Kolmogorov-Arnold Networks For Time Series Forecasting

Rémi Genet [1] Hugo Inzirillo [2]

[1]DRM, Université Paris Dauphine [2]CREST, Institut Polytechnique de Paris

Kolmogorov-Arnold Networks (Ziming Liu et al., April 2024)

Kolmogorov-Arnold Networks (KANs) revolutionized neural network architecture in 2024 by introducing a fundamentally different approach to neural computation:

- Instead of fixed activation functions (like ReLU, tanh), KANs use **learnable activation functions** based on b-splines
- Each connection between neurons has its own unique activation function
- This design is inspired by Kolmogorov's universal approximation theorem

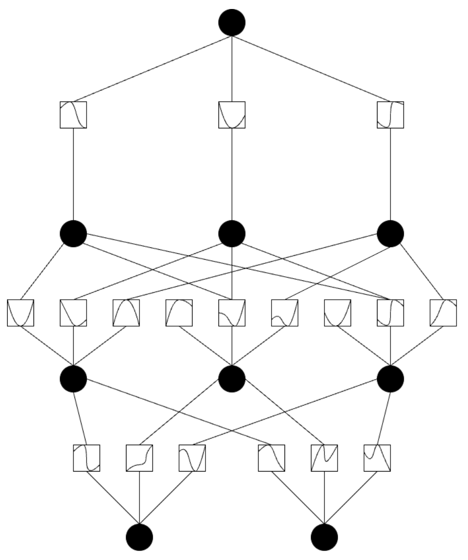


Figure 1. KAN Network architecture

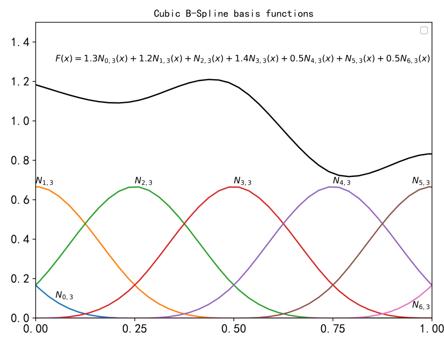


Figure 2. B-Spline Basis Function

KANs demonstrate **remarkable performance** on synthetic datasets, significantly outperforming traditional MLPs in function approximation tasks. However, there are trade-offs to consider:

- Computational overhead: The learnable activation functions increase training complexity
- Variable performance: Results on real-world datasets can be less consistent, depending on data characteristics

TKAN: Temporal Kolmogorov-Arnold Networks

While KANs offer a novel alternative to MLPs, they lack proper mechanisms for time series data. To do so we first introduced the recurrent KAN layer (RKAN), that incorporate a simple memory mecanism. The inputs that is fed to each sub KAN layers in the RKAN are created by doing:

$$s_{l,t} = W_{l,\tilde{x}} x_t + W_{l,\tilde{h}} \tilde{h}_{l,t-1}, \tag{1}$$

where $W_{l,\tilde{x}}$ is the weight of the l-th layer applied to x_t which is the input at time t. The "memory" step $\tilde{h}_{l,t}$ is defined as a combination of past hidden states, such as:

$$\tilde{h}_{l,t} = W_{hh} \tilde{h}_{l,t-1} + W_{hz} \tilde{o}_t, \tag{2}$$

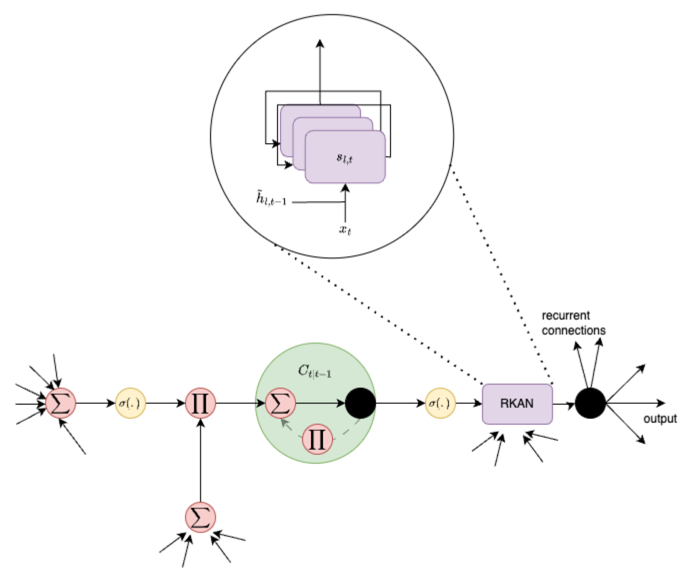


Figure 3. Two-layer TKAN architecture showing stacked recurrent layers with memory management

We then used RKAN inside a broader TKAN architecture, which memory management is inspired by LSTM but adapted for multiple layers:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{3}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{4}$$

$$r_t = \text{Concat}[\phi_1(s_{1,t}), \phi_2(s_{2,t}), ..., \phi_L(s_{L,t})] \tag{5}$$

$$o_t = \sigma(W_o r_t + b_o) \tag{6}$$

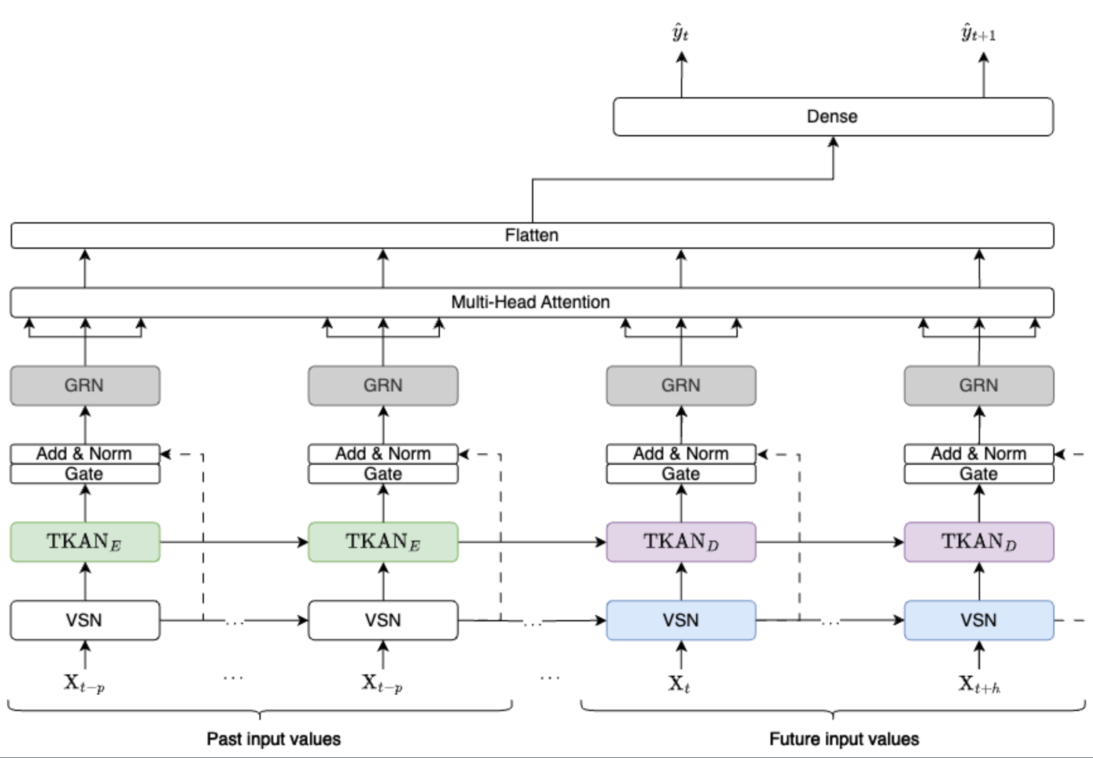
where the final cell and hidden states combine all layers:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{7}$$

$$h_t = o_t \odot \tanh(c_t) \tag{8}$$

TKAT: Temporal Kolmogorov-Arnold Transformer

The TKAT extends TKAN's capabilities by adopting a transformer-like architecture for sequence-to-sequence tasks.



SigKAN: Signature-Weighted Kolmogorov-Arnold Networks

SigKAN enhances KAN's capabilities by incorporating path signatures - a mathematical tool that captures geometric features of sequential data.

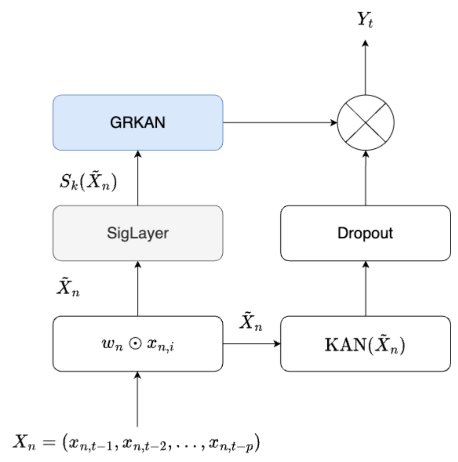


Figure 5. SigKAN layer architecture

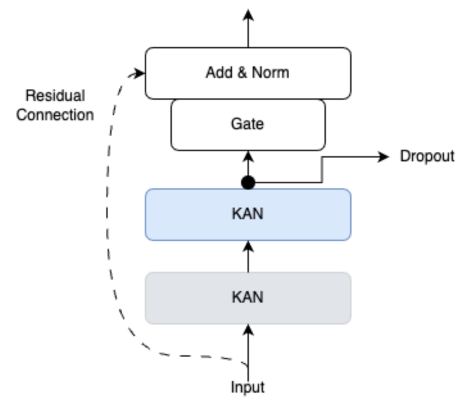


Figure 6. GRKAN Layer

The SigKAN model integrates path signatures with Kolmogorov-Arnold Networks (KAN) layers, combining a Gated Residual KAN (GRKAN) and a learnable path signature layer to enhance predictive capabilities. Key components include:

- **Gated Residual KAN:** Controls information flow through gated connections.
- **Learnable path signature layer:** Computes path signatures for each sequence with learnable parameters.

Path Signature Layer: Given an N-dimensional path $(X_t)_{t\in[0,T]}$, the first-order path signature $S(X)^n_{0,t}$ for a one-dimensional path $(X_{n,t})$ is:

$$S(X)^n_{0,t} = \int_0^t dX^n_s. \tag{9}$$

Higher-order terms involve iterated integrals of multiple paths. The second-order term $S(X)^{n,m}_{0,t}$ is:

$$S(X)^{n,m}_{0,t} = \int_0^t S(X)^n_{0,s}\, dX^m_s. \tag{10}$$

The k-th level signature is computed iteratively:

$$S(X)^{i_1,\ldots,i_k}_{0,t} = \int_0^t S(X)^{i_1,\ldots,i_{k-1}}_{0,s}\, dX^{i_k}_s. \tag{11}$$

The path signature $S(X)_{0,T}$ is the ordered set of all such terms:

$$\begin{aligned} S(X)_{0,T} = (1, S(X)^1_{0,T}, S(X)^2_{0,T}, \ldots, S(X)^N_{0,T}, S(X)^{1,1}_{0,T}, \\ S(X)^{1,2}_{0,T}, \ldots, S(X)^{N,N}_{0,T}, S(X)^{1,1,1}_{0,T}, \ldots). \end{aligned} \tag{12}$$

Gated Residual KAN (GRKAN): GRKAN uses gated residual connections to model complex temporal relationships. The GRKAN layer is defined as:

$$\begin{aligned} \text{GRN}_\omega(x) &= \text{LayerNorm}\left(x + \text{GLU}_\omega(\eta_1)\right), \\ \eta_1 &= \text{KAN}(\varphi_{\eta_1}(.), \eta_2), \\ \eta_2 &= \text{KAN}(\varphi_{\eta_2}(.), x). \end{aligned} \tag{13}$$

The GLU gating mechanism for input γ is:

$$\text{GLU}_\omega(\gamma) = \sigma(W_{4,\omega}\,\gamma + b_{4,\omega}) \odot (W_{5,\omega}\,\gamma + b_{5,\omega}), \tag{14}$$

where σ is the sigmoid activation, and $\odot$ denotes element-wise Hadamard product.

Learnable Path Signature Transformation: For each coordinate path X_n, we apply learnable weights w_n:

$$\tilde{X}_{n,i} = w_n \odot x_{n,i}. \tag{15}$$

The k-th order path signature is then:

$$S(\tilde{X}) = \left(1, \left(\int_0^1 d\tilde{X}^{i_1}_{t_1}\right)_{1\le i_1\le N}, \left(\int_0^1\int_0^{t_1} d\tilde{X}^{i_2}_{t_2} d\tilde{X}^{i_1}_{t_1}\right)_{1\le i_1,i_2\le N}, \ldots\right). \tag{16}$$

The transformed path signature vector $S(\tilde{X})$ is then:

$$S(\tilde{X}) = \left[S(\tilde{X})_1, S(\tilde{X})_2, S(\tilde{X})_3, ...\right]. \tag{17}$$

Output Layer The GRKAN output h_s is normalized and used as weights:

$$\psi = \text{SoftMax}(h_s). \tag{18}$$

The global SigKAN output is:

$$\psi \odot \text{KAN}(\tilde{X}). \tag{19}$$

Generalized SigKAN Network The SigKAN network structure across layers is:

$$\begin{aligned} h_0 &= x, \\ h_j &= \text{SoftMax}(\text{GRKAN}_j(S(\tilde{h}_{j-1}))) \odot \text{KAN}_j(h_{j-1}), \\ y &= h_L, \end{aligned} \tag{20}$$

where $j = 1, 2, \ldots, L$ denote the position of the layer. L is the final layer of the network.

Presented Models performance depending on horizon

Table 1. R^2 Average: TKAT vs Benchmark for Volume Prediction

Time	TKAT w. TKAN	TKAT w. LSTM	TKAN	GRU	LSTM
1	0.30519	0.29834	0.33736	0.36513	0.35553
3	0.21801	0.22146	0.21227	0.20067	0.06122
6	0.17955	0.17584	0.13784	0.08250	-0.22583
9	0.16476	0.15378	0.09803	0.08716	-0.29058
12	0.14908	0.15179	0.10401	0.01786	-0.47322
15	0.14504	0.12658	0.09512	0.03342	-0.40443