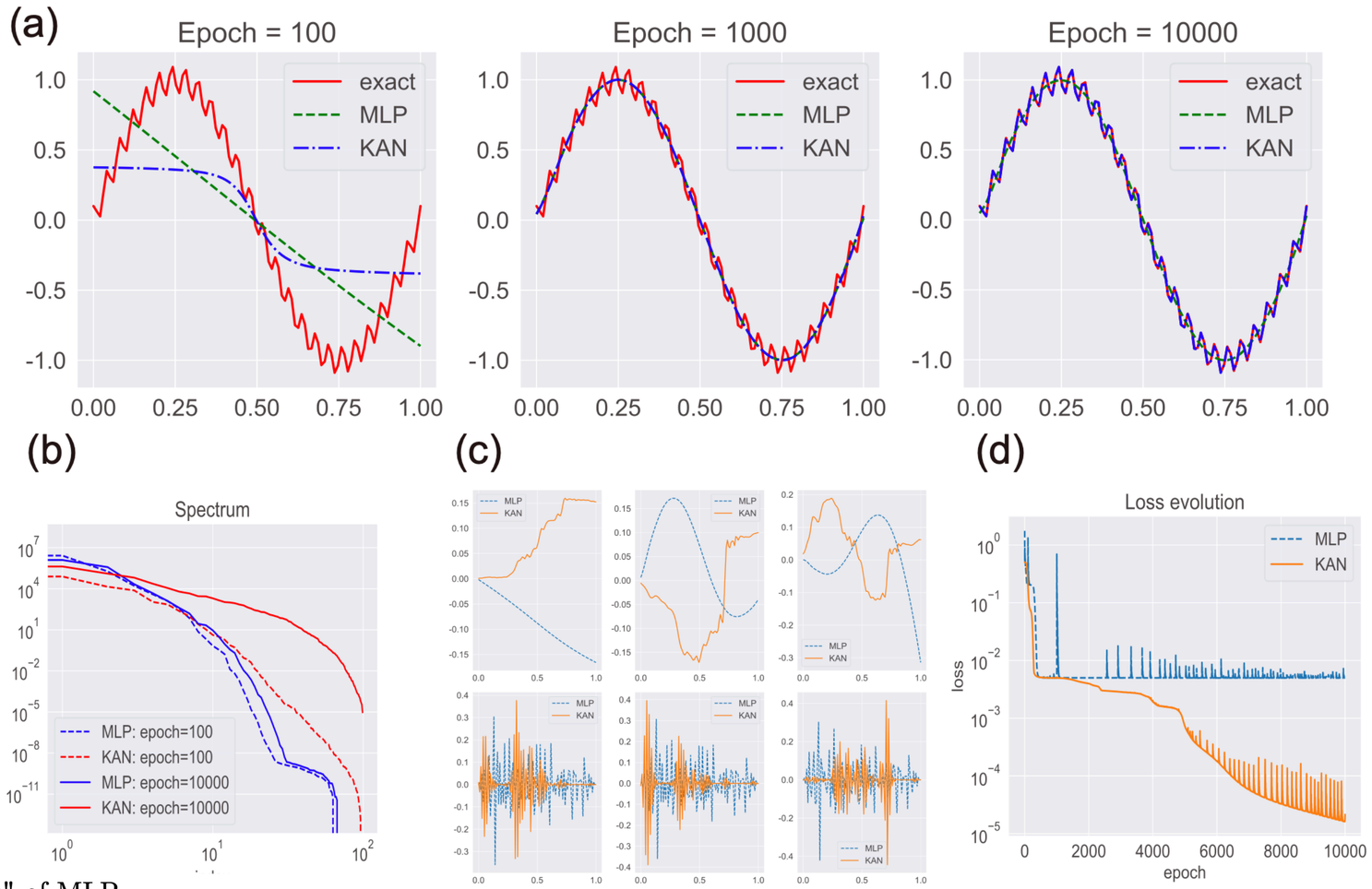


Fig. 1. Schematic of KINN: The idea of KINN is to replace MLP with Kolmogorov–Arnold Networks (KAN) in different PDEs forms (strong, energy, and inverse forms). In KAN, the main training parameters are the undetermined coefficients c_i of the B-splines in the activation function. KINN establishes the loss function based on different numerical formats of PDEs and optimizes the c_i. The virtual Grid is determined by the grid size in KAN.

KINN



Naturally, we ask whether KAN can fundamentally solve the "spectral bias" of MLP.
To validate our hypothesis, we consider the following 1D Poisson problem and its analytical solution:

$$\begin{cases} \frac{\partial^2 u(x)}{\partial x^2} = -4\pi^2 \sin(2\pi x) - 250\pi^2 \sin(50\pi x), \quad x \in [0,1] \\ u(0) = u(1) = 0 \\ u = \sin(2\pi x) + 0.1 \sin(50\pi x) \end{cases} \tag{41}$$

of MLP and KAN in fitting $u = \sin(2\pi x) + 0.1\sin(50\pi x)$. (a) Exact solutions, hs 100, 1000, and 10000, (b) Eigenvalue distributions of MLP and KAN, (c) The value sorted from largest to smallest, and the second row is the eigenvectors of the loss functions of MLP and KAN.

To simplify, we first consider function fitting. If function fitting fails, solving PDEs will certainly fail. To objectively validate, we solve the same problem as in [42], referencing the code at https://github.com/PredictiveIntelligenceLab/MultiscalePINNs. We fit $u = \sin(2\pi x)+0.1\sin(50\pi x)$ using MLP and KAN. As shown in Fig. 2a, we uniformly distribute 100 points in $[0,1]$. The structure of KAN is [1,5,5,5,1] while the structure of MLP is [1,100,100,100,100,1]. We set the same learning rate (0.001) and optimizer (Adam) for objective

KINN

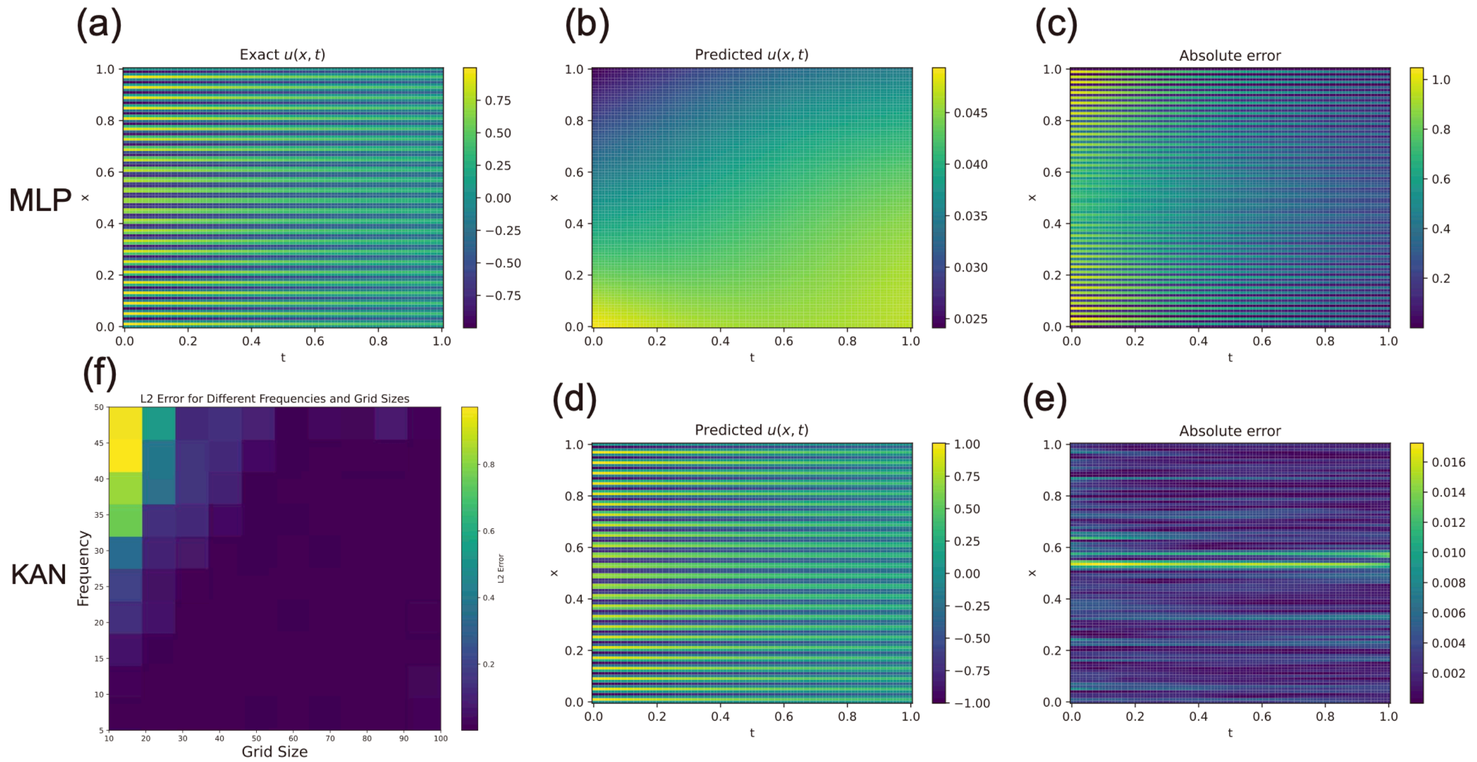
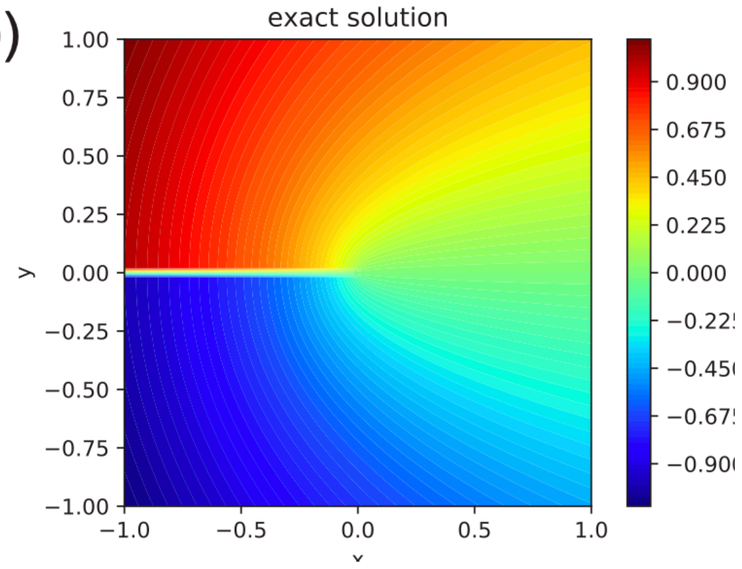


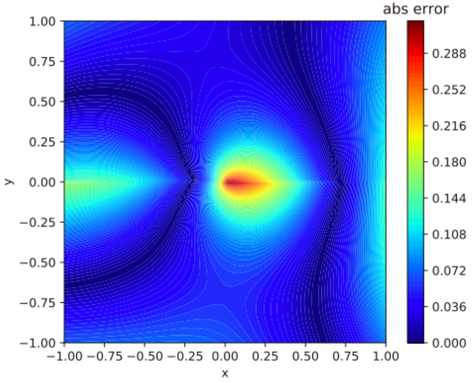
Fig. 3. MLP and KAN fitting the high and low-frequency mixed heat conduction problem. (a) Exact solution for frequency $F = 50$. (b) MLP prediction for frequency $F = 50$ after 5000 iterations. (c) The absolute error of MLP for frequency $F = 50$ after 5000 iterations. (d) KAN prediction for frequency $F = 50$ after 5000 iterations. (e) The absolute error of KAN for frequency $F = 50$ after 5000 iterations. (f) Relative error of KAN under different grid sizes and frequencies, with 3000 iterations and network structure [2,5,1].

$$
\begin{aligned}
u_t &= \frac{1}{(F\pi)^2} u_{xx}, \quad x \in [0,1], t \in [0,1] \\
u(x,0) &= \sin(F\pi x), \quad x \in [0,1] \\
u(0,t) &= u(1,t) = 0, \quad t \in [0,1]
\end{aligned}
\tag{42}
$$

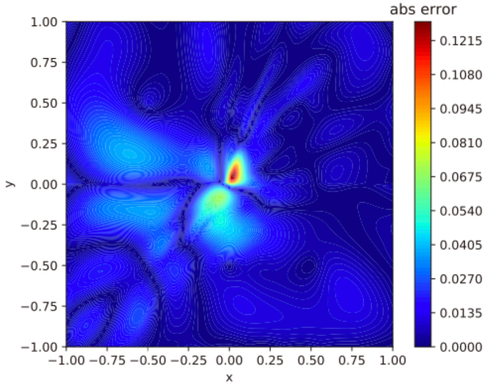
KINN

(b)

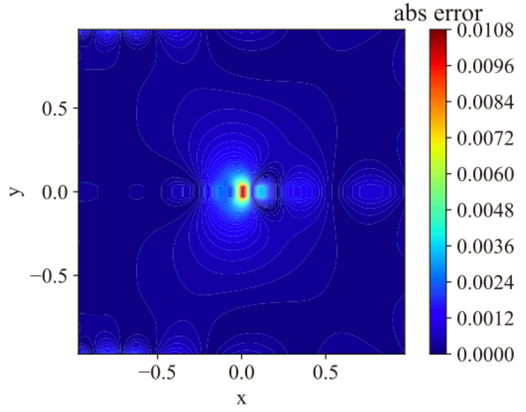




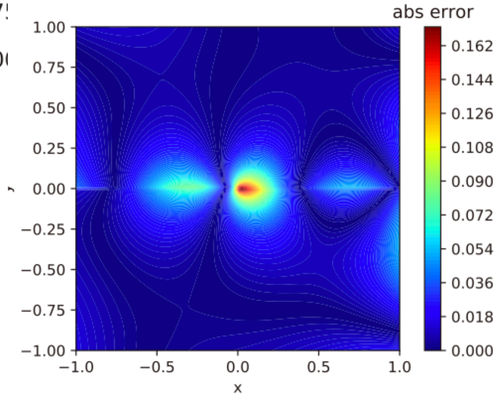
(a) CPINN_MLP



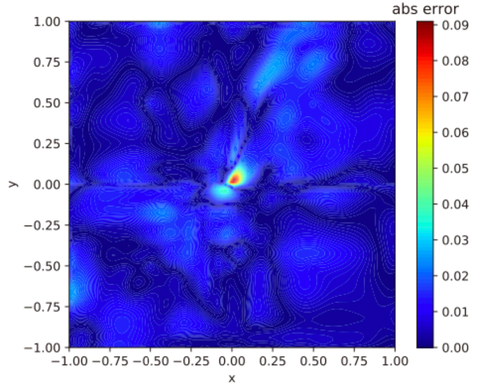
(b) DEM_MLP



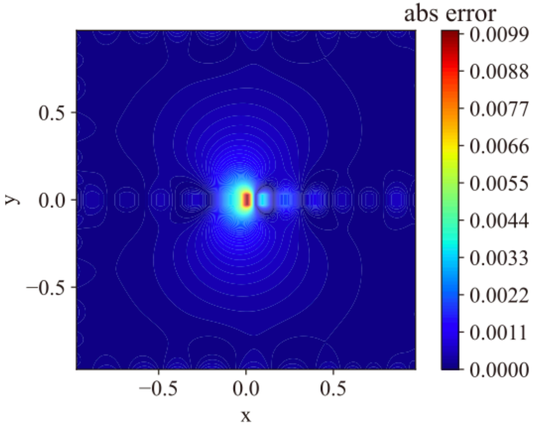
(c) BINN_MLP



(d) KINN_CPINNs

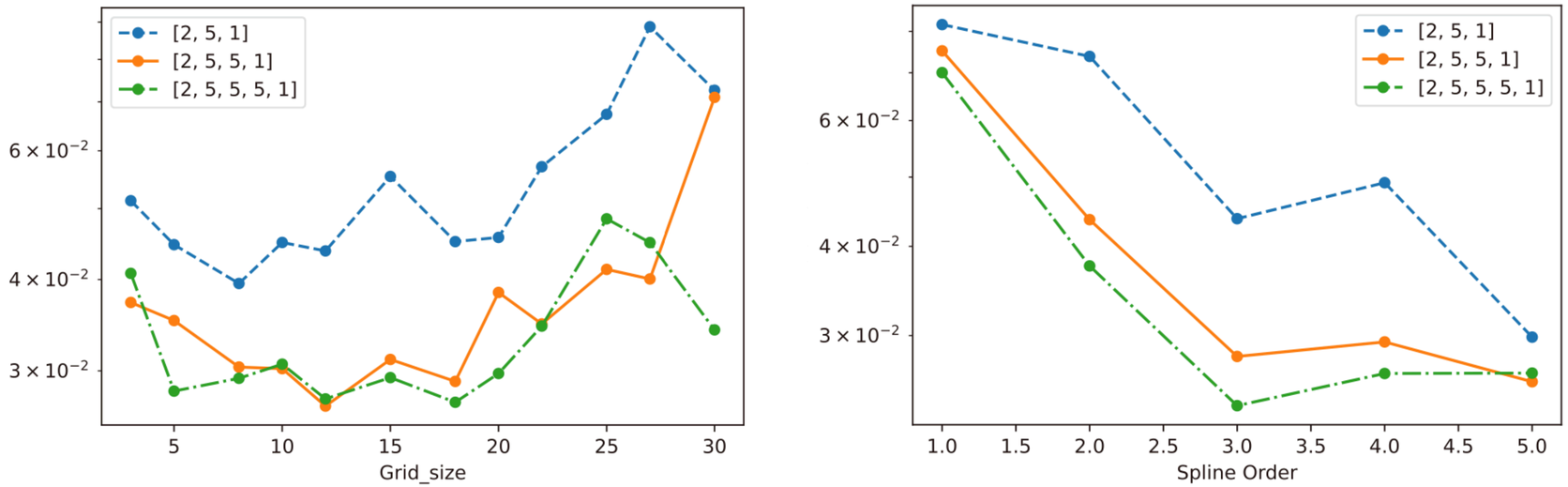


(e) KINN_DEM



(f) KINN_BINN

KINN



(a) Grid size

(b) Spline order

Fig. 8. Relative $\mathcal{L}_2$ error for KINN-CPINNs in mode III crack in different architectures of KAN with different grid sizes (a) and different B-spline orders (b).

BSRBF-KAN

BSRBF-KAN: A combination of B-splines and Radial Basic Functions in Kolmogorov-Arnold Networks [PDF] arxiv.org

HT Ta - arXiv preprint arXiv:2406.11173, 2024 - arxiv.org

24 days ago - In this paper, we introduce BSRBF-KAN, a Kolmogorov Arnold Network (KAN) that combines Bsplines and radial basis functions (RBFs) to fit input vectors in data training …

☆ Save 99 Cite Cited by 1 Related articles »

$$\phi(x) = w_b b(x) + w_s(\phi_{BS}(x) + \phi_{RBF}(x)) \tag{11}$$

where $b(x)$ and w_b are the base output and its base matrix. $\phi_{BS}(x)$ and ϕ_{RBF} are B-Spline and RBF functions, and w_s is the spline matrix. The design of BSRBF-KAN is similar to the original KAN, mentioned in Section 3.2. Given the goal of designing a combined KAN, our intuition understands that the forward and backward speeds will not surpass those of individual KANs.

$$\phi(x) = w_b b(x) + w_s spline(x)$$

$$RBF(\mathbf{x}) = \sum_{i=1}^{N} w_i \phi(r_i) = \sum_{i=1}^{N} w_i \exp\left(-\frac{1}{2}\left(\frac{\mathbf{x} - \mathbf{c}_i}{h}\right)^2\right)$$

BSRBF-KAN

Table 2: The average metric values in 5 training sessions on MNIST and Fashion-MNIST.

Dataset	Model	Train. Acc.	Val. Acc.	F1	Time (seconds)
MNIST	BSRBF-KAN	**100.00 ± 0.00**	97.55 ± 0.03	97.51 ± 0.03	231
	FastKAN	99.94 ± 0.01	97.25 ± 0.03	97.21 ± 0.03	101
	FasterKAN	98.48 ± 0.01	97.28 ± 0.06	97.25 ± 0.06	**93**
	EfficientKAN	99.37 ± 0.04	97.37 ± 0.07	97.33 ± 0.07	120
	GottliebKAN	98.44 ± 0.61	97.19 ± 0.22	97.14 ± 0.23	221
	MLP	99.44 ± 0.01	**97.62 ± 0.03**	**97.59 ± 0.03**	181
Fashion-MNIST	BSRBF-KAN	**99.19 ± 0.03**	89.33 ± 0.07	89.29 ± 0.07	211
	FastKAN	98.19 ± 0.04	**89.42 ± 0.07**	**89.38 ± 0.07**	162
	FasterKAN	94.40 ± 0.01	89.26 ± 0.06	89.17 ± 0.07	154
	EfficientKAN	94.76 ± 0.06	88.92 ± 0.08	88.85 ± 0.09	183
	GottliebKAN	90.66 ± 1.08	87.16 ± 0.24	87.07 ± 0.25	238
	MLP	93.56 ± 0.05	88.39 ± 0.06	88.36 ± 0.05	**148**
Average of MNIST + Fashion-MNIST	BSRBF-KAN	**99.60**	**93.44**	**93.40**	221
	FastKAN	99.07	93.34	93.30	131.5
	FasterKAN	96.44	93.27	93.21	123.5
	EfficientKAN	97.07	93.15	93.09	151.5
	GottliebKAN	94.55	92.18	92.11	229.5
	MLP	96.50	93.01	92.98	164.5

Train. Acc = Training Accuracy, Val. Acc. = Validation Accuracy

GraphKAN

GraphKAN: Enhancing Feature Extraction with Graph Kolmogorov Arnold Networks [PDF] arxiv.org

F Zhang, X Zhang - arXiv preprint arXiv:2406.13597, 2024 - arxiv.org

1) **Aggregates** information from the neighbors. The message function is used to aggregate the neighboring features of the target node, including the target node's own features h_v^t, the features of its neighboring nodes h_w^t, and the edge features connecting it to its neighboring nodes e_{vw}^t. This aggregation forms a message vector m_v^t that is then passed to the target node. The formula is as follows:

$$m_v^{t+1} = \sum_{w \in N(v)} M^t(h_v^t, h_w^t, e_{vw}^t) \tag{4}$$

where m_v^{t+1} is the information received by the node in the next layer $t+1$, M^t is the message function, h_v^t represents the node features in the current layer, $N(v)$ represents the set of neighboring nodes for a node v, h_w^t represents the node features of the neighboring nodes in the current layer, and e_{vw}^t represents the edge features from node to node.

2) **Extracts** the node representation. The node update function is used to update the node features of the next layer, combining the features of the current layer's nodes and the messages obtained from the message function. The formula is as follows:

$$h_v^{t+1} = U^t(h_v^t, m_v^{t+1}) \tag{5}$$

where U^t is the node update function, which takes the original node state and the message as inputs and generates the new node state.

$$h_v^{t+1} = KAN^t(h_v^t, m_v^{t+1}) = \Phi^t(h_v^t, m_v^{t+1}).$$

GraphKAN

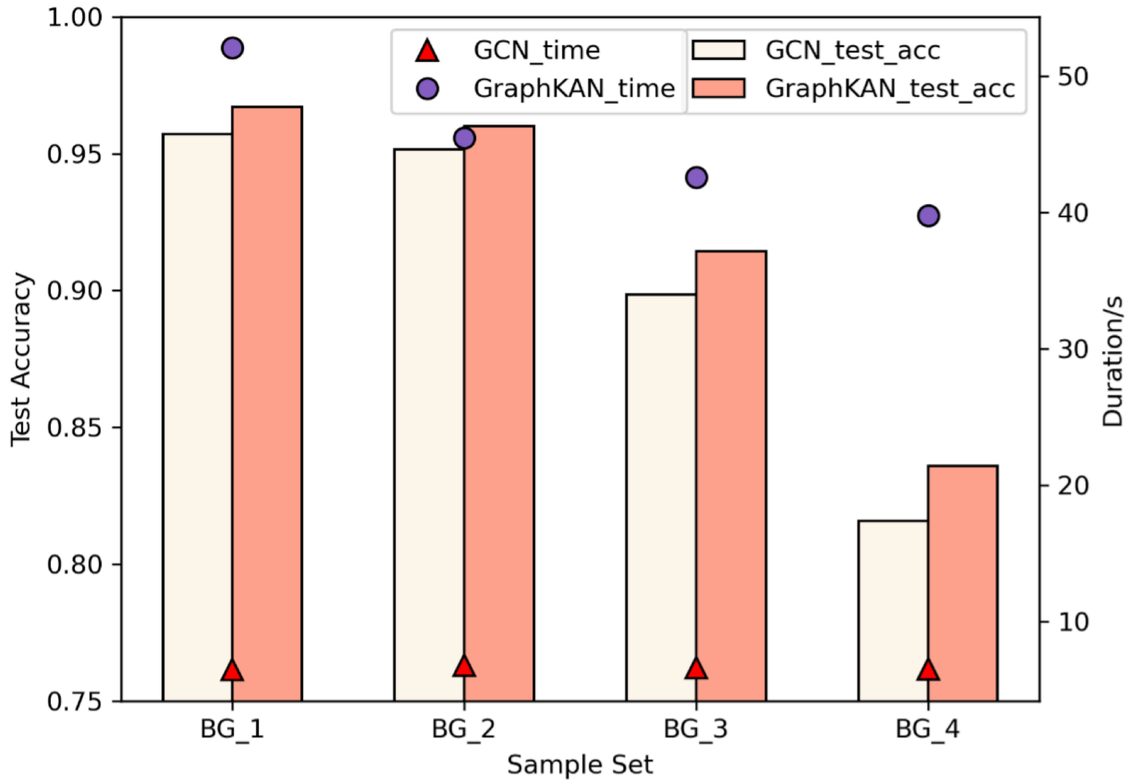


Figure 1: Comparison on testing accuracy and time consumption

Convolutional KAN

Convolutional Kolmogorov-Arnold Networks [PDF] arxiv.org

AD Bodner, AS Tepsich, JN Spolski… - arXiv preprint arXiv …, 2024 - arxiv.org

23 days ago - In this paper, we introduce the Convolutional Kolmogorov-Arnold Networks (Convolutional KANs), an innovative alternative to the standard Convolutional Neural …

☆ Save 99 Cite Cited by 1 Related articles »

$$\text{Image} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1p} \\ a_{21} & a_{22} & \cdots & a_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mp} \end{bmatrix}$$

$$\text{KAN Kernel} = \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix}$$

$$(\text{Image} * K)_{i,j} = \sum_{k=1}^{N} \sum_{l=1}^{M} \phi_{kl}(a_{i+k,j+l})$$

$$\text{Image} * \text{KAN Kernel} = \begin{bmatrix} \phi_{11}(a_{11}) + \phi_{12}(a_{12}) + \phi_{21}(a_{21}) + \phi_{22}(a_{22}) & \cdots & r_{1(p-1)} \\ \phi_{11}(a_{21}) + \phi_{12}(a_{22}) + \phi_{21}(a_{31}) + \phi_{22}(a_{32}) & \cdots & r_{2(p-1)} \\ \vdots & \ddots & \vdots \\ \phi_{11}(a_{m1}) + \phi_{12}(a_{m2}) + \phi_{21}(a_{(m+1)1}) + \phi_{22}(a_{(m+1)2}) & \cdots & r_{m(p-1)} \end{bmatrix}$$

Convolutional KAN

Model	Accuracy	Precision	Recall	F1 Score	#Params	Minutes per epoch
MNIST Dataset						
KKAN (Small)	98.90%	98.90%	98.89%	98.90%	94.875K	1.8119
Conv & KAN	98.75%	98.75%	98.75%	98.75%	95K	0.4551
KAN Conv & 1 Layer MLP	98.53%	98.54%	98.53%	98.53%	7.4K	1.7867
KAN Conv & 2 Layer MLP	98.58%	98.58%	98.58%	98.58%	164K	1.7779
KAN Conv BN & 2 Layer MLP	98.53%	98.54%	98.53%	98.53%	164K	1.7545
CNN (Medium)	99.12%	99.12%	99.12%	99.12%	157K	0.3110
CNN (Small)	97.59%	97.60%	97.59%	97.59%	2.7K	0.3056
1 Layer MLP	92.34%	92.22%	92.22%	92.22%	7.9K	0.2935

Table 1: Comparison of accuracy, precision, recall, F1 score, parameter count, and training time per epoch for the proposed models tested on the MNIST Dataset. Convolutional KANs altenative KKAN (Small) shows an increase in accuracy of 0.15% against normal convolutions alternative with the same ammount of paramenters. Also KKAN with only $\sim$ 90k parameters achieves 0.22% less accuracy than the CNN (Medium) with 157k parameters.

A Benchmarking Study of Kolmogorov-Arnold Networks on Tabular Data [PDF] arxiv.org

E Poeta, F Giobergia, E Pastor, T Cerquitelli… - arXiv preprint arXiv …, 2024 - arxiv.org

22 days ago - Kolmogorov-Arnold Networks (KANs) have very recently been introduced into the world of machine learning, quickly capturing the attention of the entire community …

☆ Save 99 Cite Related articles »

TABLE I

DATASET CHARACTERISTICS. FOR EACH DATASET, WE REPORT THE CLASSIFICATION TYPE (BINARY OR MULTICLASS), THE NUMBER OF FEATURES, NUMBER OF ROWS, MISSING VALUES, AND DATA TYPES.

Dataset name	Task	Number of Features	Number Rows	Missing values	Data Type
Breast Cancer	Binary	30	569	✗	Numerical
Spam	Binary	47	4,601	✗	Numerical
Musk	Binary	166	6,598	✗	Numerical
Dry Bean	Multiclass	16	13,611	✗	Numerical
Gamma Telescope	Binary	10	19,020	✗	Numerical
Adult	Binary	14	48,842	✓	Numerical + Categorical
Shuttle	Multiclass	7	58,000	✗	Numerical
Diabetes	Binary	21	253,680	✓	Numerical
Poker	Multiclass	10	1,025,010	✗	Numerical

TABLE II

PERFOMANCE RESULTS OF KAN AND MLP ACROSS THE DATASETS. EACH METRIC IS EXPRESSED AS AVERAGE±STANDARD DEVIATION. THE SELECTED ACCURACY, F1-SCORE, PRECISION, AND RECALL ARE THE HIGHEST AMONG THE AVERAGE METRICS FROM FIVE RUNS ACROSS ALL PARAMETER CONFIGURATIONS. THE FPR AND FNR ARE CALCULATED FOR BINARY CLASSIFICATION DATASETS ONLY. THE TRAINING TIME IN SECONDS REFERS TO THE AVERAGE EXPIRED TIME IN TRAINING FOR EACH CONFIGURATION.

Dataset	Model	Accuracy	F1 score	Precision	Recall	FPR	FNR	Training time (s)
Breast Cancer	KAN	94.56±1.569	78.69±1.692	95.03±1.791	93.43±1.685	1.97±1.606	11.163±2.548	0.09
	MLP	**96.84±0.480**	**80.44±0.423**	**96.64±0.540**	**96.64±0.536**	2.53±0.630	4.18±1.040	0.06
Spam	KAN	**94.09±0.238**	78.31±0.245	**94.10±0.249**	**93.78±0.247**	4.18±0.309	8.26±0.421	0.69
	MLP	93.96±0.182	**79.19±0.18**8	93.93±0.180	93.69±0.195	4.52±0.133	8.11±0.292	0.38
Musk	KAN	**92.45±0.358**	**83.62±0.992**	93.71±1.492	89.86±1.943	1.53±0.397	18.76±3.571	0.97
	MLP	90.44±1.125	76.47±4.119	**97.05±0.577**	**91.07±1.223**	0.45±0.134	17.42±2.383	0.43
Dry Bean	KAN	**92.80±0.158**	**92.82±0.162**	**92.87±0.169**	**92.80±0.158**	-	-	1.92
	MLP	92.75±0.088	92.77±0.085	92.81±0.076	92.75±0.088	-	-	0.86
Gamma Telescope	KAN	**86.94±0.313**	**71.01±0.331**	**87.03±0.470**	**83.97±0.287**	5.91±0.487	26.15±0.463	2.74
	MLP	85.94 ±0.042	69.56±0.049	85.83±0.083	82.12±0.061	6.05±0.127	29.71±0.213	1.45
Adult	KAN	**85.93 ±0.061**	**80.24±0.231**	**82.71±0.249**	78.52±0.443	6.41±0.254	37.39±1.180	6.66
	MLP	85.72±0.119	79.87±0.313	82.47±0.070	**78.89±1.995**	6.40±0.419	36.57±1.295	3.47
Shuttle	KAN	**99.76±0.013**	**99.70±0.021**	**99.72±0.056**	**99.76±0.013**	-	-	8.39
	MLP	99.62±0.011	99.47±0.016	99.49±0.023	99.62±0.011	-	-	4.22
Diabetes	KAN	**86.80 ±0.028**	**58.38±0.728**	73.00±0.228	**56.69±0.503**	1.74±0.166	84.86±1.170	38.74
	MLP	86.73±0.194	55.92±3.860	**74.06±1.070**	55.19±2.198	1.26±0.579	88.35±4.972	20.34
Poker	KAN	**99.91±0.038**	**99.91±0.039**	**99.91±0.039**	**99.91±0.038**	-	-	34.76
	MLP	92.44±0.209	89.07±0.610	86.24±1.186	92.44±0.209	-	-	13.98

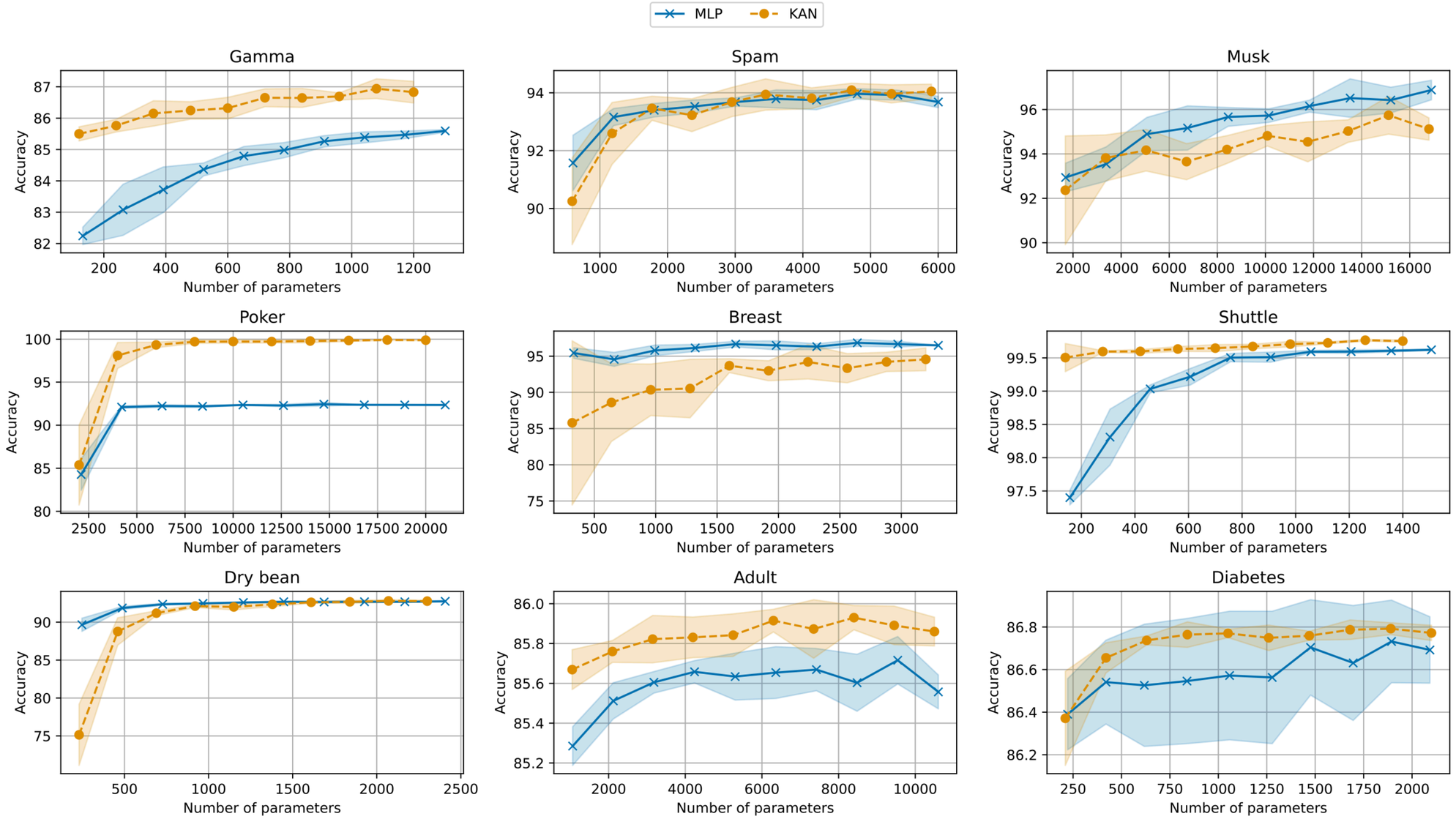


Fig. 1. Accuracy scores of KANs and MLPs as the number of parameters increases. For each dataset, we report the average accuracy across five runs for each parameter count.

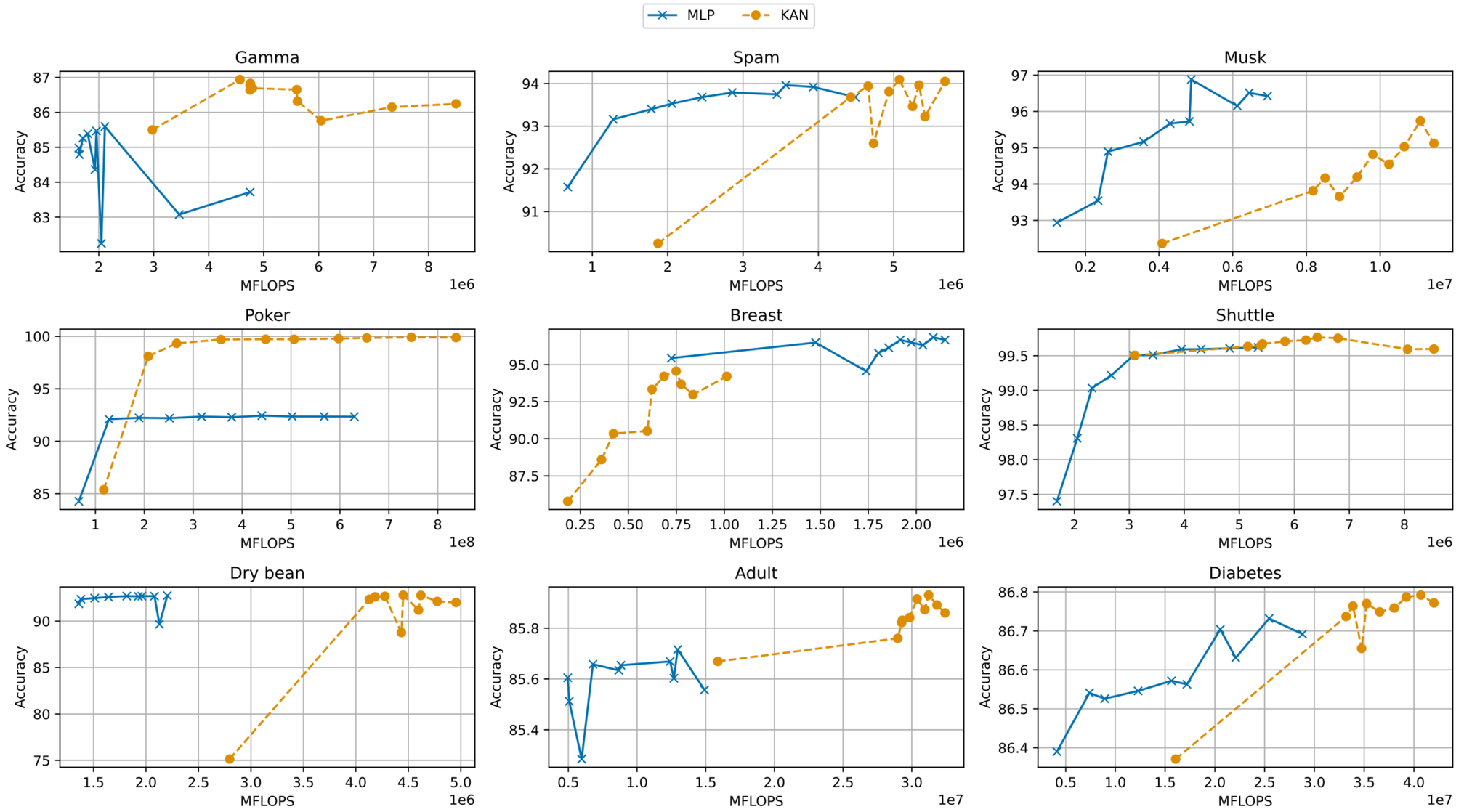


Fig. 2. Number of FLOPS. The number of FLOPS performed by each model as the number of parameters increases is reported in megaflops (MFLOPS).

rKAN

rKAN: Rational Kolmogorov-Arnold Networks [PDF] arxiv.org

AA Aghaei - arXiv preprint arXiv:2406.14495, 2024 - arxiv.org

22 days ago - The development of Kolmogorov-Arnold networks (KANs) marks a significant shift from traditional multi-layer perceptrons in deep learning. Initially, KANs employed B …

☆ Save 99 Cite Related articles »

Definition 1 (Mapped Jacobi function)**.** By applying an invertible mapping function $\varphi : \Omega \to [-1, 1]$ to the input of Jacobi polynomials, the mapped Jacobi functions can be generated as:

$$\mathcal{R}_n^{(\alpha,\beta)}(\xi) = \mathcal{J}_n^{(\alpha,\beta)}(\varphi(\xi)).$$

a Padé approximant of order $[q/k]$ for a function $F(\xi)$ is:

$$F^{[q/k]}(\xi) \approx \frac{A_q(\xi)}{B_k(\xi)} = \frac{\sum_{i=0}^{q} a_i \xi^i}{\sum_{j=0}^{k} b_j \xi^j},$$

$$R(x) = \frac{\sum_{j=0}^{m} a_j x^j}{1 + \sum_{k=1}^{n} b_k x^k} = \frac{a_0 + a_1 x + a_2 x^2 + \cdots + a_m x^m}{1 + b_1 x + b_2 x^2 + \cdots + b_n x^n},$$

which agrees with $f(x)$ to the highest possible order, which amounts to

$$\begin{aligned} f(0) &= R(0), \\ f'(0) &= R'(0), \\ f''(0) &= R''(0), \\ &\vdots \\ f^{(m+n)}(0) &= R^{(m+n)}(0). \end{aligned}$$

rKAN

$$\phi_{q,k}(\boldsymbol{\xi}) = \frac{\sum_{i=0}^{k} \theta_i^e \mathcal{R}_i^{(\alpha,\beta)}(\boldsymbol{\xi}_q)}{\sum_{i=0}^{p} \theta_i^d \mathcal{R}_i^{(\alpha,\beta)}(\boldsymbol{\xi}_q)}.$$

Padé-rKAN is defined as:

$$F(\boldsymbol{\xi}) = \sum_{k=1}^{K} \psi_k \left(\sum_{q=1}^{\nu} \phi_{q,k}(\sigma(\boldsymbol{\xi}_q)) \right),$$

rKAN

Act. Func.	Loss		Accuracy	
	Mean	Std.	Mean	Std.
Sigmoid	0.0611	0.0028	98.092	0.0937
Tanh	0.0322	0.0015	98.904	0.0695
ReLU	0.0256	0.0010	99.140	0.0434
fKAN(2) [1]	0.0252	0.0017	99.134	0.0484
fKAN(3) [1]	0.0224	0.0019	99.200	0.0787
fKAN(4) [1]	0.0217	0.0008	99.228	0.0515
fKAN(5) [1]	0.0249	0.0009	99.204	0.0467
fKAN(6) [1]	0.0290	0.0028	99.024	0.1198
rKAN(2)	0.0215	0.0012	99.268	0.0683
rKAN(3)	0.0222	0.0012	99.210	0.0464
rKAN(4)	0.0292	0.0006	99.060	0.0332
rKAN(5)	**0.0213**	0.0004	**99.293**	0.0597
rKAN(6)	0.0214	0.0027	99.218	0.0944

Table 4: Performance of different activation functions in a CNN for classifying MNIST dataset. It is observed that rKAN outperforms fKAN in certain cases.

Demonstrating the Efficacy of Kolmogorov-Arnold Networks in Vision Tasks [PDF] arxiv.org

M Cheon - arXiv preprint arXiv:2406.14916, 2024 - arxiv.org

21 days ago - In the realm of deep learning, the Kolmogorov-Arnold Network (KAN) has emerged as a potential alternative to multilayer projections (MLPs). However, its applicability …

☆ Save 99 Cite Related articles All 3 versions »

Model	GPU Usage (gb)	MNIST	CIFAR10	CIFAR100
KKAN (Small)	1.8119	**98.90%**	-	-
Conv & KAN	-	98.75%	-	-
MLP Mixer-5	1.5	-	60.26	34.81
ViT-10/4	14.7	-	57.53	30.80
ResNet-18	0.6	-	**8**6.29	**5**9.15
Ours	**above table**	98.16%	66.93%	35.49%

Table 4: Comparison of Models on MNIST, CIFAR10, and CIFAR100 Datasets

QCD Masterclass Lectures on Jet Physics and Machine Learning [PDF] arxiv.org

AJ Larkoski - arXiv preprint arXiv:2407.04897, 2024 - arxiv.org

20 days ago - These lectures were presented at the 2024 QCD Masterclass in Saint-Jacut-de-la-Mer, France. They introduce and review fundamental theorems and principles of machine …

☆ Save 99 Cite Related articles All 3 versions »

KANQAS: Kolmogorov Arnold Network for Quantum Architecture Search [PDF] arxiv.org

A Kundu, A Sarkar, A Sadhu - arXiv preprint arXiv:2406.17630, 2024 - arxiv.org

16 days ago - Quantum architecture search~(QAS) is a promising direction for optimization and automated design of quantum circuits towards quantum advantage. Recent techniques …

☆ Save 99 Cite Related articles All 2 versions »

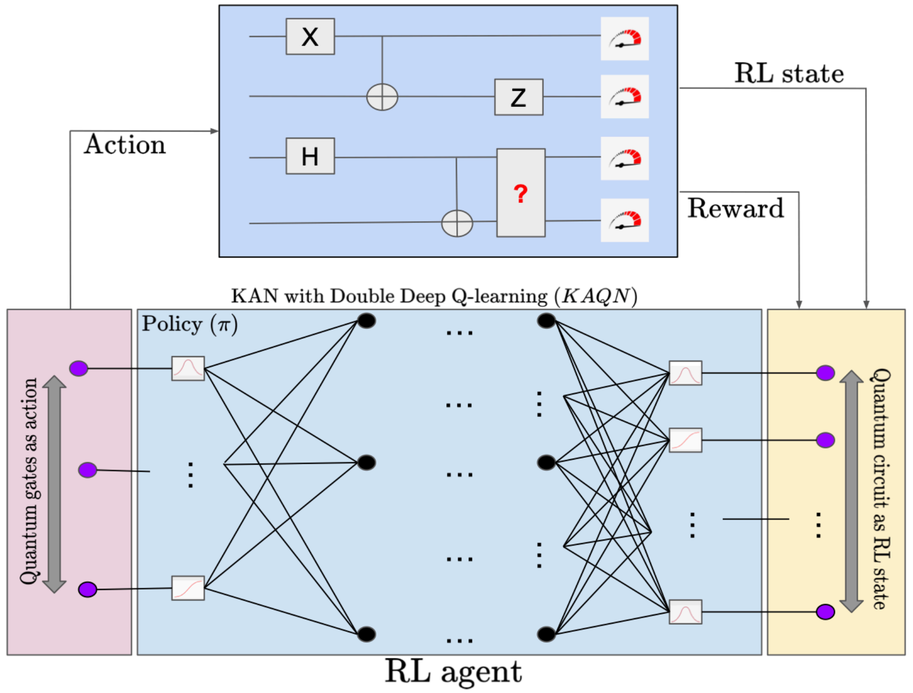
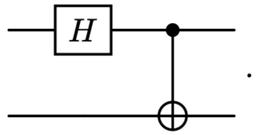


FIG. 1: The schematic for KANQAS where Kolmogorov Arnold network replaces the multi-layer perception structure in RL agent. The environment which is the quantum circuit interacts with the RL agent (containing the KAN) and gets updates after each step after the agent decides on an action following a policy.

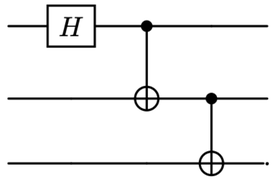
KANQAS[1]. As shown in Figure 1, the proposed framework of KANQAS features an RL agent, which contains the KAN, interacting with a quantum simulator. The agent sequentially generates output actions, which are candidates for quantum gates on the circuit. The fidelity of the state from the constructed circuit is compared to the quantum state fidelity of the circuit and is evaluated to determine how far it is from the desired goal. The reward, based on fidelity, is sent back to the RL agent. This process is repeated iteratively to train the RL agent. We show that in a noiseless scenario constructing Bell and GHZ state the probability of success and the number of optimal quantum circuit configurations to generate the states is significantly higher than MLPs. In a noisy scenario, we show that KAN can achieve a better fidelity in approximating GHZ state than MLPs, where the performance of the MLP significantly depends on the depth of the network and choice of activation function.

$$|\Phi^+\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right). \qquad |GHZ\rangle = \frac{1}{\sqrt{2}}\left(|000\rangle + |111\rangle\right)$$

a Bell state is given by



a GHZ state given by



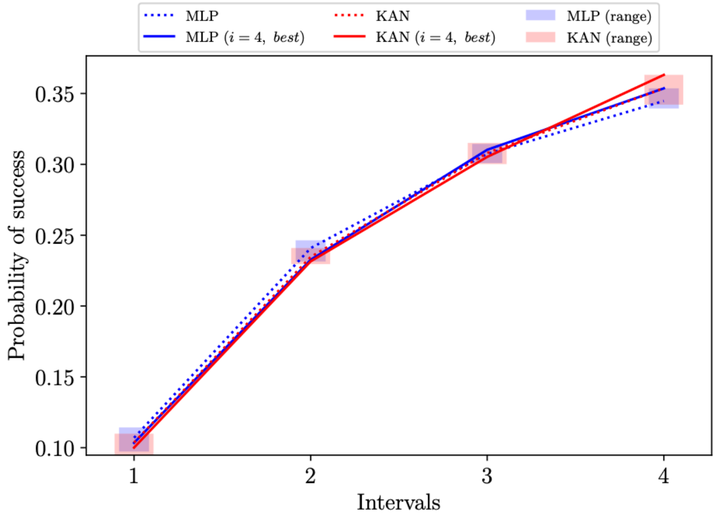
any gates). Depending on a fidelity-based reward function of the form

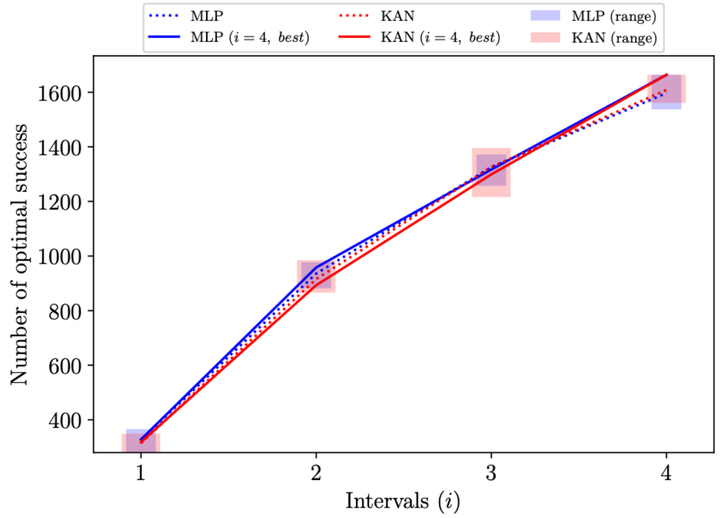
$$\mathbb{A} = \{CX, X, Y, Z, H, T\}.$$

$$R = \begin{cases} \mathcal{R}, & \text{if } F(s_t) \geq 0.98 \\ F(s_t), & \text{otherwise} \end{cases} \tag{8}$$

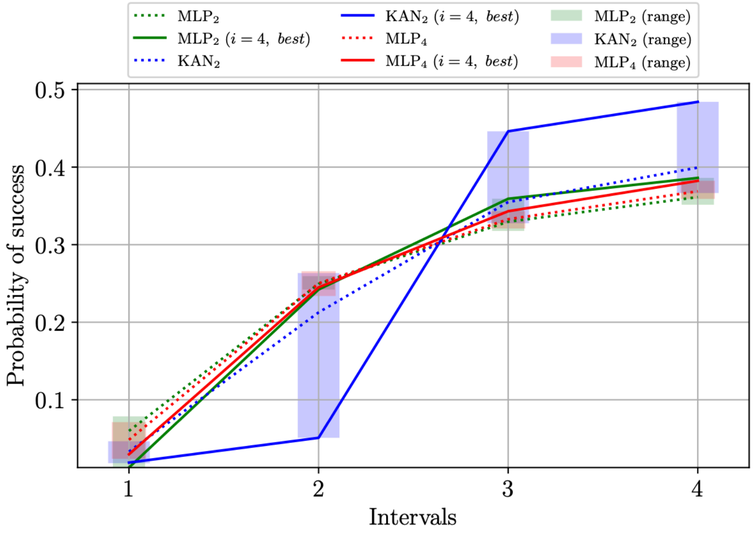
and by following an ϵ-greedy policy the RL agent sets the probability of selecting a random action. Where $F(s_t)$ is the fidelity of a state at step t generated by KANQAS and $\mathcal{R} >> F(s_t)$, is a hyperparameter.

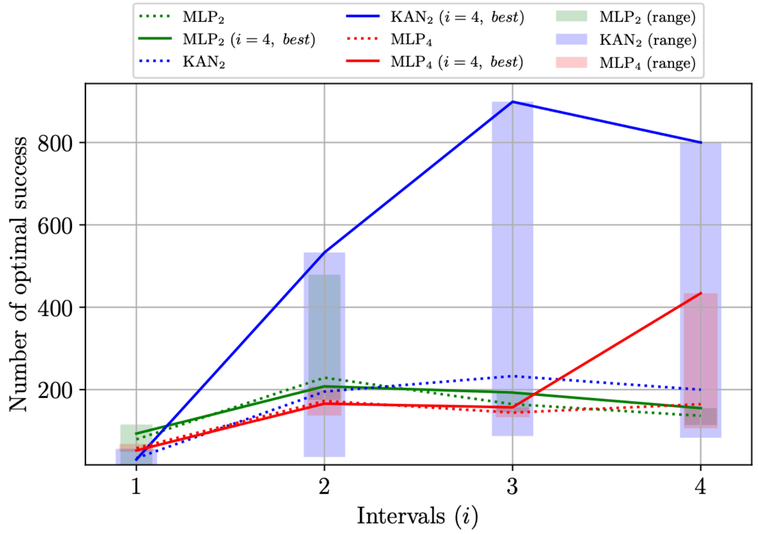
KANQAS





Bell





GHZ

KANQAS

Network	Configuration	Spline and grid	Activation func.	Fidelity
KAN	$[84, 10, 10, 12]$	B-spline, $k = 4, G = 5$	learnable	**0.7328**
$\text{MLP}_{4,30}$	$[84, 30, 30, 30, 12]$	NA	ReLU	0.5005
		NA	LeakyReLU	0.7300
$\text{MLP}_{4,100}$	$[84, 100, 100, 100, 12]$	NA	ELU	0.6830
			ReLU	0.7300
			LeakyReLU	**0.8500**

TABLE II: KAN outperforms most of the configurations of MLPs except when $\text{MLP}_{4,100}$ is operated with LeakyReLU activation function for noisy Bell state preparation with $p_{\text{x}} = 0.3$ and $p_{\text{dep}} = 0.2$.

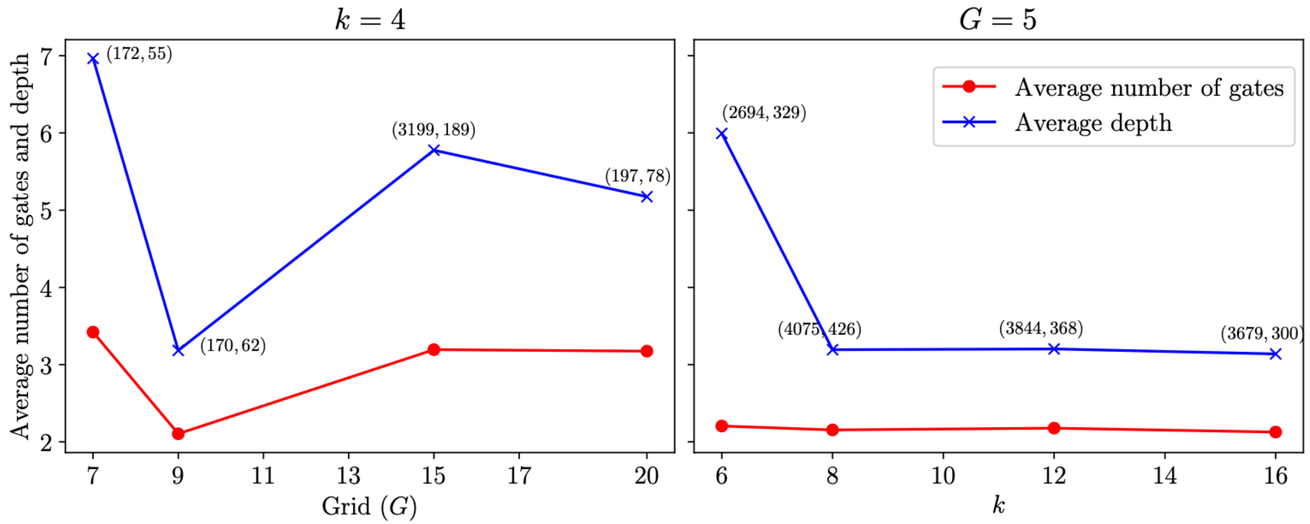


FIG. 4: The number of splines (k) has more impact in improving and stabilizing the performance of the KAN than the grid size (G). Here we measure the improvement of the KAN while constructing GHZ state by calculating the average number of gates and depth in each setting. Each point is marked as (a, b) where a is the total number of successful episodes and b total optimal successful episodes.

KANQAS

Problem	Network	Configuration	Spline and grid	Parameter count
Bell state prep. (noiseless)	KAN	$[84, 2, 12]$	B-spline, $k = 3, G = 5$	64
	MLP	$[84, 30, 30, 30, 12]$	NA	480
GHZ state prep. (noiseless)	KAN	$[288, 3, 21]$	B-spline, $k = 4, G = 5$	108
	MLP	$[288, 30, 30, 30, 21]$	NA	480
Bell state prep. (noisy)	KAN	$[84, 10, 10, 12]$	B-spline, $k = 4, G = 5$	810
	MLP	$[84, 100, 100, 100, 12]$	NA	1600

TABLE III: The required number of learnable parameters for KANs and MLPs.

Problem	Network	Configuration	Avg. time per episode	Avg time per successful episode
GHZ state prep.	KAN	$[288, 3, 21]$	7.094	4.097
	MLP	$[288, 30, 30, 30, 21]$	0.0592	0.0383

GKAN

Kolmogorov-Arnold Graph Neural Networks

G De Carlo, A Mastropietro… - arXiv preprint arXiv …, 2024 - arxiv.org

15 days ago - Graph neural networks (GNNs) excel in learning from network-like data b often lack interpretability, making their application challenging in domains requiring …

☆ Save 99 Cite Related articles All 2 versions »

Message Passing and Aggregation: Each node i in the graph has an initial feature vector $\mathbf{x}_i$. For each layer l in the GKAN, the node representations are updated through message passing and aggregation. The message from node j to node i at layer l is denoted as $\mathbf{m}_{j\to i}^{(l)}$.

1) **Message Passing**:

$$\mathbf{m}_{j\to i}^{(l)} = \text{spline}(\mathbf{x}_j^{(l-1)})$$

where $\text{spline}(\cdot)$ denotes the spline-based activation function applied to the features of node j.

2) **Aggregation**: The aggregated message for node i is obtained by summing or averaging the messages from its neighbors $\mathcal{N}(i)$:

$$\mathbf{a}_i^{(l)} = \sum_{j\in\mathcal{N}(i)} \mathbf{m}_{j\to i}^{(l)}$$

or

$$\mathbf{a}_i^{(l)} = \frac{1}{|\mathcal{N}(i)|} \sum_{j\in\mathcal{N}(i)} \mathbf{m}_{j\to i}^{(l)}$$

Node Update: The node representation is then updated using the aggregated message and a spline-based transformation:

$$\mathbf{x}_i^{(l)} = \text{spline}(\mathbf{a}_i^{(l)})$$

Spline-Based Activation Function: The spline-based activation function φ used in GKAN is defined as:

$$\varphi(x) = w_b b(x) + w_s \text{spline}(x)$$

where $b(x)$ is a basis function (e.g., SiLU), and $\text{spline}(x)$ is a B-spline function:

$$\text{spline}(x) = \sum_i c_i B_i(x)$$

with $B_i(x)$ being the basis splines.

GKAN

GKAN Layer: Each GKAN layer combines these steps, resulting in the following layer-wise update rule for node i:

$$\mathbf{x}_i^{(l)} = \varphi\left(\sum_{j \in \mathcal{N}(i)} \varphi(\mathbf{x}_j^{(l-1)})\right)$$

Output Layer: After passing through multiple GKAN layers, the final node representations are transformed using a KAN-based linear layer for tasks like classification or regression:

$$\mathbf{z}_i = \text{KANLinear}(\mathbf{x}_i^{(L)})$$

where L is the number of layers and KANLinear applies a final spline-based transformation.

TABLE II: Results in terms of accuracy (node and graph classification) and AUC-ROC (link prediction) obtained by GKAN and the compared architectures. Our framework delivers higher accuracy and ROC, being the top-performing architecture.

Dataset	GCN	GAT	GraphSAGE	GKAN (Ours)
Node Classification				
Cora	76.3	78.9	73.4	**81.2**
PubMed	77.4	78.2	73.9	**81.0**
CiteSeer	66.7	68.2	62.3	**69.4**
Link Prediction				
Cora	79.3	81.6	80.6	**83.5**
PubMed	**90.6**	86.3	82.4	82.3
CiteSeer	78.7	78.1	78.3	**80.3**
Graph Classification				
MUTAG	71.3	75.1	71.1	**85.0**
PROTEINS	73.8	72.5	70.7	**77.7**

TABLE III: Average time to complete one training epoch on the Cora dataset for each model. These times were measured while training the models on an Apple Silicon M1 processor.

Method	Avg time (s)
GCN	**0.0016**
GAT	0.020
GraphSAGE	0.065
GKAN (ours)	2.052

GKAN

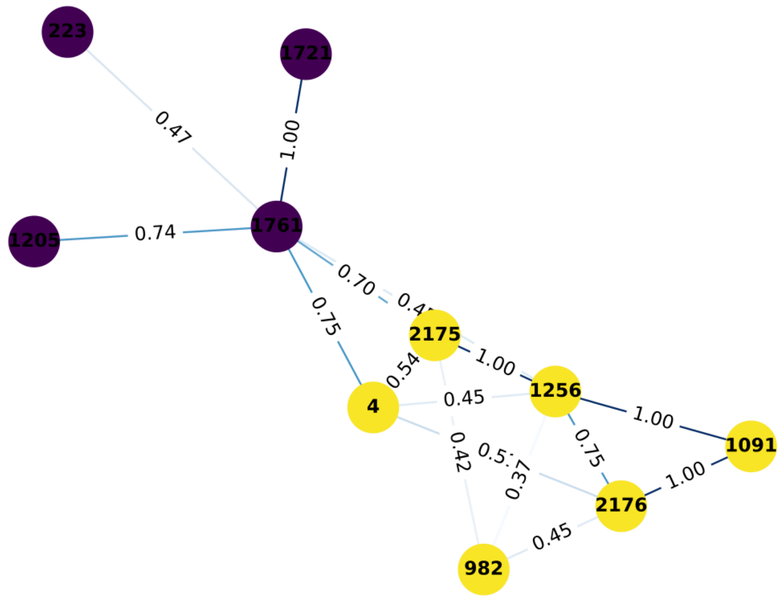


Fig. 2: Interpretability of GKAN at 2 hops from a target node (node with ID 4). The model was trained on the Cora dataset, and in this figure, we are visualizing the weights learned by the splines on the edges to achieve the prediction. It is possible to visualize the direct values that lead to a particular prediction for a number of hops up to the number of layers of the network, as this is the extent to which the messages are aggregated. For visualization purposes, the weights have been normalized.

A. *Limitations and Interpretability*

While GKAN exhibits promising results, it faces certain limitations that requires further investigation:

- **Memory Usage:** Scaling the number of splines or increasing their degree leads to a significant increase in memory consumption. This can pose a challenge when working with large graphs or when higher-degree splines are required to capture complex relationships.
- **Lack of Edge Features:** The current implementation of GKAN does not utilize edge features. Many real-world graph datasets contain valuable information about edges, and incorporating this information could potentially enhance model performance.
- **Interpretability:** : We want to make clear that we do not claim that GKAN is interpretable, but that it is more interpretable than other models. For instance, if the neural network is deep (large number of layers) then some of the information is lost. The added value is that the intepretation is given by the model itself and not by a surrogate explainability model.

KAGNNs

KAGNNs: Kolmogorov-Arnold Networks meet Graph Learning [PDF] arxiv.org

R Bresson, G Nikolentzos, G Panagopoulos… - arXiv preprint arXiv …, 2024 - arxiv.org

15 days ago - In recent years, Graph Neural Networks (GNNs) have become the de facto tool for learning node and graph representations. Most GNNs typically consist of a sequence of …

☆ Save 99 Cite Cited by 1 Related articles All 2 versions »

GIN $$\mathbf{h}_v^{(\ell)} = \texttt{MLP}^{(\ell)}\left((1+\epsilon^{(\ell)})\cdot \mathbf{h}_v^{(\ell-1)} + \sum_{u\in\mathcal{N}(v)} \mathbf{h}_u^{(\ell-1)}\right)$$

GCN $$\mathbf{h}_v^{(\ell)} = \sigma\left(\mathbf{W}^{(\ell)} \sum_{u\in\mathcal{N}(v)\cup\{v\}} \frac{\mathbf{h}_u^{(\ell-1)}}{\sqrt{(\deg(v)+1)(\deg(u)+1)}}\right)$$

KAGIN $$\mathbf{h}_v^{(\ell)} = \texttt{KAN}^{(l)}\left((1+\epsilon)\cdot \mathbf{h}_v^{(\ell-1)} + \sum_{u\in\mathcal{N}(v)} \mathbf{h}_u^{(\ell-1)}\right)$$

KAGCN $$\mathbf{h}_v^{(\ell)} = \Phi^{(\ell)}\left(\sum_{u\in\mathcal{N}(v)\cup\{v\}} \frac{\mathbf{h}_u^{(\ell-1)}}{\sqrt{(\deg(v)+1)(\deg(u)+1)}}\right)$$

KAGNNs

Table 1: Average classification accuracy (± standard deviation) of the `KAGIN`, GIN, `KAGCN`, and GCN models on the 7 node classification datasets.

Dataset	KAGIN	GIN	KAGCN	GCN
Cora	0.7620 ± 0.0077	0.5528 ± 0.1112	**0.7826** ± 0.0177	0.7566 ± 0.0718
Citeseer	0.6837 ± 0.0117	0.4748 ± 0.0574	0.6409 ± 0.0185	**0.6891** ± 0.0074
Ogbn-arxiv	0.2450 ± 0.0806	0.2110 ± 0.0628	**0.2997** ± 0.0274	0.2777± 0.0699
Cornell	**0.5243** ± 0.0813	0.4000 ± 0.1414	0.4892 ± 0.0547	0.4162 ± 0.0737
Texas	0.5973 ± 0.0505	**0.6054** ± 0.0516	0.5811 ± 0.0440	0.5892 ± 0.0415
Wisconsin	**0.5510** ± 0.0720	0.4961 ± 0.1532	0.5196 ± 0.1020	0.5216 ± 0.0729
Actor	**0.2890** ± 0.0139	0.2817 ± 0.0110	0.2755 ± 0.0134	0.2852 ± 0.0129

Table 3: Average classification accuracy (± standard deviation) of the `KAGIN` and GIN models on the 6 graph classification datasets.

	MUTAG	**DD**	**NCI1**	**PROTEINS**
GIN	85.09 ± 5.97	73.68 ± 3.89	79.57 ± 1.27	70.26 ± 3.62
KAGIN	**85.45** ± 7.38	**75.46** ± 4.16	**79.72** ± 1.79	**71.78** ± 3.12
	ENZYMES	**IMDB-B**	**IMDB-M**	
GIN	**53.72** ± 9.23	73.03 ± 2.86	**49.78** ± 3.38	
KAGIN	42.94 ± 5.69	**73.83** ± 4.35	**49.78** ± 4.69	

	ZINC-12K	**QM9**
GIN	0.4131 ± 0.0215	0.0969 ± 0.0017
KAGIN	**0.3000** ± 0.0332	**0.0618** ± 0.0007

FBKANs

Finite basis Kolmogorov-Arnold networks: domain decomposition for data-driven and physics-informed problems

[PDF] arxiv.org

AA Howard, B Jacob, SH Murphy, A Heinlein… - arXiv preprint arXiv …, 2024 - arxiv.org

14 days ago - Kolmogorov-Arnold networks (KANs) have attracted attention recently as an alternative to multilayer perceptrons (MLPs) for scientific machine learning. However, KANs …

☆ Save Cite Related articles All 3 versions »

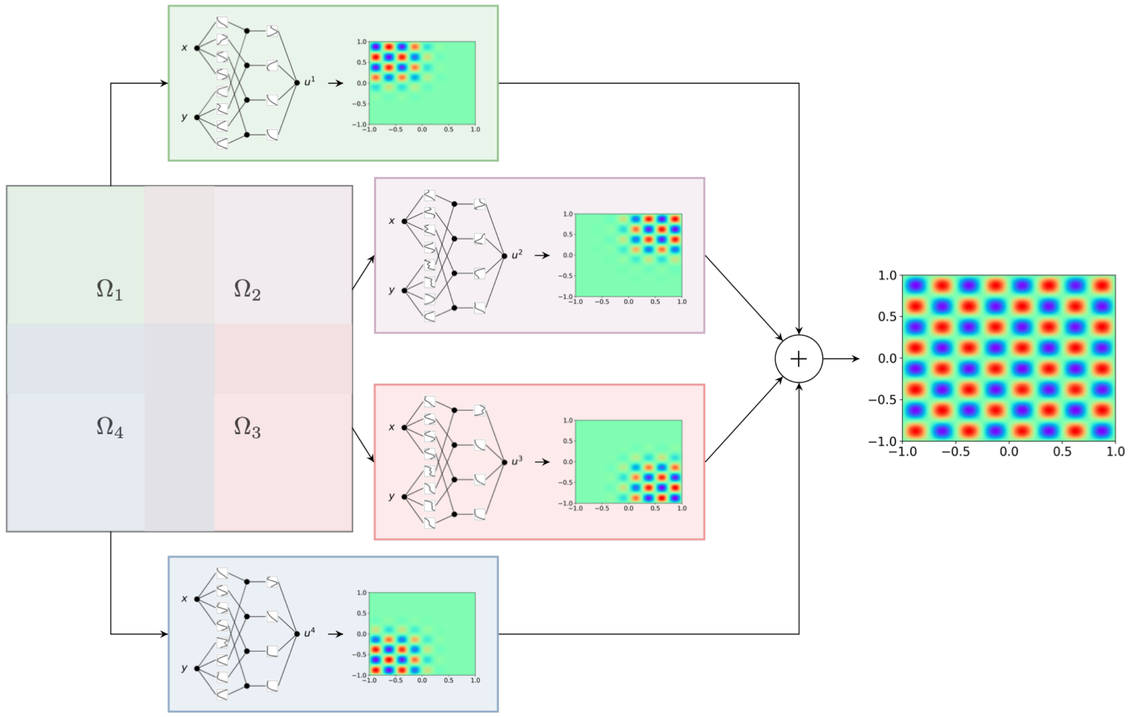


Figure 1: Graphical abstract.

FBKANs

2.2 Physics-informed neural networks

A PINN is a neural network (NN) that is trained to approximate the solution of an initial-boundary value problem (IBVP) by optimizing loss terms accounting for initial conditions, boundary conditions, and a residual term using backpropagation to calculate derivatives. In particular, the residual term represents how well the output satisfies the governing differential equation [19]. More specifically, we aim at approximating the solution f of a generic IBVP

$$\begin{aligned} f_t + \mathcal{N}_x[f] &= 0, && x \in \Omega, t \in [0,T], \\ f(x,t) &= g(x,t), && x \in \partial\Omega, t \in [0,T], \\ f(x,0) &= u(x), && x \in \Omega, \end{aligned} \tag{3}$$

$$\mathcal{L}(\theta) = \lambda_{ic}\mathcal{L}_{ic}(\theta) + \lambda_{bc}\mathcal{L}_{bc}(\theta) + \lambda_r\mathcal{L}_r(\theta) + \lambda_{data}\mathcal{L}_{data}(\theta),$$

$$\mathcal{L}(\theta) = \lambda_{ic}\mathcal{L}_{ic}(\theta) + \lambda_{bc}\mathcal{L}_{bc}(\theta) + \lambda_r\mathcal{L}_r(\theta),$$

$$\mathcal{L}_{ic}(\theta) = \frac{1}{N_{ic}} \sum_{i=1}^{N_{ic}} \left(f_\theta(x_{ic}^i, 0) - u(x_{ic}^i) \right)^2,$$

$$\mathcal{L}_{bc}(\theta) = \frac{1}{N_{bc}} \sum_{i=1}^{N_{bc}} \left(f_\theta(x_{bc}^i, t_{bc}^i) - g(x_{bc}^i, t_{bc}^i) \right)^2,$$

$$\mathcal{L}_r(\theta) = \frac{1}{N_r} \sum_{i=1}^{N_r} \left(\frac{\partial}{\partial t} f_\theta(x_r^i, t_r^i) + \mathcal{N}_x[f_\theta(x_r^i, t_r^i)] \right)^2,$$

FBKANs

$$f_\theta(x) = \sum_{j=1}^{L} \omega_j(x) f_{j,\theta^j}(x),$$

2.3 FBPINNs

In FBPINNs [37, 42, 43, 44], we decompose the space domain Ω or the space-time domain $\Omega \times [0, T]$ into overlapping subdomains. Each overlapping subdomain Ω_j is the interior of the support of a corresponding function ω_j, and all functions ω_j form a partition of unity. In particular, in the case of L overlapping subdomains, we have

$$\Omega = \bigcup_{j=1}^{L} \Omega_j, \quad \text{supp}(\omega_j) = \overline{\Omega_j}, \text{ and } \quad \sum_{j=1}^{L} \omega_j \equiv 1 \text{ in } \Omega.$$

In one dimension, a uniform overlapping domain decomposition with overlap ratio $\delta > 1$ is given by subdomains

$$\Omega_j = \left(\frac{(j-1)l - \delta l/2}{L-1}, \frac{(j-1)l + \delta l/2}{L-1} \right),$$

where $l = \max(x) - \min(x)$.

There are multiple ways to define the partition of unity functions. Here, we construct them based on the expression

$$\omega_j = \frac{\hat{\omega}_j}{\sum_{j=1}^{L} \hat{\omega}_j}, \tag{8}$$

where

$$\hat{\omega}_j(x) = \begin{cases} 1 & L = 1, \\ \left[1 + \cos\left(\pi(x - \mu_j)/\sigma_j\right)\right]^2 & L > 1, \end{cases} \tag{9}$$

with $\mu_j = l(j-1)/(L-1)$ and $\sigma_j = (\delta l/2)/(L-1)$ representing the center and half-width of each subdomain, respectively. An example of the one-dimensional partition of unity functions with $L = 4$ is depicted in fig. 2. In multiple dimensions, $\mathbf{x} \in \mathbb{R}^d$, we then obtain

$$\hat{\omega}_j(x) = \begin{cases} 1 & L = 1, \\ \Pi_{i=1}^{d} \left[1 + \cos\left(\pi(x_i - \mu_{ij})/\sigma_{ij}\right)\right]^2 & L > 1, \end{cases} \tag{10}$$

with μ_{ij} and σ_{ij} representing the center and half-width of each subdomain j in each direction i. An example of of a single partition of unity function with $d = 2$ and $L = 4$ is shown in fig. 3.

FBKANs

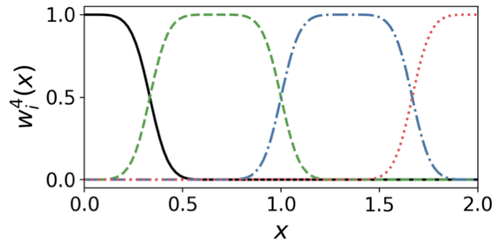


Figure 2: Example the partition of unity functions on the domain $\Omega = [0, 2]$ with $L = 4$ subdomains.

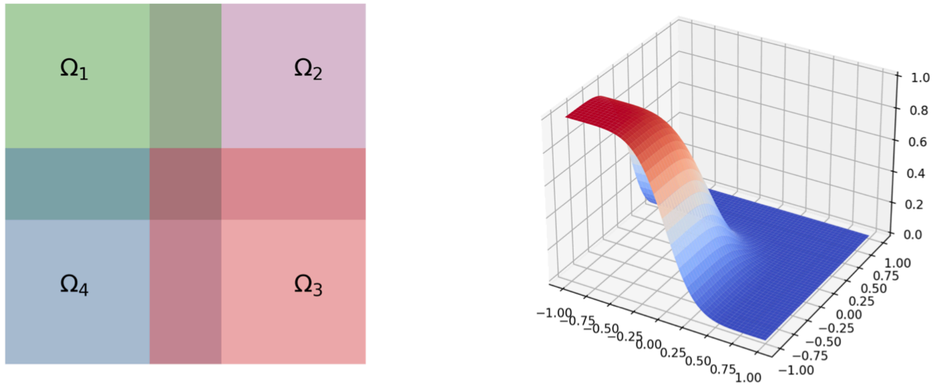


Figure 3: (Left) Example domain decomposition on the domain $\Omega = [-1, 1] \times [-1, 1]$ with $L = 4$ subdomains. (Right) One example partition of unity function $\omega_{11}(x, y)$.

FBKANs

$$\mathcal{L}_{ic}(\theta) = \frac{1}{N_{ic}} \sum_{i=1}^{N_{ic}} \left(\sum_{j=1}^{L} \omega_j(x) \mathcal{K}^j(x_{ic}^i; \theta^j) - u(x_{ic}^i) \right)^2,$$

$$\mathcal{L}_{bc}(\theta) = \frac{1}{N_{bc}} \sum_{i=1}^{N_{bc}} \left(\sum_{j=1}^{L} \omega_j(x) \mathcal{K}^j(x_{bc}^i; \theta^j) - g(x_{bc}^i, t_{bc}^i) \right)^2,$$

$$\mathcal{L}_{r}(\theta) = \frac{1}{N_{r}} \sum_{i=1}^{N_{r}} \left(\frac{\partial}{\partial t} \left[\sum_{j=1}^{L} \omega_j(x) \mathcal{K}^j(x_{r}^i; \theta^j) \right] + \mathcal{N}_x \left[\sum_{j=1}^{L} \omega_j(x) \mathcal{K}^j(x_{r}^i; \theta^j) \right] \right)^2,$$

$$\mathcal{L}_{data}(\theta) = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} \left(\sum_{j=1}^{L} \omega_j(x) \mathcal{K}^j(x_{data}^i; \theta^j) - f(x_{data}^i) \right)^2.$$

FBKANs

$$f(x) = \exp[\sin(0.3\pi x^2)]$$

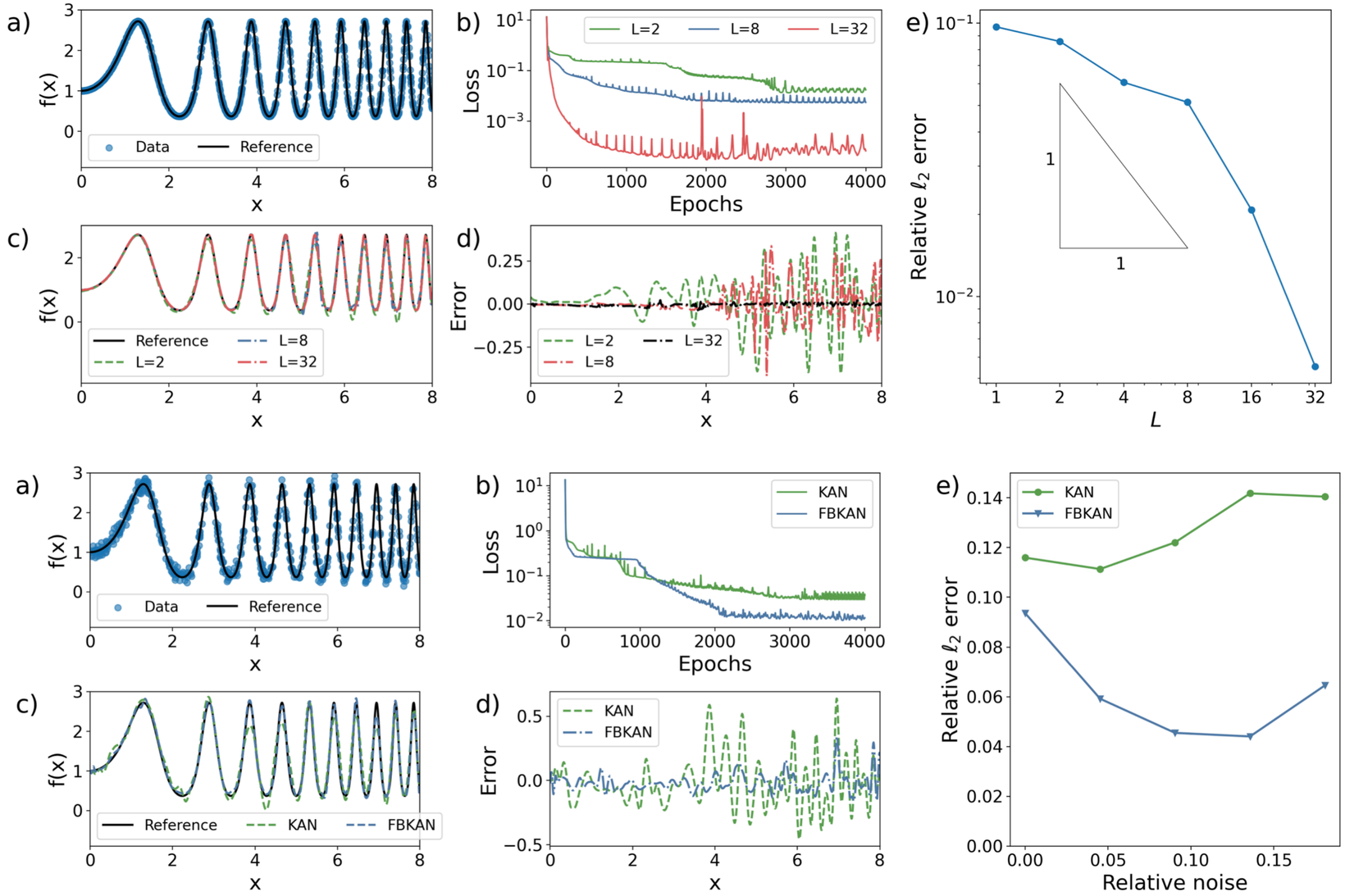


Figure 5: Results for Test 1 with noisy training data; cf. eq. (13). (a) Example training data and plot of exact $f(x)$ with 9.6% mean relative noise. (b) Loss curves (eq. (12)) for an example training with 9.6% mean relative noise. (c) Plot of the outputs $f(x)$ and $\sum_{j=1}^{L} \omega_j(x)\mathcal{K}^j(x)$ with 9.6% mean relative noise. (d) Pointwise errors $f(x) - \sum_{j=1}^{L} \omega_j(x)\mathcal{K}^j(x)$ with 9.6% mean relative noise. (e) Relative ℓ_2 error of the KANs and FBKANs with respect to the magnitude of the noise added to the training data.

FBKANs

$$f(x, y) = \sin(6\pi x^2)\sin(8\pi x^2)$$

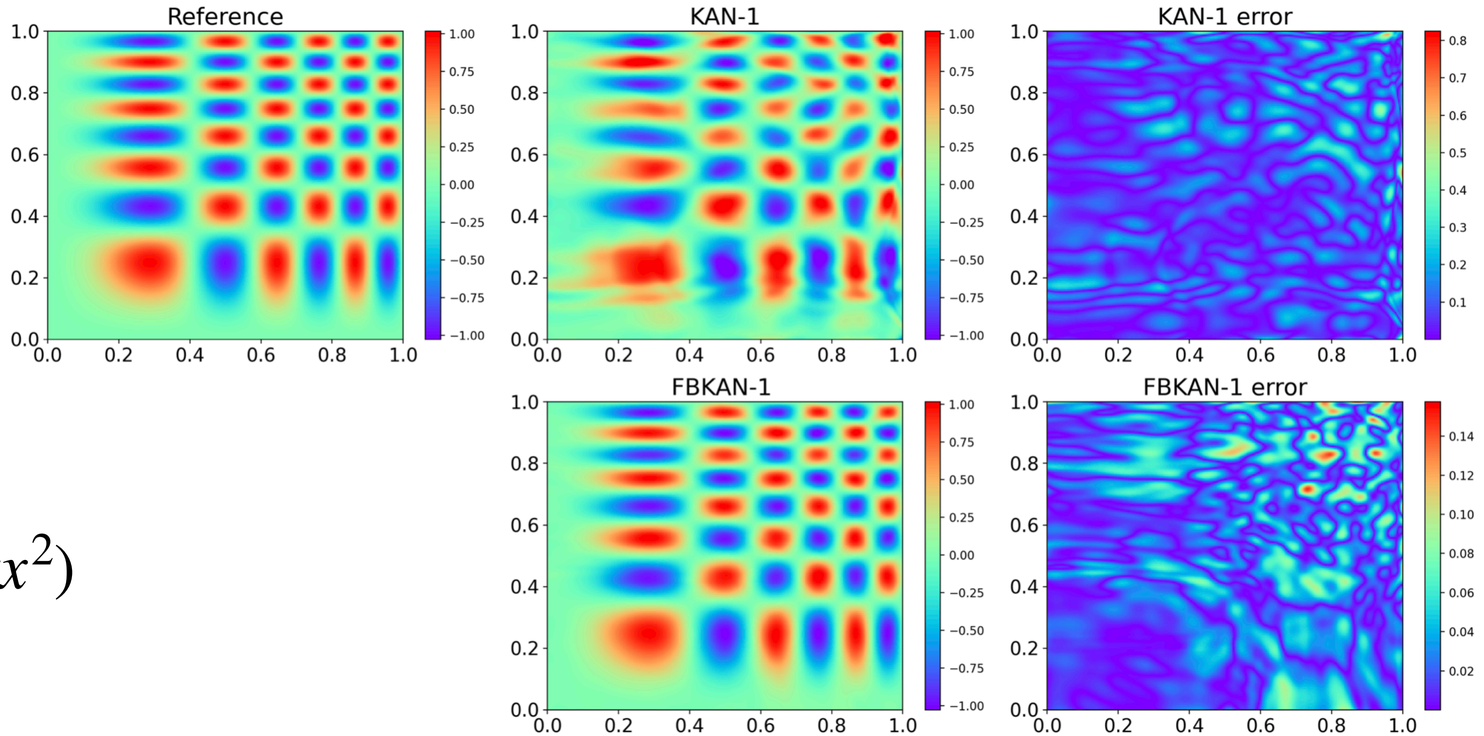
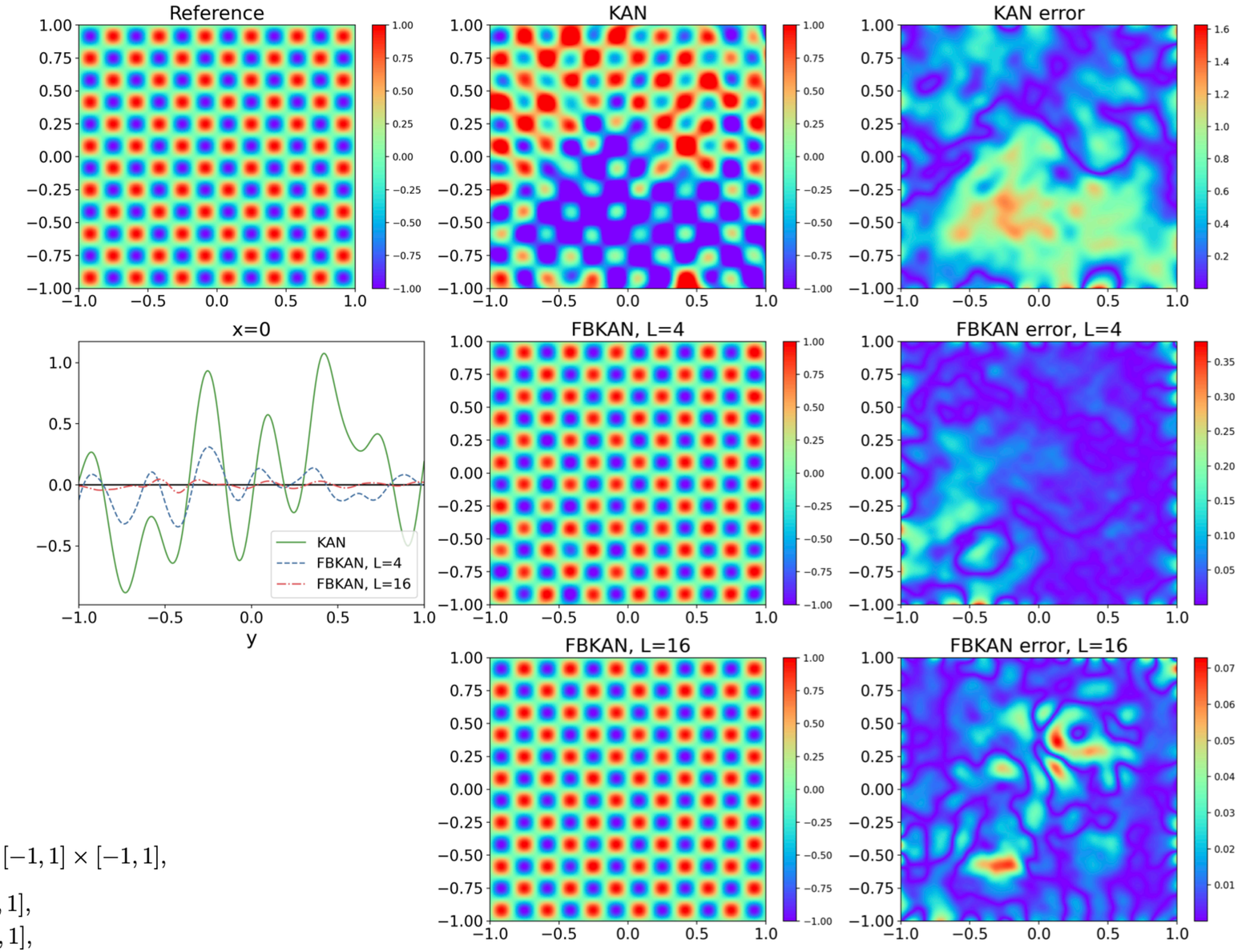


Figure 7: Reference solution (left), predicted solutions (middle) and relative errors (right) for KAN-1 and FBKAN-1 for Test 2. Note that the error plots have different color scales.

FBKANs



4.2 Test 4

We now test the Helmholtz equation

$$\frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial x^2} + k_h^2 f - q(x,y) = 0,\ (x,y) \in [-1,1] \times [-1,1],$$
$$f(-1,y) = f(1,y) = 0,\ y \in [-1,1],$$
$$f(x,-1) = f(x,1) = 0,\ x \in [-1,1],$$

with

$$q(x,y) = -(a_1\pi)^2 \sin(a_1\pi x)\sin(a_2\pi y) - (a_2\pi)^2 \sin(a_1\pi x)\sin(a_2\pi y) + k_h \sin(a_1\pi x)\sin(a_2\pi y).$$

-1 and FBKAN-1 and $a_1 = a_2 = 6$, with $L = 4$ and $L = 16$.The solution along
ft subfigure. Note that the error plots have different color scales.

[PDF] ИССЛЕДОВАНИЕ МЕТОДОВ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА ДЛЯ РАСПОЗНАВАНИЯ МОШЕННИЧЕСКИХ ТРАНЗАКЦИЙ [PDF] sciartel.ru

ВД Палагутина - НАУЧНЫЙ ЖУРНАЛ «IN SITU» - sciartel.ru

13 days ago - ИССЛЕДОВАНИЕ МЕТОДОВ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА ДЛЯ РАСПОЗНАВАНИЯ МОШЕННИЧЕСКИХ ТРАНЗАКЦИЙ Page 18 АКАДЕМИЧЕСКОЕ …

☆ Save 99 Cite Related articles »

???

RESEARCH ON ARTIFICIAL INTELLIGENCE METHODS FOR FRAUDULENT TRANSACTION DETECTION

Abstract

The article is devoted to the development of artificial intelligence methods for recognizing fraudulent transactions. The results of machine learning and neural networks are compared, and the issue of application is considered.

Keywords:

artificial intelligence, machine learning, neural networks, fraud, fraudulent transactions, fraud detection, deep learning, fraud prediction, cybersecurity

Application of Computer Calculation in the Study of Grain Boundary [PDF] mdpi.com

L Pu, C Peng, M Zhu, Y Li, L Li - Coatings, 2024 - mdpi.com

12 days ago - A grain boundary (GB) is a structure of great concern in materials research, which affects the mechanical properties and electrical conductivity of materials, but the …

☆ Save 99 Cite Related articles »

SpectralKAN

SpectralKAN: Kolmogorov-Arnold Network for Hyperspectral Images Change Detection

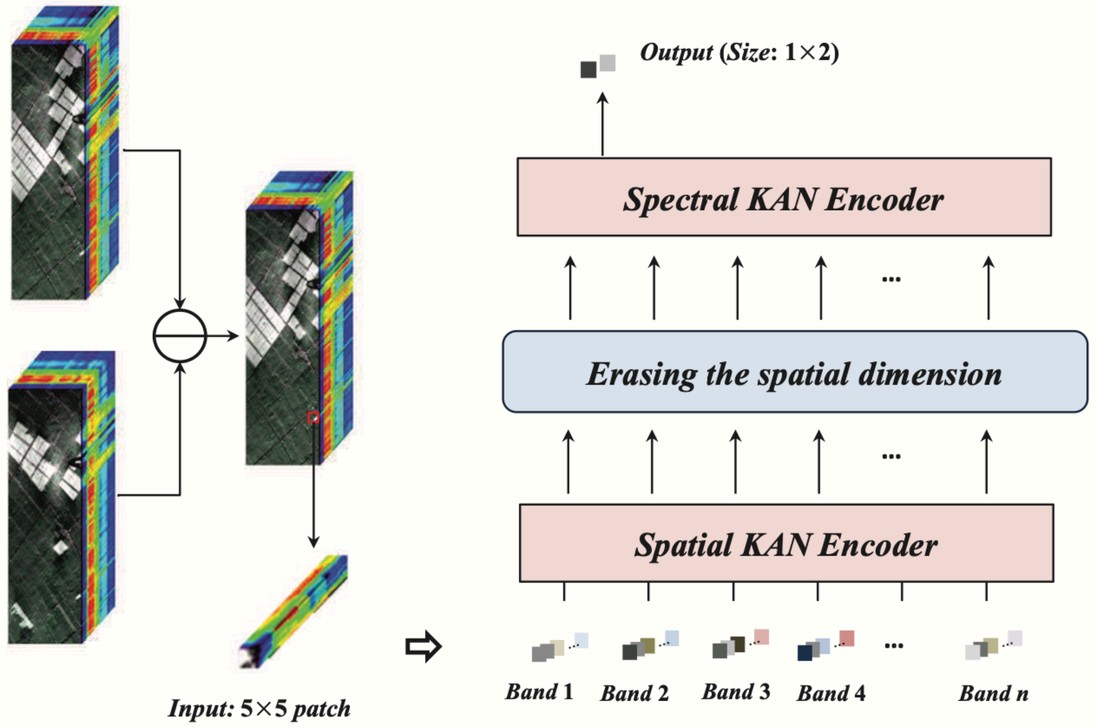
[PDF] arxiv.org

Y Wang, X Yu, Y Gao, J Sha, J Wang, L Gao… - arXiv preprint arXiv …, 2024 - arxiv.org

10 days ago - It has been verified that deep learning methods, including convolutional neural networks (CNNs), graph neural networks (GNNs), and transformers, can accurately extract …

☆ Save Cite Related articles All 2 versions »

***Abstract*—It has been verified that deep learning methods, including convolutional neural networks (CNNs), graph neural networks (GNNs), and transformers, can accurately extract features from hyperspectral images (HSIs). These algorithms perform exceptionally well on HSIs change detection (HSIs-CD). However, the downside of these impressive results is the enormous number of parameters, FLOPs, GPU memory, training and test times required. In this paper, we propose an spectral Kolmogorov-Arnold Network (KAN) for HSIs-CD (SpectralKAN). SpectralKAN represent a multivariate continuous function with a composition of activation functions to extract HSIs feature and classification. These activation functions are b-spline functions with different parameters that can simulate various functions. In SpectralKAN, a KAN encoder is proposed to enhance computational efficiency for HSIs. And a spatial-spectral KAN encoder is introduced, where the spatial KAN encoder extracts spatial features and compresses the spatial dimensions from patch size to one. The spectral KAN encoder then extracts spectral features and classifies them into changed and unchanged categories. We use five HSIs-CD datasets to verify the effectiveness of SpectralKAN. Experimental verification has shown that SpectralKAN maintains high HSIs-CD accuracy while requiring fewer parameters, FLOPs, GPU memory, training and testing times, thereby increasing the efficiency of HSIs-CD. The code will be available at https://github.com/yanhengwang-heu/SpectralKAN.**



SpectralKAN

We assume that there is a pair of co-registered bi-temporal HSIs, denoted as $\mathbf{X_1}$ and $\mathbf{X_2}$. We can get the difference map $\mathbf{X} = \mathbf{X_1} - \mathbf{X_2}$. Then, $\mathbf{X}$ is cropped into numerous patches $\mathbf{x} = [x_1, x_2, \dots]$, $x \in \mathbb{R}^{p \times p \times b}$ to be used as the input of SpectralKAN. $p \times p$ is the patch size, b is the number of bands in $\mathbf{X}$. We reshape x from $\mathbb{R}^{p \times p \times b}$ into $\mathbb{R}^{b \times p^2}$. x is fed into the spatial KAN encoder to get the output $z \in \mathbb{R}^{b \times 1}$

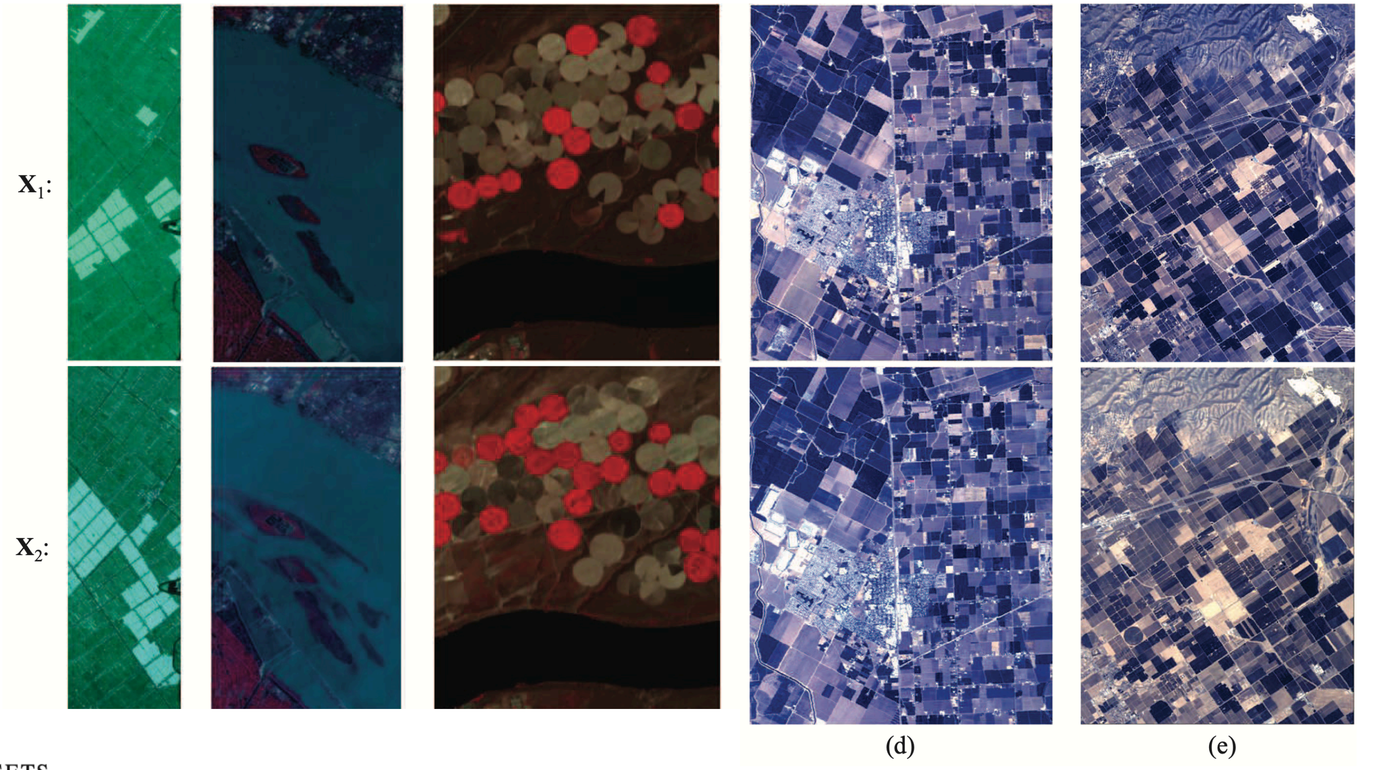
$$z = \mathbf{\Phi}_{1,L_m-1} \odot \cdots \odot \mathbf{\Phi}_{1,1} \odot \mathbf{\Phi}_{1,0} \odot x \tag{6}$$

where $\mathbf{\Phi}_{1,l_m}$ represents the activate functions of the $l_m + 1$ layer spatial KAN encoder. There are L_m layers in total. The final layer of spatial KAN encoder has only one node that compress the spatial dimension of x patch size to one. Then, we reshape z from $\mathbb{R}^{b \times 1}$ to $\mathbb{R}^b$, eliminating the spatial dimension of z. In spite of having only spectral dimensions, z still retains spatial feature information. In the end, we input z into the spectral KAN encoder to extract spectral features and obtain the CD result directly from the last layer of spectral KAN encoder.

$$y' = \mathbf{\Phi}_{2,L_n-1} \odot \cdots \odot \mathbf{\Phi}_{2,1} \odot \mathbf{\Phi}_{2,0} \odot z \tag{7}$$

where $\mathbf{\Phi}_{2,l_n}$ represents the activate functions of the l_n+1 layer spectral KAN encoder. There are L_n layers in total. $y' \in \mathbb{R}^2$ is the output of SpectralKAN. We use the cross-entropy function

SpectralKAN



temporal HSIs, and the second row is the the after-temporal HSIs. (a)

TABLE I
DETAILS OF FIVE HSIS-CD DATASETS

Dataset	Satellite Sensor	Image Times	Cover Land	Size ($h \times w \times b$)	Spatial Resolution	Main Change
Farmland	EO-1 Hyperion	05. 2006 and 04. 2007	Yancheng, China	450 × 140 × 155	30m	land cover and river
river	EO-1 Hyperion	05. 2013 and 12. 2013	Jiangsu, China	431 × 241 × 198	30m	substance in river
USA	EO-1 Hyperion	05. 2004 and 05. 2007	Hermiston, USA	307 × 241 × 154	30m	land cover and river
Bay Area	AVIRIS	2013 and 2015	Bay Area, USA	600 × 500 × 224	20m	land cover
Santa Barbara	AVIRIS	2013 and 2014	Santa Barbara, USA	984 × 740 × 224	20m	land cover

TABLE II
THE NUMBER OF TRAINING AND TEST SETS IN THE FIVE DATASETS, IN UNITS OF PIXELS (p).

Dataset	Unchanged	Changed	Unknown	Training Set		Testing Set	
				Unchanged	Changed	Unchanged	Changed
Farmland	44723	18277	0	447	182	44276	18095
river	101885	9698	0	1018	96	100867	9602
USA	57311	16676	0	573	166	56738	16510
Bay Area	34211	39270	226519	342	392	33869	38878
Santa Barbara	80418	52134	595608	804	521	79614	51613

SpectralKAN

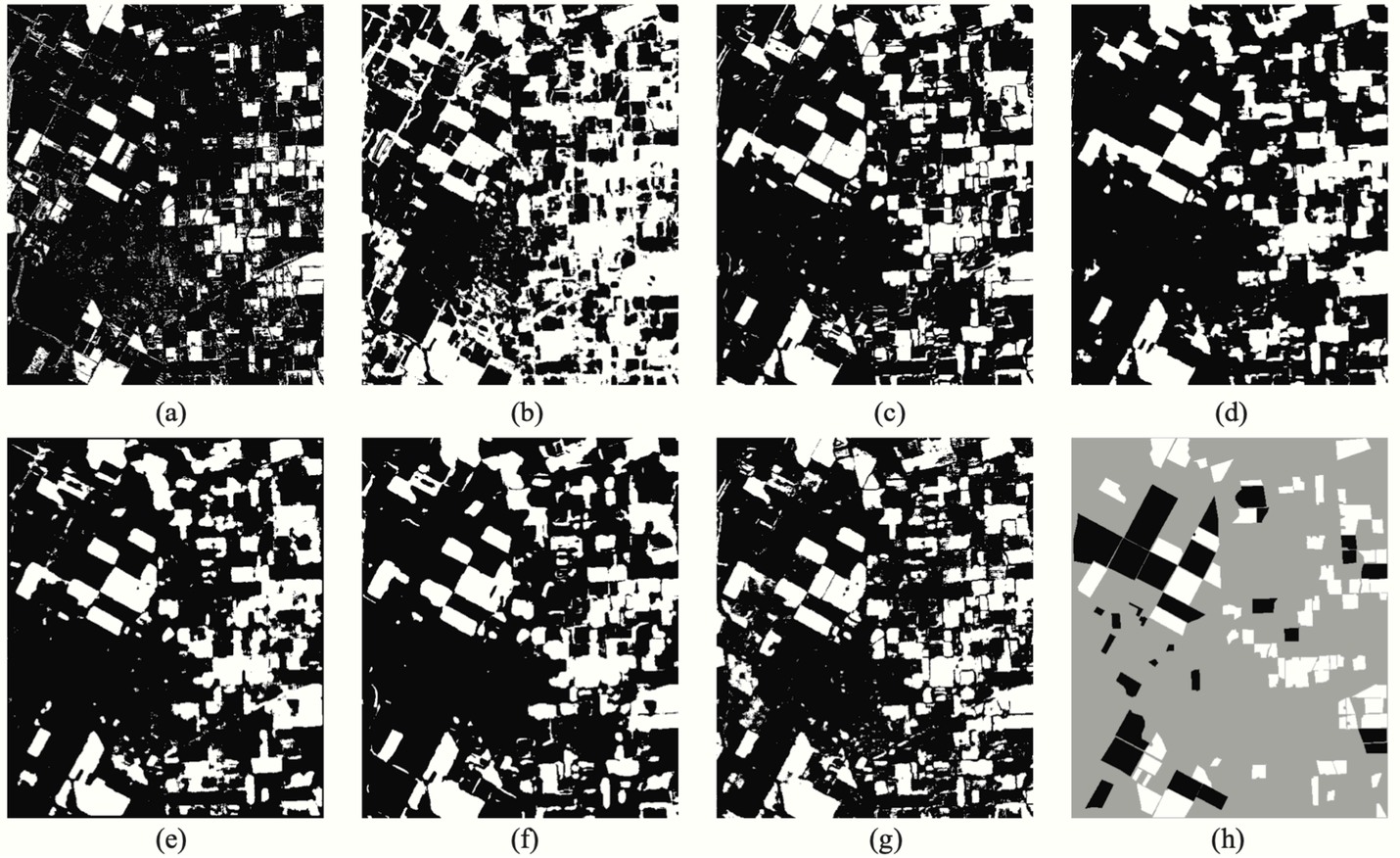


Fig. 7. HSIs-CD results on Bay Area datasets. (a) CVA. (b) SVM. (c) SST-Former. (d) CSANet. (e) TriTF. (f) DA-Former. (g) SpectralKAN. (h) Ground Truth. The white pixels are changed, and the black pixels are unchanged. The gray pixels are unknown on ground truth.

TABLE VII
THE OA AND $\mathcal{K}$ COMPARISON OF STATE-OF-THE-ART HSI-CD METHODS ON SANTA BARBARA.

	OA	$\mathcal{K}$	Parameters(k)	FLOPs(M)	Memory(M)	Training Times(s)	Testing Times(s)
CVA	0.8325	0.6519	-	-	-	-	6.09
SVM	0.9634	0.9344	-	-	-	0.02	4.00
SST-Former	0.9752	0.9478	2534	150	1544	149.85	134.62
CSANet	0.9916	0.9823	2468	147	1368	144.2	75.25
TriTF	0.9854	0.9693	186	22	2464	134.07	250.28
DA-Former	0.992	0.9832	416	33	1984	1590.66	140.03
SpectralKAN	0.9776	0.9531	10	0.1	981	30.10	34.72

Physics-Informed Holomorphic Neural Networks (PIHNNs): Solving Linear Elasticity Problems

[PDF] arxiv.org

M Calafà, E Hovad, AP Engsig-Karup… - arXiv preprint arXiv …, 2024 - arxiv.org

10 days ago - We propose physics-informed holomorphic neural networks (PIHNNs) as a method to solve boundary value problems where the solution can be represented via …

☆ Save Cite Related articles All 3 versions »

HyperKAN: Kolmogorov-Arnold Networks make Hyperspectral Image Classificators Smarter

[PDF] arxiv.org

V Lobanov, N Firsov, E Myasnikov… - arXiv preprint arXiv …, 2024 - arxiv.org

5 days ago - In traditional neural network architectures, a multilayer perceptron (MLP) is typically employed as a classification block following the

☆ Save 99 Cite Related articles All 2 versions »

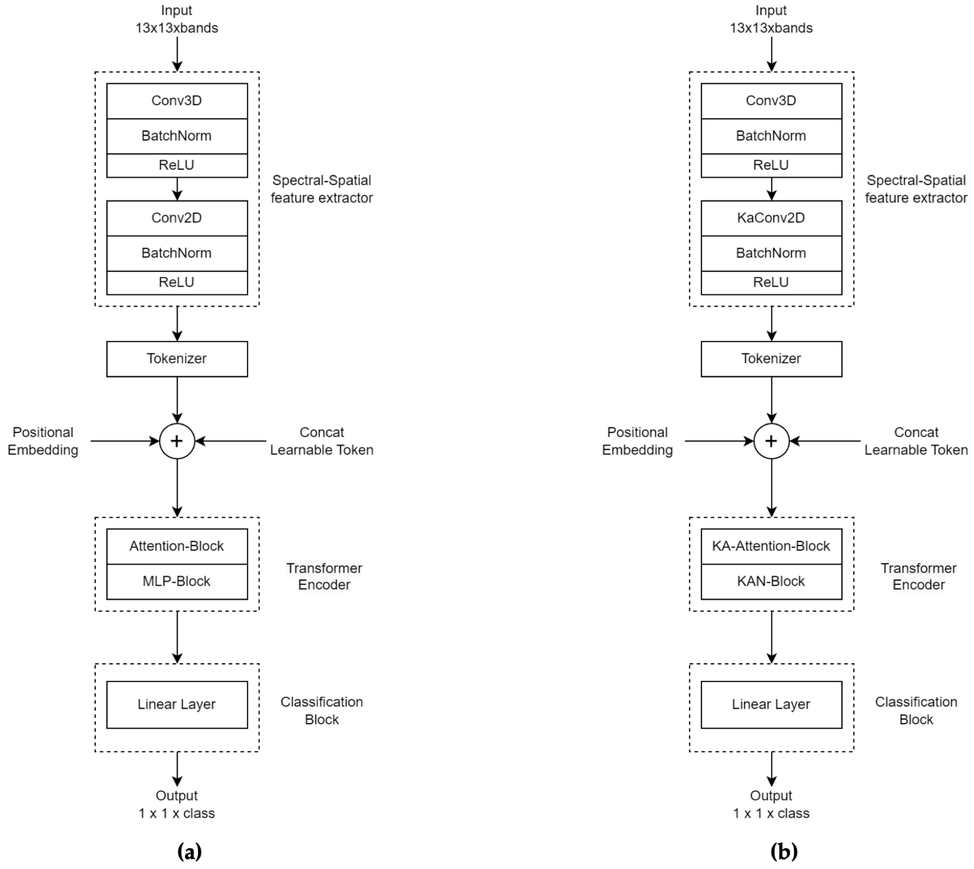


Figure 8. The SSFTT architecture **(a)** and the proposed modification of the SSFTT architecture **(b)**.

Table 2. Overall classification accuracy (OA) for various neural network models and datasets, in percent.

Model Name	Dataset Name							Average gain
	PaviaU*	PaviaC	Salinas	Indian Pines*	Houston13	Houston 18	KSC	
Baseline approaches								
MLP	91.43	97.1	92.32	79.2	87.58	86.27	86.24	2.71
KAN	**94.12**	**98.25**	**93.25**	**84.0**	**94.19**	**87.20**	**88.10**	
Spectral features only								
1DCNN	92.9	97.4	92.0	83.7	92.1	86.7	79.1	4.5
1DCNN KAN	**95.6**	**99.1**	**94.6**	**90.2**	**96.5**	**90.6**	**89.3**	
M1DCNN	94.7	98.5	92.8	84.8	81.3	90.4	85.0	4.6
M1DCNN KAN	**95.3**	**99.1**	**95.2**	**89.2**	**97.4**	**93.2**	**90.4**	
Spectral-spatial features (window size 3x3)								
3DCNN Luo	96.09	98.40	93.30	83.01	94.50	91.79	87.52	2.56
3DCNN Luo KAN	**96.69**	**98.69**	**94.4**	**90.15**	**98.06**	**93.37**	**91.8**	
Spectral-spatial features (window size 7x7)								
3DCNN He	94.4	97.6	93.0	83.4	89.3	87.9	90.1	2
3DCNN He KAN	**95.5**	**98.3**	**94.4**	**88.2**	**90.9**	**90.8**	**91.6**	
NM3DCNN	94.3	98.2	93.0	85.4	91.3	89.9	91.2	3
NM3DCNN KAN	**95.8**	**98.7**	**96.2**	**90.7**	**96.6**	**91.9**	**94.4**	
Spectral-spatial features (window size 13x13), 30 principal components								
SSFTT	99.68	99.90	99.97	98.89	98.59	96.03	86.32	0.86
SSFTT KAN	**99.80**	**99.91**	**99.98**	**99.05**	**98.85**	**96.04**	**91.10**	

* The training sample size for PaviaU and Indian Pines consisted of 20% samples per class, while for other datasets it was 10% per class.

Sepsis: 패혈증

TCKIN: A Novel Integrated Network Model for Predicting Mortality Risk in Sepsis Patients

[PDF] arxiv.org

F Dong - arXiv preprint arXiv:2407.06560, 2024 - arxiv.org

2 days ago - Sepsis poses a major global health threat, accounting for millions of deaths annually and significant economic costs. Accurate predictions of morta

In this study, we introduce the Temporal-Constant KAN Integrated Network (TCKIN) designed to predict the mortality risk of ICU sepsis patients Figure 1 illustrates the main flowchart of the TCKIN model. Our principa contributions are outlined as follows:

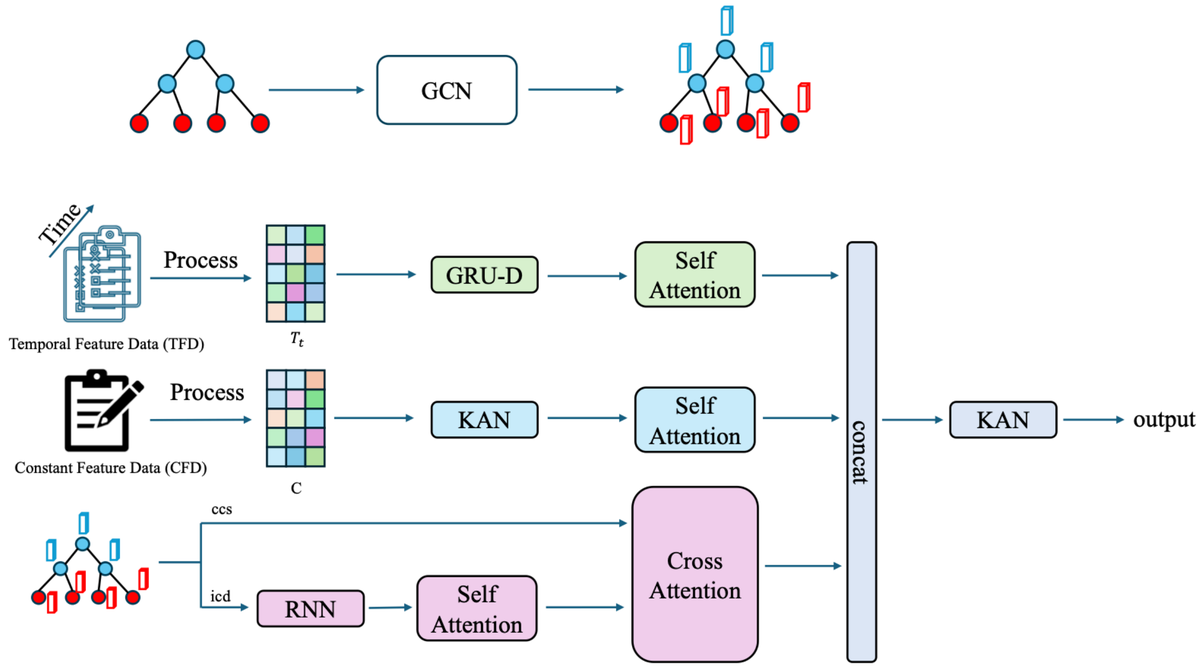


Figure 1: main flowchart of the TCKIN

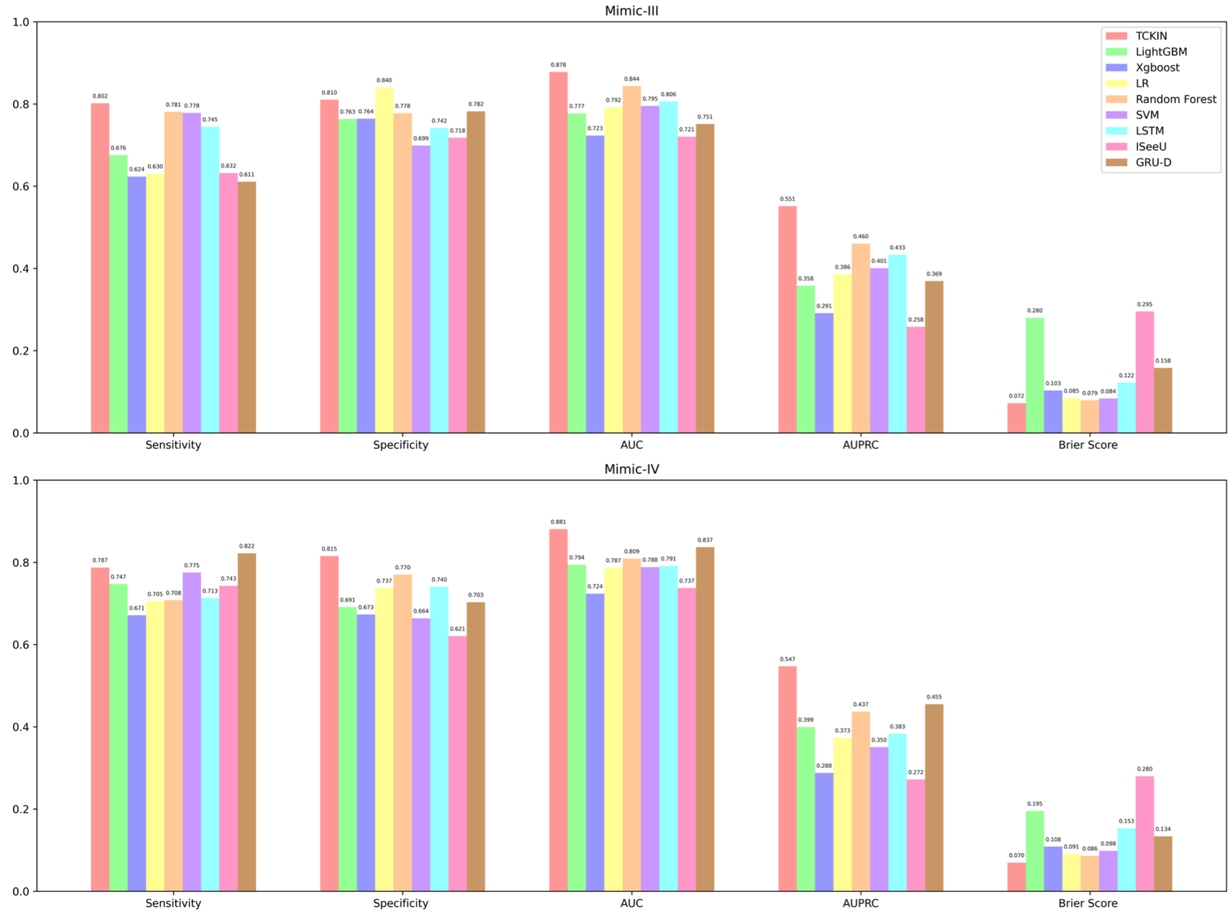


Figure 6: Comparison of performance metrics across various models for predicting mortality risk in ICU sepsis patients using MIMIC-III and MIMIC-IV datasets.