Kolmogorov-Arnold Networks

Artur M. Oliveira
Universidade Federal de Minas Gerais
PPGEE
Belo Horizonte, Brasil
oliveiraarturm@hotmail.com

***Abstract*—Inspired by the Kolmogorov-Arnold representation theorem, the authors of [1] developed networks called KANs (Kolmogorov-Arnold Networks). KANs are an alternative to multilayer perceptrons due to their better performance and interpretability. Furthermore, KANs require smaller computation graphs than MLPs. KANs are promising alternatives to MLPs, opening up a series of opportunities for the fields of Artificial Intelligence and science. This paper will analyze such networks, highlighting their main theoretical foundations and their performance when compared to multilayer perceptron networks.**

I. Introduction

The perceptron networks, proposed by Rosenblatt et al. in 1957 [2], can be considered the starting point of neural networks. Despite the initial success, the perceptron was sidelined due to some limitations at the time, such as computational power and the perceptron's inability to solve complex computational problems, with the most famous being XOR, where the perceptron and its separating hyperplane were unable to correctly classify the data.

With the development of the backpropagation algorithm proposed by Hinton et al., training multi-layer networks, such as the MLP (Multi-Layer Perceptron), became computationally more accessible and easier to implement. MLPs started to be widely adopted.

Unlike the traditional perceptron, MLPs can map highly nonlinear functions and still have excellent generalization error. This is due to the fact that MLP networks make a superposition of a series of nonlinear activation functions.

However, despite being widely used, MLPs have some disadvantages. In models such as transformers, for example, MLPs consume a large part of the parameters that are not directly related to data embedding [1]. Furthermore, they tend to be difficult to interpret, functioning as a sort of "black box," especially when compared to other parts of the model, such as the attention layers. This makes it more challenging to understand why certain predictions are made without the aid of additional analysis tools. Finally, another disadvantage of MLP networks is that in applications like transformers, MLP networks require large computation graphs.

Motivated by these reasons, Liu et al. [1] developed the KAN networks. These networks are an alternative to MLPs because they offer greater interpretability, helping scientists to discover and rediscover opportunities in the fields of mathematics and physics. Another point mentioned is the accuracy and efficiency of the new approach, surpassing MLPs in both areas.

Kolmogorov-Arnold networks are based on the Kolmogorov-Arnold theorem, which states that any continuous function (within a finite interval) of multiple variables can be represented by a finite number of univariate functions [3].

Although this theorem is relatively old, it has never been fully generalized for multi-layer neural networks. This is the major contribution that [1] claims to have made.

Despite the Kolmogorov-Arnold theorem being valid, univariate functions can become complex. To avoid this issue, KANs restrict themselves to using only splines. Thus, the networks may not express the target function with 100% accuracy, but they provide a more than reasonable approximation. Therefore, KANs combine the advantages of splines and MLPs, overcoming the limitations of both. Splines are accurate for low-dimensional functions, allow local adjustments and resolution changes, but face the problem of the curse of dimensionality (COD) due to the difficulty in exploring compositional structures. On the other hand, MLPs, with their feature-learning capability, suffer less from COD but are less accurate in low dimensions because they do not optimize univariate functions well. KANs integrate MLPs in the outer part and splines in the inner part, allowing not only the learning of compositional structures but also the optimization of univariate functions with high precision.

Although recent, the paper by Liu et al. [1] has already garnered many citations, and as reported in [4], several other applications of KANs beyond those mentioned in [1] are already being developed. Examples can be found in the fields of computer vision and medical science. There is also potential for application in areas such as finance, environmental monitoring, medical diagnostics, etc.

II. Organization

In Section III, a literature review will be presented, considering the main works and some applications of KANs.

In Section IV, an analysis of the architectures of KANs will be provided. In Section V, the methodology of the experiments conducted for the comparison between MLP and KAN networks will be presented. In Section VI, the results obtained from this analysis and their discussion will be shown. Finally, Section VII will provide the conclusion.

III. Literature Review

Although it seemed extremely useful, the Kolmogorov-Arnold theorem was considered useless and unfeasible in

neural networks for a long period. In [5], in the article entitled *Representation properties of networks: Kolmogorov's theorem is irrelevant*, the Kolmogorov-Arnoldtheorem was considered irrelevant because univariate functions can become complex to the point of being non-smooth and even fractal. In [6], the authors corroborated this point. Other attempts to apply it to neural networks were made in the late 20th century, but the computational power and algorithm optimization did not allow for future advancements. More specifically, the computational resources available at the time could not effectively handle the large number of function superpositions and the complex nonlinear relationships, limiting the practical application of KANs [7] [8].

In [9], the authors show that KANs expand the scope of optimization by introducing learnable activation functions. This means that, during the optimization process, not only the weights and biases of the network are adjusted, but also the parameters of the activation functions. Although this increases the complexity of training, it also provides greater flexibility, allowing the KAN to more effectively capture complex relationships in the data.

In [10], Dhiman states that traditional neural networks often demand a large number of parameters and computational resources, especially in deep convolutional networks (CNNs). The KAN, by using learnable univariate functions, can considerably reduce the number of parameters. For example, Convolutional KANs (Conv-KAN) achieve high accuracy using far fewer parameters compared to traditional CNNs, resulting in a reduction in computational costs.

Other works, besides [1], have also proven the superiority of KANs over MLPs. In [11], the authors propose the use of Kolmogorov-Arnold Networks for time series prediction, taking advantage of their adaptive activation functions. The results show that KANs outperform traditional MLP networks in a real-world satellite traffic prediction task, offering higher accuracy with fewer parameters. The motivation of the study is to explore the flexibility of KANs to improve predictive modeling, especially in environments with limited computational resources.

The study in [12] explored the application of KANs in computer vision tasks, a field that is still underexplored. The authors tested the KAN on various datasets, such as MNIST, CIFAR10, and CIFAR100, and compared its performance with MLP-Mixer and ResNet-18. The results showed that the KAN outperformed the MLP-Mixer in CIFAR10 and CIFAR100 but lagged slightly behind ResNet-18. The motivation of the work was to demonstrate the potential of KANs for visual tasks, offering a promising alternative.

In contrast, in [13], the authors investigated the impact of noise in the data on the performance of KANs. It was observed that even a small amount of noise could significantly degrade the network's performance. To mitigate this effect, Shen et al. proposed an oversampling technique combined with filtering to remove noise, using kernel filtering with diffusion maps to preprocess the data before training. The experiments showed that by adding i.i.d. noise with a fixed signal-to-noise ratio and increasing the amount of training data, the test loss (RMSE) followed a performance trend proportional to $\mathcal{O}\left(\frac{1}{r^2}\right)$ as the number of data points increased. It was concluded that the combination of oversampling and filtering could reduce the negative effects of noise, but determining the optimal variance for filtering and substantially expanding the training data increases costs, as it is necessary to multiply the dataset several times in relation to the initial clean data. Thus, noise reduces the effectiveness of Kolmogorov-Arnold networks.

IV. Analysis of Architectures

The initial problem of KANs is to learn a function such that its output is approximately equal to the label for all points. According to the equation below, it is enough to learn the functions $\phi_{q,p}$ and Φ_q.

$$f(\mathbf{x}) = f(x_1, \ldots, x_n) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right)$$

The functions to be learned can be parameterized by B-spline curves. A KAN can consist of several layers. The form of the KAN can be represented as follows:

$$[n_0, n_1, \ldots, n_L]$$

Where n_i is the number of nodes in the i-th layer of the network. Let i denote the i-th neuron and l denote the l-th layer. The i-th neuron of the l-th layer is given by (l, i), and the activation value of the neuron (l, i) is denoted by $x_{l,i}$. The activation function that connects (l, i) and $(l+1, j)$ is denoted by:

$$\phi_{l,j,i}, \quad l = 0, \cdots, L-1, \quad i = 1, \cdots, n_l, \quad j = 1, \cdots, n_{l+1}.$$

The pre-activation of $\phi_{l,j,i}$ is simply $x_{l,i}$; the post-activation of $\phi_{l,j,i}$ is denoted by $\tilde{x}_{l,j,i} \equiv \phi_{l,j,i}(x_{l,i})$. The activation value of the neuron $(l+1, j)$ is simply the sum of all post-activation inputs:

$$x_{l+1,j} = \sum_{i=1}^{n_l} \tilde{x}_{l,j,i} = \sum_{i=1}^{n_l} \phi_{l,j,i}(x_{l,i}), \quad j = 1, \cdots, n_{l+1}.$$

In matrix form, this is written as

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

where $\mathbf{\Phi}_l$ is the matrix of functions corresponding to the l-th KAN layer. A general KAN network is a composition of L layers: given an input vector $\mathbf{x}_0 \in \mathbb{R}^{n_0}$, the output of the KAN is

$$\text{KAN}(\mathbf{x}) = (\mathbf{\Phi}_{L-1} \circ \mathbf{\Phi}_{L-2} \circ \cdots \circ \mathbf{\Phi}_1 \circ \mathbf{\Phi}_0)\mathbf{x}.$$

An example architecture can be seen in the figure below:

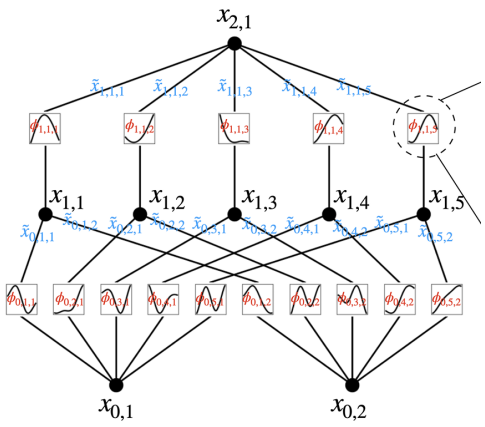

Fig. 1. Architecture of a KAN with two inputs and one output

In the case of a KAN (Kernel-based Artificial Network), the total number of parameters grows in the order of

$$O(N^2L(G+k)) \sim O(N^2LG).$$

In contrast, an MLP (Multilayer Perceptron) with depth L and width N requires only

$$O(N^2L)$$

parameters. This result seems to suggest that the MLP network uses fewer parameters, however, KAN networks have a smaller width N compared to MLP networks. This not only reduces the total number of parameters but also improves generalization capacity and promotes better interpretability.

As shown in the figure below, KAN networks return a lower generalization error than MLPs for the same number of parameters.

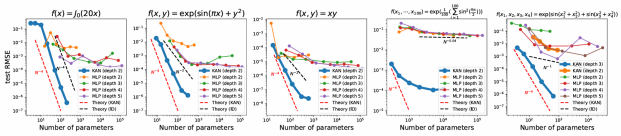

Fig. 2. Comparison of KANs with MLPs in five illustrative examples

Another point mentioned in [1] is the ability of KANs to finely adjust by increasing the resolution of the splines. As the grid becomes finer, each spline segment covers a smaller area of the domain, reducing approximation errors.

A consideration made by the authors is that small KANs generalize better. This was proven by comparing a KAN [2,1,1] with a [2,5,1]. Overfitting probably occurred.

Lastly, the authors make considerations to make KANs more interpretable. This becomes valuable when determining the ideal size for a KAN. Two of the simplification techniques mentioned in [1] are sparsification and pruning. In sparsification, a process analogous to the L1 norm of MLP networks is used; however, the L1 norm in the case of KANs is insufficient, and entropy-based regularization is needed. In pruning, an input and output score is defined according to the equation below:

$$I_{l,i} = \max_k \left(|\phi_{l-1,i,k}|\right),$$
$$O_{l,i} = \max_j \left(|\phi_{l+1,j,i}|\right),$$

neurons with scores below a certain threshold are pruned from the network.

V. Methodology

In order to compare the efficiency of KANs, several comparative experiments were conducted between KANs and MLPs. The execution time, accuracy, and number of parameters were compared. The KANs were configured to present 5 hidden units and were trained with $k = 3$, grid = 3, and $\lambda = 0.001$. The MLPs were configured to have two internal layers, each with 50 neurons. The training method for the MLPs was backpropagation with stochastic gradient descent as the optimizer.

For the KANs, the implementation was based on [1], using the library provided by the authors on GitHub.

For the comparative analysis, the datasets used were the approximations of the following functions:

- $f(x,y) = e^{\sin(\pi x)+y^2}$
- $f(x,y) = e^{\sin(\pi x)+y^2} + \epsilon\,, \quad \epsilon \sim \mathcal{N}(\mu,\sigma^2).$
- $f(x,y) = \sin(e^{x^2-y}) + \ln(1+x^2+y^2) + \cos(\pi xy).$
- $f(x,y) = x^2e^{-y} + \cos(\pi xy)$
- $\cos(\pi xy) + \epsilon\,, \quad \epsilon \sim \mathcal{N}(\mu,\sigma^2).$

For all cases, 2000 points for x and y were generated. The data were split 50/50 for training and testing.

VI. Results and Discussion

For the function $f(x,y) = e^{\sin(\pi x)+y^2}$, the results were as follows:

- KAN
  - Number of parameters: 75
  - Training time: 39s
  - Test error: 0.02
- MLP
  - Number of parameters: 2751
  - Training time: 65.38s
  - Test error: 0.02155

For the function $f(x,y) = e^{\sin(\pi x)+y^2} + \epsilon, \quad \epsilon \sim \mathcal{N}(1.85, 0.1^2)$, the results were as follows:

- KAN
  - Number of parameters: 75
  - Execution time: 27s
  - Test error: 0.105
- MLP
  - Number of parameters: 2751
  - Training time: 95.18s
  - Test error: 0.102

For the function $f(x, y) = \sin(e^{x^2-y}) + \ln(1 + x^2 + y^2) + \cos(\pi xy)$, the results were as follows:

- KAN
  - Number of parameters: 75
  - Execution time: 29s
  - Test error: 0.0346
- MLP
  - Number of parameters: 2751
  - Training time: 116.134s
  - Test error: 0.0174

For the function $f(x, y) = x^2 e^{-y} + \cos(\pi xy)$, the results were as follows:

- KAN
  - Number of parameters: 75
  - Execution time: 25s
  - Test error: 0.0256
- MLP
  - Number of parameters: 2751
  - Training time: 53.79s
  - Test error: 0.025989

For the function $\cos(\pi xy) + \epsilon, \quad \epsilon \sim \mathcal{N}(0.6, 0.1^2)$, the results were as follows:

- KAN
  - Number of parameters: 75
  - Execution time: 28s
  - Test error: 0.109
- MLP
  - Number of parameters: 2751
  - Training time: 66.259s
  - Test error: 0.104

The architecture of the KANs used in the tests had the following form:

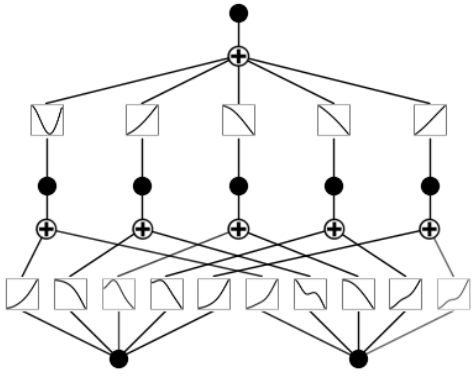

Fig. 3. Architecture of the KANs used in testing

As can be seen from the results, KANs were faster to train in all situations compared to MLPs. This demonstrates that the performance claims presented in [1] are indeed valid.

Another highlight was the number of parameters. The KAN networks managed to achieve test error performance comparable to MLPs with significantly fewer parameters. Although in some situations the test error (expressed in RMSE) was slightly inferior, the difference was not significant.

Lastly, it is worth highlighting the accuracy of KAN networks in the presence of noise. Despite the noise negatively affecting the test error, as pointed out in [13], the result remained acceptable and did not fall short compared to MLPs.

VII. Conclusions

It can be concluded, therefore, from the experiments and the literature review, that Kolmogorov-Arnold Networks demonstrate promising performance compared to Multilayer Perceptron Networks, offering significant advantages in reducing the number of parameters and computational costs. The experiments showed that KANs maintain comparable accuracy even in the presence of noise in the data. Additionally, their ability to learn adaptive univariate functions makes them an attractive alternative in several applications.

However, since it is a relatively new concept, it still requires more robust testing. In summary, KANs offer an innovative and powerful approach within the field of neural networks, with potential to revolutionize predictive modeling in various contexts, but the continuous improvement of their training and interpretation techniques will be crucial for their widespread adoption. The future of KANs is promising, and ongoing development in this area could open new frontiers for Artificial Intelligence applications.

VIII. Appendix

All codes used can be found on GitHub.

References

[1] Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljačić, T. Y. Hou, and M. Tegmark, "Kan: Kolmogorov-arnold networks," 2024. [Online]. Available: https://arxiv.org/abs/2404.19756

[2] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[3] J. Schmidt-Hieber, "The kolmogorov–arnold representation theorem revisited," *Neural Networks*, vol. 137, pp. 119–126, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608021000289

[4] T. Ji, Y. Hou, and D. Zhang, "A comprehensive survey on kolmogorov arnold networks (kan)," 2025. [Online]. Available: https://arxiv.org/abs/2407.11075

[5] F. Girosi and T. Poggio, "Representation properties of networks: Kolmogorov's theorem is irrelevant," *Neural Computation*, vol. 1, no. 4, pp. 465–469, 1989.

[6] T. Poggio, "How deep sparse networks avoid the curse of dimensionality: Efficiently computable functions are compositionally sparse," *CBMM Memo*, vol. 10, 2022.

[7] J. Braun and M. Griebel, "On a constructive proof of kolmogorov's superposition theorem," *Constructive Approximation*, vol. 30, no. 3, pp. 653–675, 2009.

[8] J. Cheon, "Improving computational efficiency in convolutional kolmogorov-arnold networks," *Neural Computing and Applications*, vol. 37, no. 1, pp. 15–30, 2024.

[9] ——, "Improving computational efficiency in convolutional kolmogorov-arnold networks," *Neural Computing and Applications*, vol. 37, no. 1, pp. 15–30, 2024.

[10] P. Dhiman, "Applications of kolmogorov-arnold networks in hyperspectral image classification," *Remote Sensing*, vol. 16, no. 2, pp. 205–220, 2024.
[11] C. J. Vaca-Rubio, L. Blanco, R. Pereira, and M. Caus, "Kolmogorov-arnold networks (kans) for time series analysis," 2024. [Online]. Available: https://arxiv.org/abs/2405.08790
[12] M. Cheon, "Demonstrating the efficacy of kolmogorov-arnold networks in vision tasks," 2024. [Online]. Available: https://arxiv.org/abs/2406.14916
[13] H. Shen, C. Zeng, J. Wang, and Q. Wang, "Reduced effectiveness of kolmogorov-arnold networks on functions with noise," 2024. [Online]. Available: https://arxiv.org/abs/2407.14882