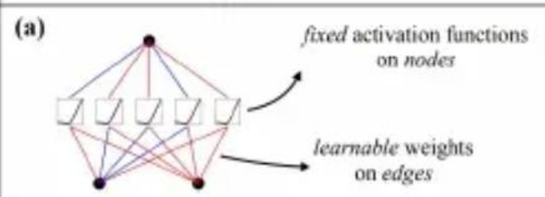
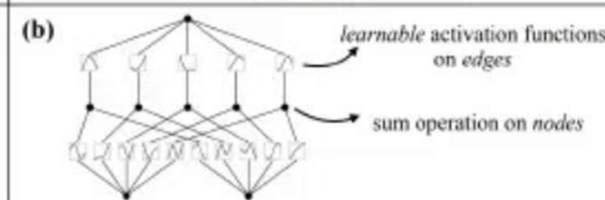
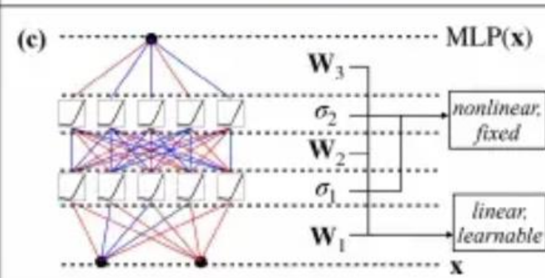
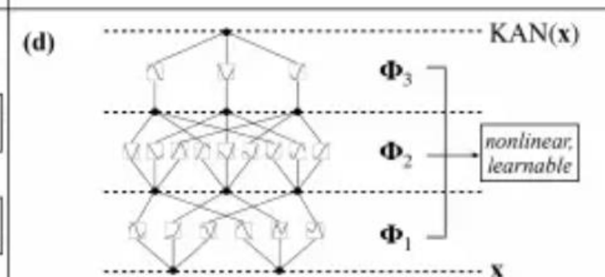
KANs – Kolmogorov Arnold Networks

Germano Vasconcelos

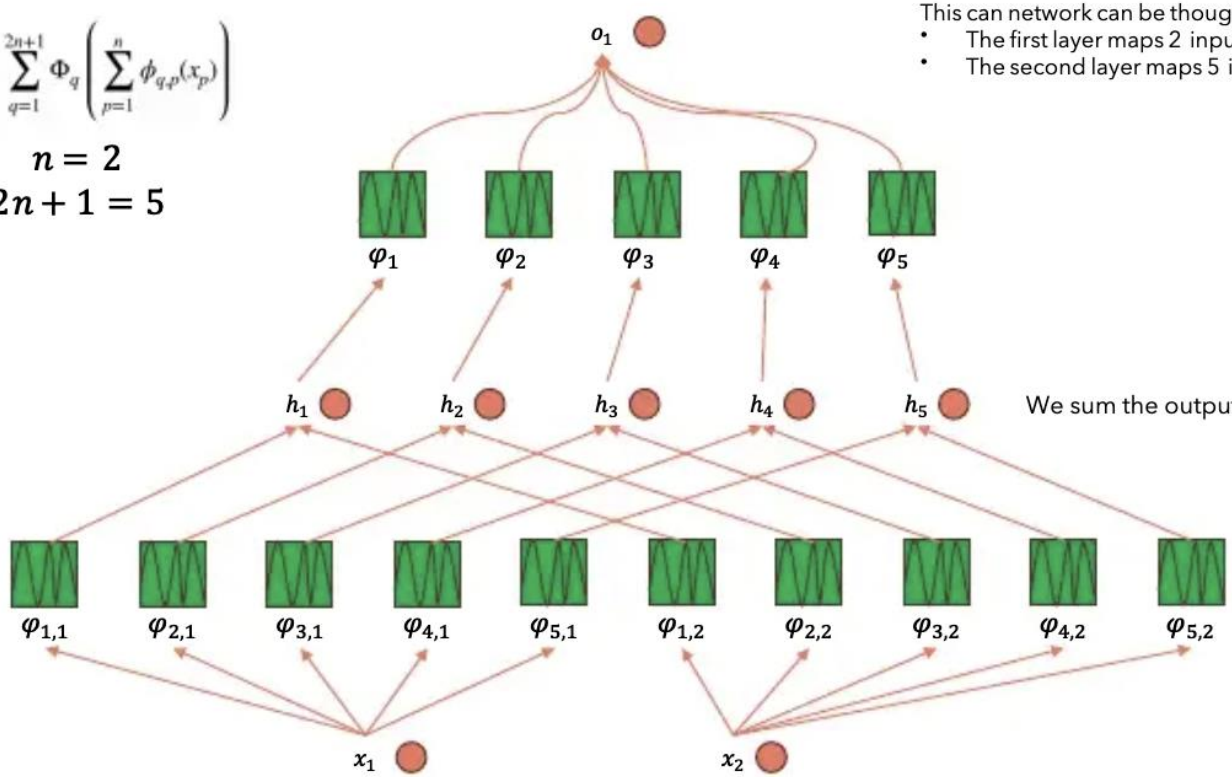
MLP vs KAN

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)		

Funcionamento da KAN

$$f(\mathbf{x}) = \sum_{q=1}^{2n+1} \Phi_q\left(\sum_{p=1}^{n} \phi_{q,p}(x_p)\right)$$

$$n = 2$$
$$2n + 1 = 5$$



This can network can be thought of as two layers applied in sequence:

- The first layer maps 2 input features into 5 output features.
- The second layer maps 5 input features into 1 output feature.

We sum the output of the learnable functions

Instead of having learnable weights, we have **learnable functions**

Multi-layer KAN

$$
\mathbf{x}_{l+1} = \underbrace{\begin{pmatrix} \phi_{l,1,1}(\cdot) & \phi_{l,1,2}(\cdot) & \cdots & \phi_{l,1,n_l}(\cdot) \\ \phi_{l,2,1}(\cdot) & \phi_{l,2,2}(\cdot) & \cdots & \phi_{l,2,n_l}(\cdot) \\ \vdots & \vdots & & \vdots \\ \phi_{l,n_{l+1},1}(\cdot) & \phi_{l,n_{l+1},2}(\cdot) & \cdots & \phi_{l,n_{l+1},n_l}(\cdot) \end{pmatrix}}_{\Phi_l} \mathbf{x}_l,
$$

The breakthrough occurs when we notice the analogy between MLPs and KANs. In MLPs, once we define a layer (which is composed of a linear transformation and nonlinearties), we can stack more layers to make the network deeper. To build deep KANs, we should first answer: "what is a KAN layer?" It turns out that a KAN layer with n_{in}-dimensional inputs and n_{out}-dimensional outputs can be defined as a matrix of 1D functions

$$
\Phi = \{\phi_{q,p}\}, \qquad p = 1, 2, \cdots, n_{\text{in}}, \qquad q = 1, 2 \cdots, n_{\text{out}}, \tag{2.2}
$$

where the functions $\phi_{q,p}$ have trainable parameters, as detaild below. In the Kolmogov-Arnold theorem, the inner functions form a KAN layer with $n_{\text{in}} = n$ and $n_{\text{out}} = 2n + 1$, and the outer functions form a KAN layer with $n_{\text{in}} = 2n + 1$ and $n_{\text{out}} = 1$. So the Kolmogorov-Arnold representations in Eq. (2.1) are simply compositions of two KAN layers. Now it becomes clear what it means to have deeper Kolmogorov-Arnold representations: simply stack more KAN layers!

Layer 2
5 input features, 1 output features
total of 5 functions to "learn"

Layer 1
2 input features, 5 output features
total of 10 functions to "learn"

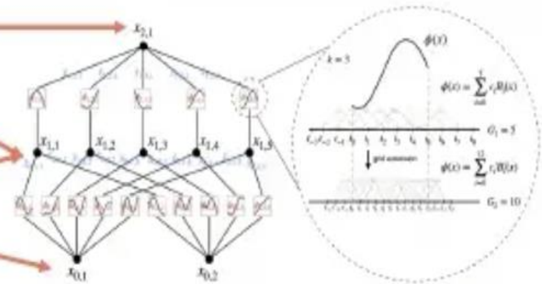
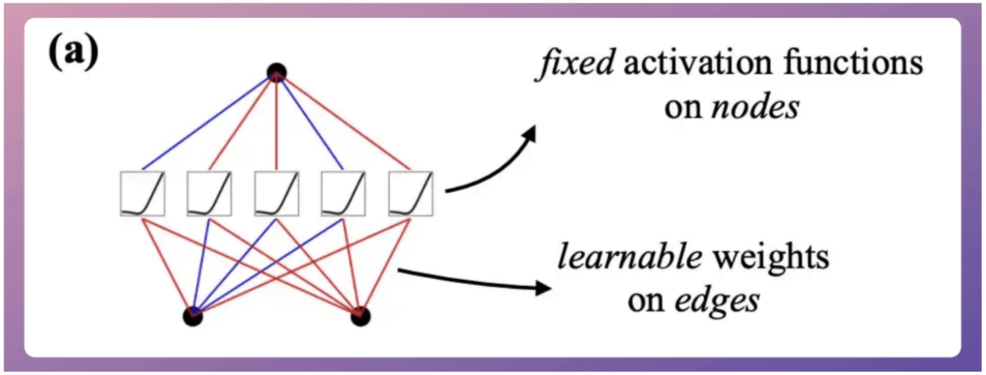
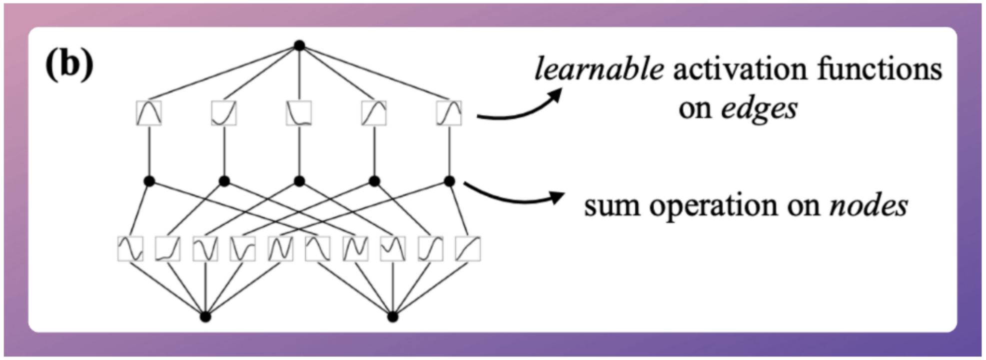


Figure 2.2: Left: Notations of activations that flow through the network. Right: an activation function is parameterized as a B-spline, which allows switching between course-grained and fine-grained grids.

Funcionamento da KAN



KANs, however, move the activation function to the edges and make them trainable parameters:



$2x^2 - 3x + 4$

$$\phi^1 = \begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1n} \\ \phi_{21} & \phi_{12} & \cdots & \phi_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{m1} & \phi_{m2} & \cdots & \phi_{mn} \end{bmatrix}$$

$4x^3 + 5x^2 + x - 2$

To obtain the output of one layer, we can pass the input vector through these functions:

Transformation matrix ϕ^1

Input X

$$z^1 = \begin{bmatrix} \phi_{11}(.) & \phi_{12}(.) & \cdots & \phi_{1n}(.) \\ \phi_{21}(.) & \phi_{12}(.) & \cdots & \phi_{2n}(.) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{m1}(.) & \phi_{m2}(.) & \cdots & \phi_{mn}(.) \end{bmatrix} * \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Output of first layer

Interpretability

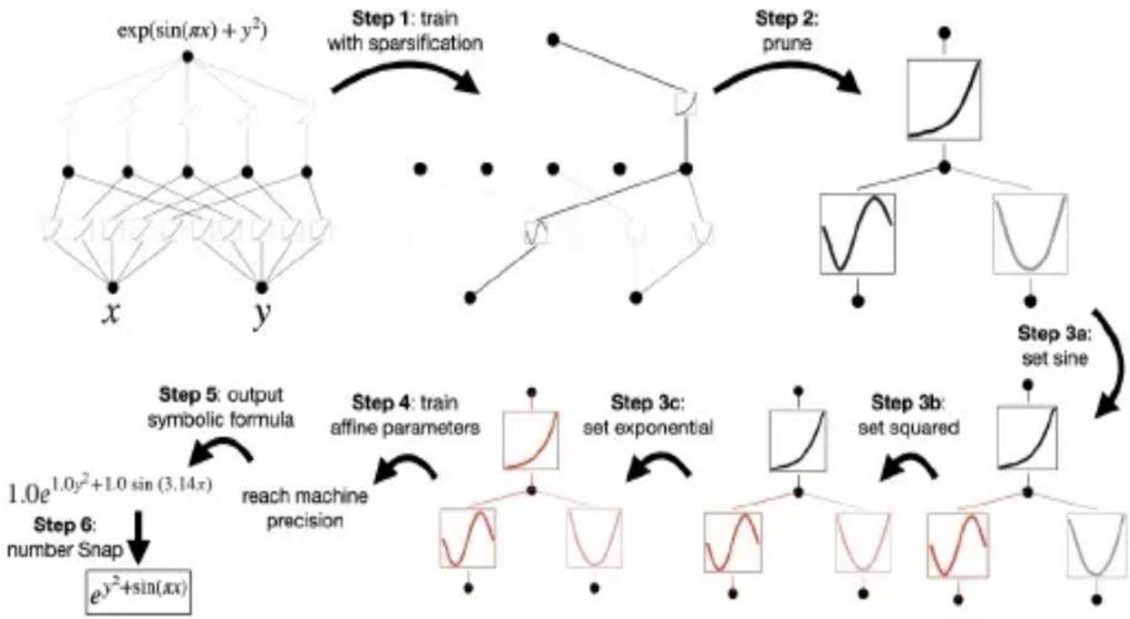


Figure 2.4: An example of how to do symbolic regression with KAN.

Variações da KAN

- **Convolutional KANs (Conv-KANs):** This variant integrates the KAN's learnable, spline-based activation functions into convolutional layers, offering improved parameter efficiency and performance in computer vision tasks compared to traditional Convolutional Neural Networks (CNNs).
- **Temporal KANs (T-KANs and MT-KANs):** Designed for time series forecasting, T-KANs and their multivariate counterpart (MT-KANs) incorporate recurrent and gating mechanisms to capture complex temporal dependencies and long-term memory in sequential data, outperforming traditional Recurrent Neural Networks (RNNs) in some cases.
- **Graph KANs (GKANs and KA-GNNs):** These architectures apply the KAN principles to graph-structured data. They replace fixed operations in traditional Graph Convolutional Networks (GCNs) with learnable univariate functions, demonstrating higher accuracy in node classification and molecular property prediction tasks.
- **Wavelet KANs (Wav-KANs):** This version uses wavelet functions as the basis for the learnable activation functions, which helps capture both low-frequency and high-frequency structural patterns in data. This has shown benefits in tasks like environmental monitoring and hyperspectral image classification.
- **Quantum KANs (QKANs/VQKANs):** An integration of KANs with quantum computing principles, exploring the use of quantum linear algebra tools and parameterized quantum circuits for potential application in quantum machine learning tasks involving high-dimensional inputs.

Variações da KAN

- **Interpretable Variants:**
  - **MonoKAN (Monotonic KAN):** Uses cubic Hermite splines with conditions for monotonicity to ensure the model is more explainable, useful in fields like medical diagnosis where monotonic relationships are expected.
  - **CoxKAN:** Combines KAN with the Cox Proportional Hazards Model for interpretable and high-performance survival analysis in medical datasets.
- **Hybrid Models:** Researchers have also developed hybrid approaches, such as **KAN-XGBoost** for intrusion detection and **U-KAN** for medical image segmentation (integrating KAN into U-Net architectures), to leverage the strengths of KANs within established machine learning frameworks.

These new versions aim to address some of the original KAN's limitations, such as computational efficiency and applicability to different data types, while maintaining the core advantages of interpretability and parameter efficiency on specific tasks.

Variações da KAN

TabKANet: Tabular Data Modeling with Kolmogorov-Arnold Network and Transformer

Weihao Gao[1] , Zheng Gong[1] , Zhuo Deng[1],Fuju Rong[1], Chucheng Chen[1], Lan Ma[1,*]
[1]Shenzhen International Graduate School, Tsinghua University

2024

Kolmogorov–Arnold Transformer

Xingyi Yang Xinchao Wang
National University of Singapore
xyang@u.nus.edu; xinchao@nus.edu.sg

2024

A COMPREHENSIVE SURVEY ON KOLMOGOROV ARNOLD NETWORKS (KAN)

Tianrui Ji
Department of Applied Mathematics
Xi'an Jiaotong-Liverpool University
Tianrui.Ji23@student.xjtlu.edu.cn

Yuntian Hou
Department of AI and Advanced Computing
Xi'an Jiaotong-Liverpool University
Yuntian.Hou20@student.xjtlu.edu.cn

Di Zhang
Department of AI and Advanced Computing
Xi'an Jiaotong-Liverpool University
Di.Zhang@xjtlu.edu.cn

January 29, 2025

2025

Kolmogorov Arnold Networks (KAN) Paper Explained

https://www.youtube.com/watch?v=7zpz_AlFW2w