COSC278 SP24

Professor: Yujun Yan

Group 13: Ray (Congrui) Yu, Xinyue Liang

May 21, 2024

Word Count: 2636

Validating KAN Model's Performance Advantage on Different Tasks Comparing to Other Machine Learning Models

Ray (Congrui) Yu, Xinyue Liang

Section 0. Abstract

In the recently published work "KAN: Kolmogorov-Arnold Networks" (Ziming et. Al, 2024), researchers from MIT proposed the KAN as a powerful alternative to the Multi-Layer Perceptron (MLP). The authors claimed that KANs require less complex model design and fewer neurons with fewer parameters, being more accurate and more interpretable, as well as possessing faster neural scaling laws.

In this essay, we want to examine the stated performance advantage through comparing KAN, MLP and Random Forests' performance on different types of datasets

including mathematical model estimation as well as simulation for real world physical models.

Based on our result, we reached a conclusion that the KAN model does require significantly less neurons and layers thus resulting in simpler model design. Meanwhile, KAN has a significantly higher accuracy on mathematical models but performs relatively close and sometimes worse on simulation for real world physical models and classification problems when compared to MLP models.

Beyond that, we did further analysis on what resulted in the performance difference on different types of datasets and proposed potential areas of usage that can benefit from KANs' advantages.

Section 1. Introduction

According to the information provided by Ziming et al. (2024), their idea starts from the theorem established by Vladimir Arnold and Andrey Kolmogorov, which is: "a multivariate continuous function f on a bounded domain could be represented by a finite composition of continuous functions of single variable along with the binary operation of addition." The theorem is called "Kolmogorov - Arnold representation theorem."

The KAN model has certain similarities with traditional MLP models such as they are both fully connected. On the other hand, differences also exist including learnable activation functions in KAN instead of the linear weight matrices in MLP.

We want to perform the validation process on KAN model in this essay since the performance advantage on KAN versus tradition model is large: according to the article (Ziming et al, 2024), a 2 Layer 10 neuron each layer KAN model is “100 times more accurate” and “100 times more parameter efficient” than a 4 Layer width-100 MLP model. Also, the original article seems to put less emphasis on the dataset they perform tasks on. Thus, we want to confirm the ability of KAN to generalize on different tasks.

The main objective of this study is to examine the performance of KAN models on different tasks, validating the performance difference of KAN mentioned in the original article compared to other models. And beyond that, we might be able to find areas and tasks which are suitable for KAN models’ advantages.

This report will first further introduce the theoretical background information on KAN including the model architecture, how they are different from traditional models and how to construct a KAN model. We will also go through the comparison models such as Random Forest and MLP briefly. Also, the dataset we will be using will be introduced. Then we will illustrate the method and tools we applied as well as the whole experiment process. Finally, we will present our conclusion and examine related and possible future works. The ending part would be the work distribution of each group member on this project.

Section 2. Preliminaries

As mentioned in section 1, the KAN model is based on the “Kolmogorov – Arnold representation theorem” proposed by Vladimir Arnold and Andrey Kolmogorov. The theorem could be represented in the following simpler equation for two inputs x1 and x2:

$$f(x1, x2) = \Phi 2(\varphi 2,1(\varphi 1,1(x1) + \varphi 1,2(x2))) \quad \text{(Isaak, 2024)}$$

“ Here, φ1,1 and φ1,2 are specific univariate functions for each input, combined and then processed through another function Φ2 in the subsequent layer.” The univariate functions are acting as weights as well as activation functions. To be more specific, according to Ziming et al. (2024), the 1D functions are parametrized as B _ spline curves with learnable coefficients of local B – spline basis functions.

The major model we want to compare to is Multi – Layer Perceptirons (MLP), since it’s the model mentioned most times in the original article as a performance benchmark. The MLP , which is well known and studied, is based on the universal approximation theorem and is a multi-layer neural network consisting of fully connected neurons and non linear kind of activation functions.

Beyond MLP , we also tested KANs’ performance compared to the Random Forest and even linear regression models as a baseline benchmark. Random Forest is normally composed of multiple decision trees (classifiers in this case) and then trained through the Classification and Regression Tree (CART) algorithm (IBM, 2024).

Moreover, we performed linear regression on each dataset to get a basic idea on aspects such as linearity of the datasets.

As for the choice of datasets, we choose 5 different datasets with different types of tasks (classification and regression) varied from mathematical models to real world simulation datasets.

Section 3. Proposed Method

Firstly, for each dataset, we tune the KAN model as well as the other three comparison models by running each model under different hyper parameter settings. We also change the model design such as trying different layer combinations and neuron numbers along this process until we

reach an optimal state where the validation accuracy is highest; training accuracy is not significantly higher than validation accuracy (which means the model does not show significant sign of overfit), and the model design is simple when satisfying the two standards above.

We tested and considered the comparison metrics proposed by the original article. However, we found out that a 2 layer KAN is already approaching optimal but a 4 layer MLP could still be far from its optimal design. So we ended up choosing to compare the models in their optimal settings.

We also performed feature engineering on the later 3 real world physics simulation datasets. We left out less irrelevant features such as id numbers. However, we also remove certain data which is highly correlated with the target to preserve and test how the KAN model handles non linearity.

For both mathematical model datasets and real world event prediction datasets, we choose Mean Absolute Error (MAE), Mean Square Error (MSE) and R^2 as our metrics for evaluating accuracy performance. On the classifier dataset, we use accuracy, F1 score and confusion matrices to evaluate the model performance.

Please note that we did not strictly record the physical resource used during the experiments. But there was no outstanding time difference for training KAN to converge on each dataset when compared to other models. The experiments were performed on the same machine equipped with a 3080 Ti graphics card.

For the KAN model, we used the architecture code provided by the authors of the original package (Ziming et. al, 2024). As for MLP and RF, we used a package provided by SciKit – Learn. We implemented the linear regression code on our own, it was a re-use of our COSC 274 course assignment code.

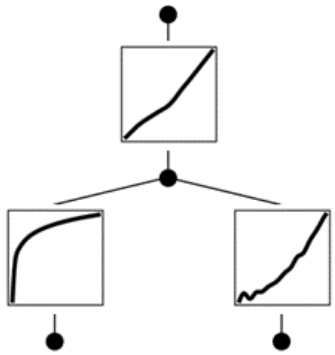
For the five datasets, we applied the models of following design:

Dataset 1: Math model: z_1 = np.log(x_14) + y_1 * y_1:**

Random Forest: depth 3

` MLP: 8 Layer MLP with 1411 neurons and 347867 parameters

KAN: 2 Layer KAN with 3 neurons and 3 parameters. (grid = 15, steps = 100)

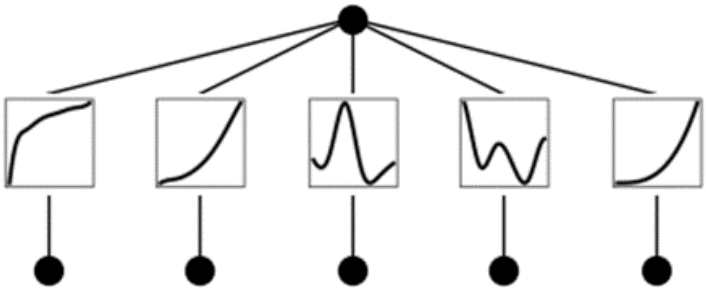


Dataset 2: z_2 = np.log(x_14) + np.exp(y_1 * y_1) + np.sin(a_1**b_1) + c_1**3:**

Random Forest: depth 3

MLP: Same Structure

KAN: 1 Layer KAN with 5 neurons and 5 parameters (grid = 5, steps = 100)



Dataset 3: CERN electron collision prediction:

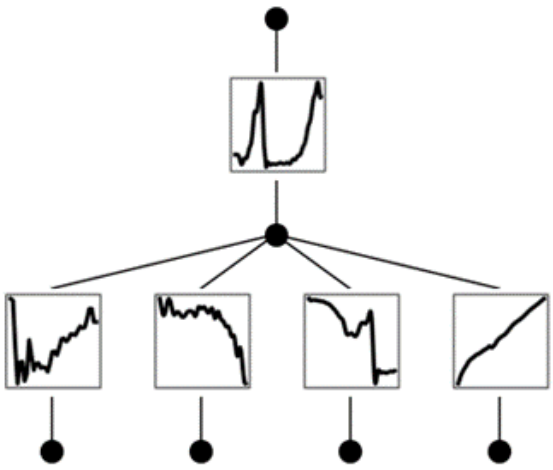
MLP: Same Structure (600 epochs)

KAN: 3 layer KAN with 38 neurons and 38 parameters (grid = 10, steps = 200)

Dataset 4: Solar Radiation Dataset

MLP: Same Structure (600 epochs)

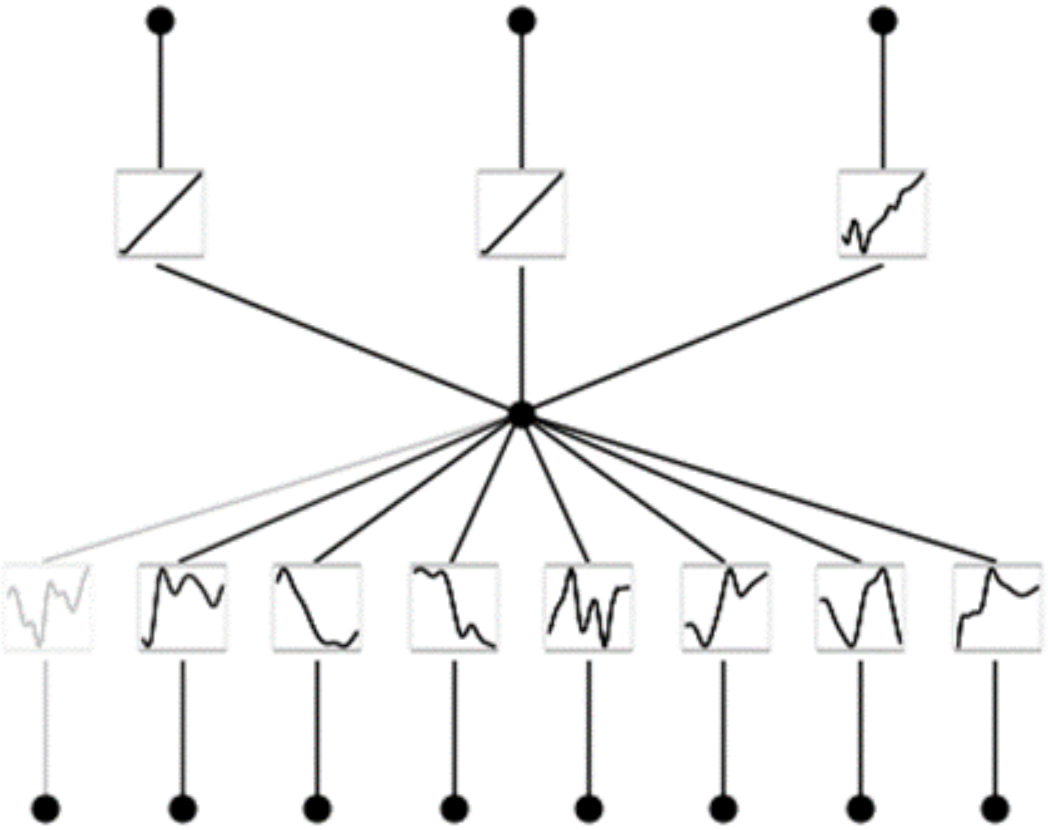
KAN: 2 layer KAN with 5 neurons and 5 parameters (grid = 30, steps = 150)



Dataset 5: Stellar object classification

MLP: 6 layer MLP 632 neurons, 82755 parameters, trained for 2000 epochs.

KAN: 2 layer KAN with 11 neurons and 11 parameters



Section 4. Experiments

In the experiment, we first want to find out if KAN is indeed approximating functions effectively as described in the original paper. So we first used it against a simple nonlinear function **z_1 = np.log(x_1**4) + y_1 * y_1** with x_1 and y_1 being randomly and uniformly simulated 1000 data points between 0 to 10 and 0 to 1 respectively. The results were indeed exceptional as it with only 3 parameters has more than 10 times less MSE than the 8 layered MLP with over 300,000 parameters.

$$z = log(x^4) + y^2$$

		MAE	MSE	R^2
Linear Regression		1.10875	2.20419	0.80772
Random Forest	Depth = 3	0.34699	0.21538	0.98121
MLP	347867 Parameters, 1411 Neurons	0.03492	0.00829	0.99927
KAN	3 Parameters, 3 Neurons, 15 Grids	0.01892	0.00077	0.99993

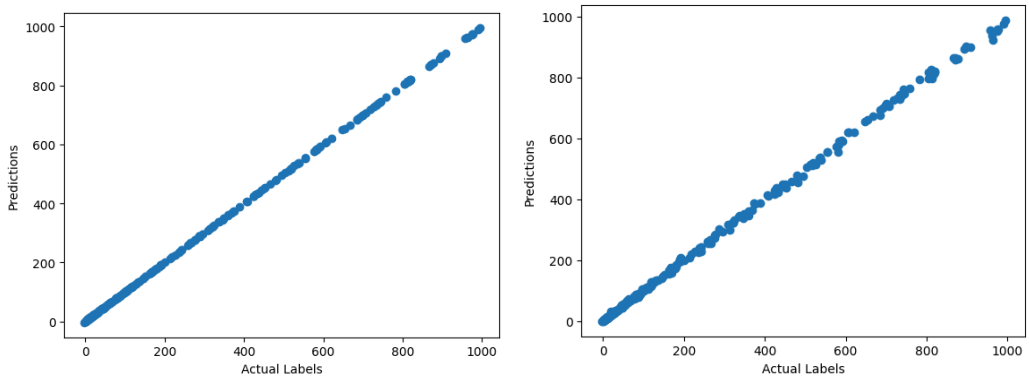
As a result, we proposed an even “more” non-linear mathematical function to test KAN’s ability. We chose the function **np.log(x_1**4) + np.exp(y_1 * y_1) + np.sin(a_1**b_1) + c_1**3** and also randomly and uniformly simulated 1000 data points for a_1, b_1, and c_1. The results were even more exceptional as the KAN with only 5 parameters is able to have almost 100 times less

MSE than the same MLP used in the prior function estimation.

$$z = log(x^4) + e^{y^2} + sin(a^b) + c^3$$

		MAE	MSE	R^2
Linear Regression		98.90166	13084.03732	0.84710
Random Forest	Depth = 3	18.29563	477.44645	0.99442
MLP	347867 Parameters, 1411 Neurons	4.30174	45.30521	0.99947
KAN	5 Parameters, 5 Neurons, 5 Grids	0.52789	0.53562	0.999993

If we take a look at KAN's predictions (plot on the left) vs the actual labels, we can clearly see that it forms an almost perfect y=x straight line as they almost have the exact same values with a R-squared indefinitely reaching 1. Compared to MLP's predictions (plot on the right), KAN's estimations are simply outperforming its counterpart.



After looking at KAN's exceptional performance with mathematical functions, we decided to look at KAN's prediction abilities in real world datasets, especially datasets with non-linearities and are governed by some mathematical functions, such as real-world physics simulation/observation datasets.

The first dataset we ran the experiment on is a dataset from 100,000 dielectron events coming from the CERN large hadron collider. (McCauley, Thomas; (2014). https://opendata.cern.ch/record/304) It includes features such as the total energy, the direction and traverse of momentum, pseudorapidity, the phi angle, as well as the charge of both electrons of collision. We want to use this information to predict the invariant mass of the two electrons after the collision.

CERN Electron Collision Dataset

		MAE	MSE	R^2
Linear Regression		14.66096	402.36411	0.37944
Random Forest	Depth = 3	14.07770	363.72032	0.43904
MLP	347867 Parameters, 1411 Neurons	1.87026	7.18147	0.98835
KAN	38 Parameters, 38 Neurons, 10 Grids	3.51792	30.56056	0.95042

The results were as expected such that MLP performs greatly over simpler models like a linear regression or a 3-depth random forest. MLPs are designed to handle these complex

multidimensional data, such as this real world electro collision data. However, KAN's results seemed to be significantly worse than the MLP despite as high as 38 parameters being added. From this dataset, it can be seen that KAN's performance after adding noises and measurement errors is likely to degrade from its almost perfect predictions on mathematical functions.

Furthermore, we wanted to test KAN's performance on an even simpler dataset with more straightforward physics properties and compare it to the MLP. So we have this solar radiation (https://www.kaggle.com/datasets/dronio/SolarEnergy) dataset that has only 5 relevant features with very explainable and simple physics properties (many features such as object identifiers were removed).

	Radiation	Temperature	Pressure	Humidity	WindDirection(Degrees)	Speed
Radiation	1.000000	0.734955	0.119016	-0.226171	-0.230324	0.073627
Temperature	0.734955	1.000000	0.311173	-0.285055	-0.259421	-0.031458
Pressure	0.119016	0.311173	1.000000	-0.223973	-0.229010	-0.083639
Humidity	-0.226171	-0.285055	-0.223973	1.000000	-0.001833	-0.211624
WindDirection(Degrees)	-0.230324	-0.259421	-0.229010	-0.001833	1.000000	0.073092
Speed	0.073627	-0.031458	-0.083639	-0.211624	0.073092	1.000000

However, after we did the correlation analysis, temperature has an extremely high correlation with our predictor "radiation", which makes even the simple linear regression a great model for prediction. We removed this feature in order to test KAN's ability on non-linear datasets. It can be seen that KAN is able to indeed outperform the MLP with just 5 parameters; however, this is not significant at all. But we can see that KAN still has the ability just like MLP to capture most of the non-linearities within the dataset unlike the linear regression and random forest models. The performance gain for KAN is just not enough to convince people that KAN can outperform the traditional neural network in real life datasets.

Solar Radiation Dataset

		MAE	MSE	R^2
Linear Regression		238.79620	88891.37504	0.107601
Random Forest	Depth = 3	154.26317	50901.95465	0.48898
MLP	347867 Parameters, 1411 Neurons	140.23567	45243.23317	0.54579
KAN	5 Parameters, 5 Neurons, 30 Grids	139.42025	44902.35546	0.54921

Lastly, we wanted to try the KAN model on a classification task that can correctly predict three types of objects in space (star, galaxy, or quasar). This dataset (fedesoriano. (January 2022). Stellar Classification Dataset - SDSS17. Retrieved [Date Retrieved] from https://www.kaggle.com/fedesoriano/stellar-classification-dataset-sdss17) contains physics properties (especially light wave properties) of the space objects and tries to determine what kind of objects emitted these light wave properties at various angles. Once again, we can see this dataset is nowhere close to a complex problem as a simple logistic regression is able to achieve an accuracy over 95%. As a result, the KAN with 11 parameters is able to outperform all the other machine learning models with just a slight higher accuracy. However, since other basic models also perform with a close accuracy, this classification task may not be a great one to fully test the abilities of the KAN or even the MLP (since the previous model of MLP tends to overfit with too many parameters, we had to drop the number of layers and parameters as a result).

Stellar Classification Dataset

		Accuracy	F1- Score	Confusion Matrix
Logistic Regression		0.9544	0.95	[[8550 149 192] [335 2508 4] [1 3 3258]]
Random Forest	Depth = 3	0.96233	0.96	[[8678 149 64] [351 2495 1] [0 0 3262]]
MLP	82755 Parameters, 632 Neurons	0.962	0.96	[[8643 145 103] [279 2564 4] [39 0 3223]]
KAN	11 Parameters, 11 Neurons, 20 Grids	0.96826	0.97	[[8682 123 86] [261 2582 4] [2 0 3260]]

Section 5. Related and Future Work

As the original research we set our project upon is the first one proposed the KAN model, which makes it the first article in the area, there were not many related previous papers available. The original research by Ziming et al (2024) is the only article which compares the KAN model to other machine learning models.

But there are several studies which influenced and inspired the development of KAN. For instance, the idea of trainable activation functions is also studied by Mohit et al. (2019) Also, another major motivation that propels Ziming (2024) and his team to develop KAN is to improve Mechanistic Interpretability (MI), which is an emerging field studied by researchers such as Kevin et al. (2022)

Our project aims to fill the relatively blank area of validating KAN's performance since it is a recently developed model and there are not many works published testing it.

We think that one potential field of future KAN research is dynamically combining KAN with traditional machine learning models, such as the work on Convolutional KANs by Alex et al (2024) and a project by akaashdash's team about Kansformers (2024).

Section 6. Conclusions

In this project, we come to a conclusion that although for mathematical models, KAN does achieve both the simplicity and accuracy performance claimed by the original article, on real world datasets with more noise it actually only achieves accuracy performance of the same level as other models we tested or even performs slightly worse.

Our results suggested that KAN could be very suitable for math computation. In the meantime, we analyzed the reason for KAN's disadvantage on real world datasets: Firstly, KAN requires more and higher quality feature engineering to achieve better prediction and classification results. Secondly, when setting up hyper parameters, KAN tends to be more sensitive to the settings than MLP models, so more precise settings could be helpful in improving KAN models' performance.

Due to the time span for this project, we did not put our attention on the physical resource for training each type of model, which could be a potential area for future research. Also, we did not try image, video or audio datasets, which is certainly a limitation of this project and we are looking forward to expanding the project to broader types of datasets.

Section 7. Division of Work

For this project, our two group members shares the tasks equally in the following manner:

1. Ray (Congrui) Yu's main role is to run the experiments and completing the section 4, which is the experiment related section in the report. He is also responsible for the first half of the presentation.

2. Xinyue Liang's task is to complete most of the report except section 4, (which means section 0-3 and 5-7). Meanwhile, he is responsible for the second half of the presentation.

The collaboration and the project as a whole is smooth. One thing we can possibly improve on in future projects is to increase the team size and test if our distribution of work is reasonable in larger teams.

Reference

Ziming, Liu. Arora, A., Shah, P., Dash, A., & Ghosh, S. (2024). **KANsformers: Improved Transformers for Knowledge Augmented Learning**. arXiv preprint arXiv:2404.19756. Retrieved from https://arxiv.org/html/2404.19756v1

Dash, A., Shah, P., Arora, A., & Ghosh, S. (2024). **KANsformers**. GitHub repository. Retrieved from https://github.com/akaashdash/kansformers

IBM. (n.d.). **Random forest**. Retrieved from https://www.ibm.com/topics/random-forest#:~:text=Random%20forest%20is%20a%20commonly,both%20classification%20and%20regression%20problems

Tepsich, A.(2024). **Convolutional-KANs**. GitHub repository. Retrieved from https://github.com/AntonioTepsich/Convolutional-KANs

fedesoriano.(2022, January). **Stellar Classification Dataset - SDSS17**. Retrieved [May, 21st], from https://www.kaggle.com/fedesoriano/stellar-classification-dataset-sdss17

dronio.(n.d.). **Solar Energy Data**. Retrieved from https://www.kaggle.com/datasets/dronio/SolarEnergy/data

McCauley, T.(2014). **Open Data at CERN**. Retrieved from https://opendata.cern.ch/record/304

Mwangi, I. (2023, March 16). **A simplified explanation of the new Kolmogorov-Arnold Network (KAN) from MIT**. Medium. Retrieved from https://medium.com/@isaakmwangi2018/a-simplified-explanation-of-the-new-kolmogorov-arnold-network-kan-from-mit-cbb59793a040

Goyal, M.,Goyal, R., & Lall, B. (2019). **Learning activation functions: A new paradigm for understanding neural networks**. arXiv preprint arXiv:1906.09529. Retrieved from https://arxiv.org/html/1906.09529

Meng, K., Bau, D., Andonian, A., & Belinkov, Y. (2022). **Locating and editing factual associations in GPT**. Advances in Neural Information Processing Systems, 35, 17359-17372.