Emirhan Burak Yılmaz

1- Kolmogorov-Arnold Representation Theorem

If f is a multivariate continous function on a bounded domain, then f can be written as a finite composition of continuous functions of a single variable and the binary operation of addition. Formally, for any continuous function $f:[0,1]^n \longrightarrow R$,

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

where $\phi_{q,p} : [0,1] \longrightarrow R$ and $\Phi_q : R \longrightarrow R$.

In a sense, the theorem showed that the only true multivariate function is the sum, since every other function can be written using univariate functions and summing.

The theorem is profoundly significant for function approximation because, unlike the Universal Approximation Theorem which requires networks of infinite width for infinity small error tolerance, the Kolmogorov-Arnold Representation Theorem provides an exact finite representation using only $2n+1$ nodes for a smooth/continous function having n input variables. The theorem achieves this through a hierarchical decomposition that transforms the n-dimentional approximation problem into a series of 1-dimentional leaning tasks, offering immunity to the curse of dimensionality since it operates in 1-dimentional spaces. Also it offers an inherent interpretability as each component function can be individually analyzed.

The Kolmogorov-Arnold Representation Theorem (KART) and the Universal Approximation Theorem (UAT) both guarantee function approximation but differ in structure and approach. First of all, KART decomposes a multivariate function into a sum of univarite functions, whereas UAT approximates functions via weighted sums of multivariate activations. Therefore their approximation mechanism is different:

$$f(x,y) = \Phi_1(\phi_{1,1}(x) + \phi_{1,2}(y)) + \ldots \Phi_5(\ldots) \qquad \text{(KART)}$$

$$f(x,y) \approx \sum w_i \sigma(w_i x + w_y y + b_i) \qquad \text{(UAT)}$$

Emirhan Burak Yilmaz

Another difference is that while KART offers interpretability, UAT ends up being black-box. Considering the use cases, KART is ideal for low-dimentional, symbolic problems such as physics equations, whereas UAT is better for high dimentional data such as images.

2- KAN Architecture

KANs replace tradicional fixed activation functions and learnable weights in neural networks with learnable univariate functions. Learnable activation functions exist on edges (which corresponds to the weights of a tradicional NN), and their sum is taken on nodes.

KANs approximate multivariate functions by composing multiple univariate functions (activations), where each function transforms a single variable before being combined and passed through additional univariate functions in deep layers. These univariate transformations are typically modeled using B-splines, which are smooth, flexible basis functions with learnable coefficient parameters. This design aligns with KART and provides a theoritically grounded and interpretable approach to learning complex functional relationships.

3- Comparison with DNNs

KANs offer a compelling alternative to standard neural networks by incorporating learnable activation functions parameterized by B-Splines. This structure enchances interpretability, as each input's transformation can be directly visualized and analyzed. Moreover, KANs benefit from internal degree of freedom which introduced via spline based activations, enabling them to model complex nonlinearities effectively. On top of that, connecting layers also adds an external degrees of freedom, which helps to combine the univarite functions together for the approximation of multivariate function. Finally, because KANs are based on functional composition rather than deep stacking of layers, they can, in some cases, approximate target functions using fewer parameters than DNNs.

Emirhan Burak Yilmaz

Despite these advantages, KANs face notable limitations. Training is generally more expensive than for corresponding neural networks. This is because spline-based activation functions involve nontrivial computations and may not be as well-optimized for GPU acceleration as standard pointwise activations like ReLU. Furthermore, each input-output connection require a seperate spline transformation, which increases the number of parameters significantly, especially in high-dimensional settings. While KANs are promising for structured or tabular data with moderate dimensionality, scaling them to very high-dimensional inputs (e.g. images) remains challenging without architectural adaptations or dimensionality reduction.