FCKAN: Evaluating KAN for Time Series Classification and Extrinsic Regression

Gabriel da Costa Merlin[1], **Adilson Junior Alves Medronha**[1], **Diego Furtado Silva**[1]

[1]Institute of Mathematical and Computer Sciences, University of São Paulo
São Carlos, São Paulo, Brazil

{gabrielcmerlin, adilson.medronha, diegofsilva}@usp.br

Abstract. *Time Series Classification (TSC) and Time Series Extrinsic Regression (TSER) are critical tasks across diverse fields. While Fully Convolutional Networks (FCNs) effectively capture temporal dependencies, Kolmogorov–Arnold Networks (KANs) offer greater flexibility and interpretability. However, integrating KANs with temporal encoders and their application to regression tasks remain largely unexplored. This paper introduces FCKAN and Hybrid FCN-KAN, two novel architectures that combine FCNs and KANs for TSC and TSER. The first is an end-to-end model, while the second is a hybrid approach that leverages a pre-trained FCN as a feature extractor followed by a KAN. We conduct experiments on 147 benchmark datasets. For TSC, both architectures outperform non-temporal baselines and achieve competitive performance with FCNs. In TSER, although all models are statistically equivalent, temporal models consistently outperform non-temporal baselines.*

1. Introduction

Time Series Classification (TSC) and Time Series Extrinsic Regression (TSER) are essential tasks in various domains, including healthcare, finance, and engineering [Ismail Fawaz et al. 2019, Tan et al. 2021a]. Recent advances in deep learning strategies, particularly convolutional architectures such as Fully Convolutional Networks (FCNs), have shown strong performance by extracting temporal features directly from raw input. Kolmogorov–Arnold Networks (KANs) is a newly proposed model class that replaces scalar weights with learnable spline functions, offering improved flexibility and interpretability over traditional Multilayer Perceptron (MLP). While promising results have been demonstrated on classification benchmarks, KANs remain relatively unexplored in conjunction with robust temporal encoders for TSC and TSER tasks, which is particularly interesting, given that KANs do not possess intrinsic spatial or temporal awareness.

This paper proposes FCKAN[1], a novel architecture that integrates FCNs and KANs in a unified end-to-end framework for both TSC and TSER. We also introduce Hybrid FCN-KAN, a hybrid two-stage variant in which a pre-trained FCN serves as a temporal feature extractor for a downstream KAN. Our experiments on diverse benchmark datasets aim to evaluate the robustness and overall effectiveness of combining convolutional encoders with spline-based function approximators, thereby bridging the gap between high-capacity feature extraction and interpretable modeling.

[1]https://github.com/gabrielcmerlin/FCKAN

Our extensive experiments on TSC datasets revealed that the Hybrid FCN-KAN significantly outperformed non-temporal baselines (MLP and KAN) and achieved competitive performance against standalone FCNs. While the end-to-end FCKAN model also showed strong, statistically equivalent results, the hybrid approach often presented marginal gains, suggesting benefits from its independent feature extraction process. Our comprehensive evaluation of TSER datasets reinforces these findings regarding the superiority of temporal models. In TSER, while all models proved statistically equivalent, the FCN generally achieved the best average rank, and a clear advantage was consistently observed for all models designed with temporal recognition capabilities (FCN, Hybrid FCN-KAN, and FCKAN) compared to non-temporal baselines (MLP and KAN). It consistently highlights that effectively capturing temporal characteristics is crucial for superior performance in TSC and TSER tasks.

2. Related Work

Deep learning models have been instrumental in advancing time series analysis. For TSC, FCNs [Wang et al. 2016] and ResNets [Ismail Fawaz et al. 2019] have demonstrated strong performance by learning hierarchical representations from raw data. InceptionTime [Ismail Fawaz et al. 2020] further improved results through multi-scale convolutional blocks, while LITETime [Ismail-Fawaz et al. 2022] focused on improving efficiency with minimal performance degradation. TSER models typically adapt classification architectures by replacing the output layer with regression heads [Tan et al. 2021a]. However, regression tasks remain more challenging and less studied than classification.

KANs [Liu et al. 2024] is a recent architecture inspired by the Kolmogorov–Arnold representation theorem. Unlike conventional MLPs, KANs employ learnable univariate spline functions, offering more flexible and interpretable modeling. A recent study [Dong et al. 2024] showed that KANs can match MLP performance on TSC tasks, but their integration with temporal feature extractors such as CNNs has not been explored. Likewise, their potential for TSER tasks remains unknown. These gaps motivate our work, which investigates how KANs perform when combined with a convolution-based encoder in classification and regression experimental settings.

3. Background

The rapid advancement of sensor technologies has led to the generation of vast amounts of time series data across various important fields, including healthcare, finance, energy, and environmental monitoring [El Maachi et al. 2020, Ang and Seng 2016]. An effective and efficient analysis of these sequential data is crucial for accurate prediction and informed decision-making. This section provides a clear and concise definition of time series and presents key machine learning tasks related to this type of data.

Definition 1 *A time series is formally defined as an ordered sequence of n observations representing measurements taken over successive points in time. This sequence can be expressed as: $S = (s_1, s_2, ..., s_n)$, where each observation $s_t \in \mathbb{R}^d$ for all $t \in [1, n]$. When $d = 1$, the series is referred to as univariate, indicating a single measurement at each time point. Conversely, if $d > 1$, the series is multivariate, signifying multiple concurrent and possibly correlated measurements at each time point. It is typically assumed that consecutive observations in a time series are equally spaced in time, reflecting data collection often systematically performed at a constant frequency.*

Definition 2 *TSC is the task of assigning a given time series to one of several predefined discrete categories or classes [Bagnall et al. 2016, Ismail Fawaz et al. 2019]. Given a dataset of m time series $X = \{S_1, S_2, ..., S_m\}$, where each S_i adheres to the structure defined above, each series S_i is associated with a corresponding class label $y_i \in \{c_1, c_2, ..., c_k\}$. Here, $\{c_1, c_2, ..., c_k\}$ denotes the set of k possible classes. The primary objective of a TSC algorithm is to learn a mapping function $f : S \rightarrow Y$ that can accurately predict the class label y for any previously unseen time series S.*

Definition 3 *TSER aims to predict a continuous, real-valued target variable from an entire time series [Tan et al. 2021b]. This type of problem arises when the desired output is a measurable physical or abstract quantity, typically derived from complex patterns rather than a discrete category. It is crucial to distinguish TSER from time series forecasting: TSER predicts a single, continuous value for an entire time series, whereas time series forecasting predicts future values* ***within*** *a given time series [Lim and Zohren 2021]. Formally, for a dataset of m time series $X = \{S_1, S_2, ..., S_m\}$, each S_i is associated with a continuous target value $r_i \in \mathbb{R}$. A TSER algorithm aims to learn a function $g : S \rightarrow \mathbb{R}$ that accurately predicts the real-valued target r for any input time series S.*

4. Experimental Methodology

This section is divided into four parts to ensure a clear and pedagogically structured presentation: Datasets, Data Preprocessing, Machine Learning Models, and Experimental Setup. This separation enables a clearer exposition, facilitates reproducibility, and guides the reader through a coherent narrative from raw data to final evaluation.

4.1. Datasets

For TSC tasks, model evaluation was conducted using the UCR TSC Archive [Dau et al. 2018]. This comprehensive repository comprises 128 univariate datasets, each split into default training and testing sets. The diversity of these datasets, spanning domains such as healthcare and environmental monitoring, renders them highly suitable for robust model assessment. Furthermore, the archive presents notable challenges, including missing values, variable sample lengths, and datasets of limited size.

We evaluated TSER problems using the Monash, UEA & UCR TSER Repository [Tan et al. 2021a]. This comprehensive collection comprises 19 univariate and multivariate datasets from diverse domains, such as sentiment analysis, climate science, and energy monitoring. These datasets present similar complexities to those in the widely used TSC archive, including missing data and heterogeneous sample lengths.

4.2. Data Preprocessing

The dataset presents several challenges common to real-world data, including missing values and variable-length time series. Missing values can introduce bias or reduce statistical power, so we impute them with zero. Although a simple strategy, this approach is effective when absence implies null or negligible information. It assumes zero does not significantly distort the data distribution for the target task.

Additionally, many samples have variable lengths, complicating model input requirements. To standardize, we apply zero-padding by appending zeros to shorter sequences until all reach the same length. It ensures uniform input dimensions compatible with downstream models and prevents errors due to inconsistent data shapes.

4.3. Machine Learning Models

We compare several competitive models in the TSC task. MLP and FCN are well-established baselines in the literature, while KAN is a recently proposed model that we adopt as a novel baseline for fair comparison. Additionally, we introduce two new strategies: Hybrid FCN-KAN and FCKAN, which are our proposed novel architectures. For the TSER task, all architectures remain the same, except for the final fully connected layer, which uses $n =$ number of classes neurons in TSC and a single neuron in TSER.

4.3.1. Multilayer Perceptron (MLP)

The MLP is a classical deep learning baseline for time series tasks. It consists of fully connected layers that learn non-linear transformations of the input data through compositions of affine operations and element-wise activation functions. While MLPs lack explicit mechanisms for capturing sequential or local temporal structures, they can still approximate complex input-output mappings given sufficiently expressive architectures.

In time series applications, the input sequence is flattened into a one-dimensional vector, discarding temporal ordering and making the model permutation-invariant. As such, MLPs cannot exploit temporal inductive biases, unlike convolutional or recurrent architectures. Nevertheless, their simplicity and generality make them a valuable reference point for evaluating models incorporating temporal structure.

The MLP implemented in this study follows the architecture used in prior benchmarks [Wang et al. 2016]. As illustrated in Figure 1, it comprises three fully connected layers with 500 neurons each and ReLU activations. Dropout is applied after each layer with rates of 0.1, 0.2, and 0.2, respectively, followed by a final dropout layer with a rate of 0.3 before the output. The final layer uses a softmax activation for classification or a linear unit for regression. This deep and regularized design ensures that the MLP remains a robust baseline for non-sequential modeling in TSC and TSER tasks.

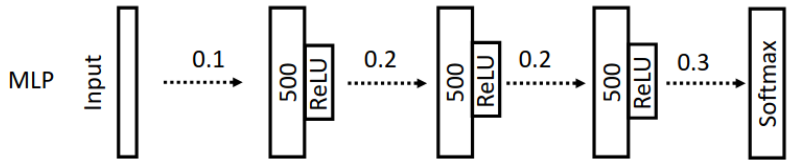

Figure 1. Architecture of the MLP model. Adapted from [Wang et al. 2016].

4.3.2. Fully Convolutional Networks (FCN)

FCNs are a class of neural architectures originally developed for image processing tasks, where spatial locality and translational invariance are critical. These models consist exclusively of convolutional layers and are capable of processing inputs of arbitrary size by producing dense, structured outputs. In the time series domain, the core architectural principles of FCNs have been successfully adapted to model temporal dependencies by treating time as the analog of spatial position [Wang et al. 2016]. The key insight lies in the interpretation of temporal signals as one-dimensional spatial data. In this formulation,

each point in time is treated as a spatial position, and the convolutional filters—originally designed to detect local spatial features—are carefully repurposed to extract local temporal patterns. This transfer of spatial inductive bias into the temporal domain enables FCNs to effectively capture local trends, motifs, and important dependencies in time series data, much like how they capture edges or textures in images.

Moreover, the hierarchical nature of FCNs allows for multi-scale temporal feature extraction: shallow layers capture short-term dependencies, while deeper layers integrate longer-term patterns. This makes FCNs particularly well-suited for TSC and TSER tasks, where both local and global temporal structures are informative. Additionally, due to their convolutional nature, FCNs are inherently more efficient than recurrent models, as they enable parallel processing and exhibit fewer constraints on temporal dependencies. Overall, the adaptation of FCNs to the time series setting leverages their spatial design principles to effectively model temporal structures, offering both interpretability and computational advantages in high-dimensional sequential data contexts.

4.3.3. Kolmogorov–Arnold Networks (KAN)

KANs are inspired by the fundamental Kolmogorov-Arnold representation theorem, a foundational result in function approximation. This theorem states that any continuous multivariate function can be expressed as a finite composition of univariate continuous functions and an addition operation [Liu et al. 2024]. Specifically, for a continuous function $f : [0, 1]^n \to \mathbb{R}$, it can be decomposed into Equation (1):

$$f(x_1, x_2, ..., x_n) = \sum_{q=0}^{2n} \phi_q \left(\sum_{p=1}^{n} \psi_{p,q}(x_p) \right) \tag{1}$$

where ϕ_q and $\psi_{p,q}$ are continuous univariate functions. This decomposition is highly significant because it clearly demonstrates that even very high-dimensional functions can be precisely represented using only simple operations on single variables and summations, thereby significantly simplifying their approximation.

In the context of KANs, this robust theoretical foundation is leveraged to design neural networks with architectures that explicitly mirror this decomposition. KANs aim to achieve efficient and interpretable representations while learning from high-dimensional data by structuring networks as nested univariate transformations combined with addition operations. This approach is advantageous for problems where the inherent dimensionality of the input space poses difficult challenges for conventional neural network designs.

4.3.4. Hybrid FCN-KAN Approach

KANs inherently lack temporal or spatial inductive biases, as they operate in a permutation-invariant manner and process inputs without considering sequential or spatial order. This limitation hinders their ability to model the temporal dependencies essential in time series data directly. To address this, the hybrid FCN-KAN approach employs a pre-trained FCN as a fixed feature extractor to provide temporally-aware representations

to the KAN. Specifically, the best-performing FCN model for each dataset is selected by assessing the loss in the training phase, its final fully connected classification layer is removed, and the remaining convolutional layers are frozen. Raw time series inputs are passed through this frozen FCN to produce latent features encoding temporal structure. The KAN is subsequently trained from scratch on these extracted features, allowing it to focus on modeling complex nonlinear relationships without the burden of learning temporal patterns from raw data. This strategy effectively separates temporal feature extraction from functional approximation, enabling a modular evaluation of KAN's expressive power when the temporal context is externally supplied.

4.3.5. FCKAN: Joint Training of FCN and KAN

The FCKAN approach integrates the FCN and KAN into a unified architecture trained end-to-end from scratch. Rather than using a pre-trained FCN or freezing any layers, the FCN's final fully connected layer is removed to enable a direct connection with the KAN, as shown in Figure 2. Both networks' parameters are jointly optimized, allowing them to co-adapt during training. The FCN dynamically learns to extract temporal features. Simultaneously, the KAN models nonlinear relationships on the evolving latent representations. The absence of frozen layers permits full backpropagation through the entire network, potentially resulting in a more expressive and flexible model that captures temporal dynamics and functional complexity synergistically. This fully integrated, joint learning strategy contrasts with the hybrid FCN-KAN, which treats temporal feature extraction and functional approximation as sequential, decoupled stages.

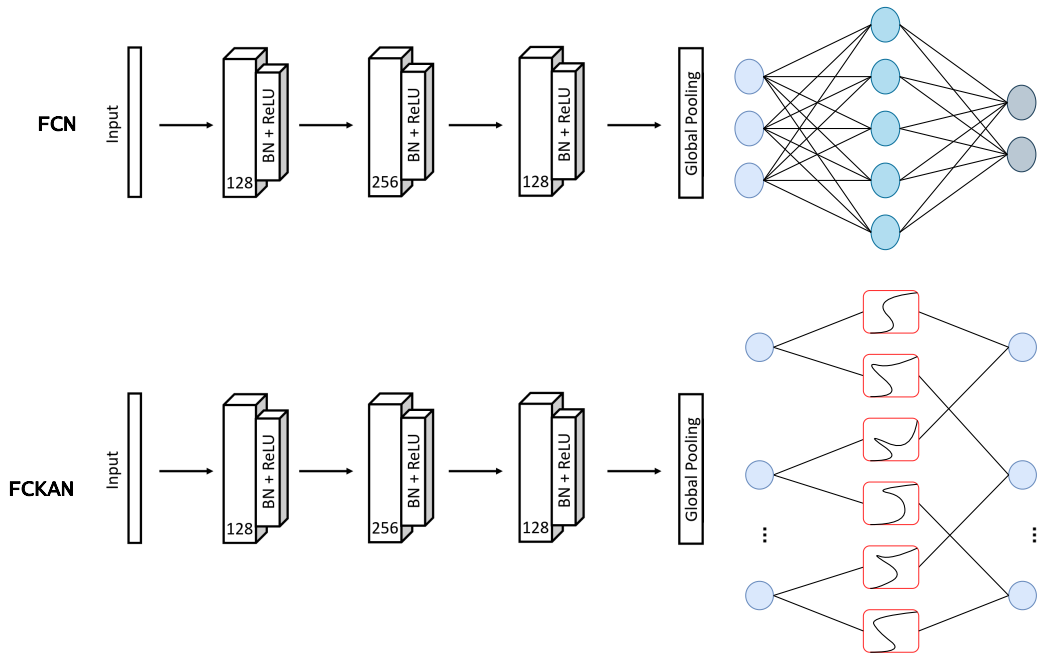

Figure 2. Comparison between FCN and FCKAN architectures: FCKAN replaces the MLP decision layer found in FCN with a KAN module.

4.4. Experimental Setup

We implemented all experiments for TSC and TSER tasks in PyTorch [Paszke et al. 2019]. The MLP was trained for 100 epochs using the Adam optimizer with a learning rate of 1×10^{-3} and the FCN for 2000 epochs under the same optimization settings. The KAN model was implemented using the CPU-based PyKAN library, available at GitHub [2], with the following configuration: L-BFGS optimizer, 100 optimization steps, two hidden layers with 40 units each (`[40, 40]`), grid size of 5, and polynomial order (`korder`) of 3. These hyperparameters were selected based on preliminary experiments and provided the best performance during early research.

For both the Hybrid FCN-KAN and FCKAN architectures, the same hyperparameters as their base counterparts were maintained, except for the number of training epochs. The Hybrid FCN-KAN was trained for 100 epochs for efficiency, while the FCKAN model was trained for up to 2000 epochs with early stopping based on training loss since the benchmark datasets do not have validation split. Since only train-test splits were available (i.e., no separate validation set), early stopping was implemented by carefully monitoring the training loss. Training was automatically stopped if the loss did not improve by at least 10% over the previous best value within the last 50 epochs. This strategy primarily aimed to mitigate the increased computational cost and slower convergence of the FCKAN architecture while slightly reducing overfitting.

5. Results and Discussion

Although both tasks addressed in this work, TSC and TSER, share the same data structure, they differ substantially in their objectives, evaluation metrics, and modeling challenges. To facilitate a more focused and pedagogically sound discussion, we structure this section into two separate parts. The first covers TSC, while the second focuses on TSER.

5.1. Time Series Classification (TSC)

Table 1 presents the overall performance of each model averaged across all 128 datasets. For each architecture, we report the mean and standard deviation of four key metrics: Accuracy, Weighted F1-score, Recall, and Precision. This global summary offers a concise and comprehensive view of each model's general behavior across the full benchmark, avoiding the impracticality of visualizing metrics for every individual dataset.

Table 1. Mean ± standard deviation of performance metrics across all 128 UCR datasets. Rows are ordered by descending accuracy (higher is better). Boldface highlights the best-performing model for each metric.

Model	Accuracy	Weighted F1-score	Recall	Precision
Hybrid FCN-KAN	$\mathbf{0.802 \pm 0.183}$	$\mathbf{0.800 \pm 0.185}$	$\mathbf{0.802 \pm 0.183}$	$\mathbf{0.810 \pm 0.177}$
FCKAN	0.774 ± 0.190	0.764 ± 0.208	0.774 ± 0.190	0.781 ± 0.202
FCN	0.772 ± 0.219	0.760 ± 0.235	0.772 ± 0.219	0.770 ± 0.230
KAN	0.666 ± 0.259	0.663 ± 0.263	0.666 ± 0.259	0.656 ± 0.664
MLP	0.628 ± 0.232	0.594 ± 0.258	0.628 ± 0.232	0.602 ± 0.267

To statistically compare model performance across the benchmark, we applied the Nemenyi post-hoc test and summarized results using a Critical Difference (CD) diagram

[2] `https://github.com/KindXiaoming/pykan`

(Figure 3), created with Aeon Toolkit [Middlehurst et al. 2024]. The diagram ranks models based on average performance and identifies groups with no significant difference. Results indicate that the non-temporal models, MLP and KAN, perform statistically worse than temporal models across all datasets, although KAN outperforms MLP on average. Among the temporal models, Hybrid FCN-KAN achieved the best overall average rank but is statistically tied with both FCN and FCKAN in performance.

Pairwise Win/Tie/Loss counts across all datasets further clarify comparative results (Table 2). The Hybrid FCN-KAN model dominates most pairwise comparisons, particularly against MLP, with 99 wins and only 24 losses recorded. Focusing on non-temporal models, KAN consistently outperforms MLP, confirming the advantages of its architecture despite lacking explicit temporal modeling. Among the proposed hybrid models, Hybrid FCN-KAN and FCKAN perform comparably, with Hybrid FCN-KAN showing a slight edge in several comparisons. This difference may arise from the challenges of end-to-end training in FCKAN, where gradient propagation from the KAN component back to the convolutional feature extractor can be less effective compared to the pre-trained, frozen feature extractor approach used in Hybrid FCN-KAN. Overall, these results highlight that integrating KAN modules into FCN architectures significantly improves classification performance, while end-to-end joint training requires careful consideration to avoid potential optimization difficulties.

Table 2. Pairwise Win/Tie/Loss counts between all model pairs across 128 TSC datasets. Each cell reports results for the model in the row compared against the model in the column.

	MLP	KAN	FCN	Hybrid FCN-KAN	FCKAN
MLP	-	49/2/77	28/2/98	24/5/99	38/1/89
KAN	-	-	46/4/78	42/4/82	51/2/75
FCN	-	-	-	56/10/62	70/4/54
Hybrid FCN-KAN	-	-	-	-	70/4/54
FCKAN	-	-	-	-	-

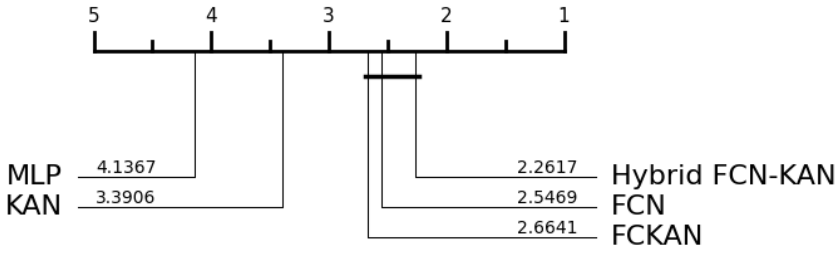

Figure 3. Critical Difference for TSC. Horizontal black bar indicates groups of models with no statistically significant difference (closer to 1, the better).

5.2. Time Series Extrinsic Regression

For the TSER task, we evaluated model performance across all 19 univariate and multivariate datasets from the Monash, UEA & UCR TSER Repository. Table 3 presents the

mean and standard deviation of our metrics through the datasets: RMSE, MSE, MAE and R2. Meanwhile, Table 4 shows the RMSE for each model on these datasets.

Table 3. Mean ± standard deviation of performance metrics across all 19 datasets. Rows are ordered by increasing RMSE (lower is better). Boldface highlights the best-performing model for each metric.

Model	RMSE	MSE	MAE	R^2
FCN	$\mathbf{(1.9 \pm 2.6) \times 10^1}$	$\mathbf{(1.0 \pm 2.1) \times 10^3}$	$\mathbf{(1.3 \pm 1.7) \times 10^1}$	-5.4 ± 23.3
Hybrid FCN-KAN	$(2.6 \pm 4.0) \times 10^1$	$(2.2 \pm 5.7) \times 10^3$	$(1.4 \pm 1.8) \times 10^1$	0.1 ± 0.8
FCKAN	$(2.7 \pm 4.2) \times 10^1$	$(2.4 \pm 5.3) \times 10^3$	$(1.9 \pm 2.9) \times 10^1$	$\mathbf{0.2 \pm 0.3}$
MLP	$(4.0 \pm 7.4) \times 10^1$	$(6.8 \pm 21.9) \times 10^3$	$(2.9 \pm 5.6) \times 10^1$	-1.2 ± 4.6
KAN	$(5.7 \pm 13.8) \times 10^1$	$(2.1 \pm 8.4) \times 10^4$	$(4.2 \pm 10.3) \times 10^1$	$(-1.6 \pm 6.9) \times 10^4$

Table 4. RMSE regression results across all 19 datasets. Boldface highlights the best-performing model for each dataset.

Dataset	MLP	KAN	FCN	Hybrid FCN-KAN	FCKAN
AppliancesEnergy	3.31	4.57	3.27	3.16	**2.87**
AustraliaRainfall	8.31	**6.88**	9.00	16.43	8.49
BIDMC32HR	21.35	12.83	11.32	11.27	**7.66**
BIDMC32RR	4.87	**2.76**	6.20	4.90	3.35
BIDMC32SpO2	21.41	**2.46**	5.77	5.82	5.07
BeijingPM10Quality	119.04	121.76	**93.83**	93.75	114.43
BeijingPM25Quality	93.57	102.99	61.72	**60.18**	88.46
BenzeneConcentration	**3.94**	6.47	4.86	5.12	6.95
Covid3Month	0.05	24.06	0.07	0.05	**0.04**
FloodModeling1	0.02	0.03	**0.01**	**0.01**	0.02
FloodModeling2	0.02	0.01	**0.01**	**0.01**	0.02
FloodModeling3	0.03	0.02	**0.01**	**0.01**	0.02
HouseholdPowerConsumption1	309.98	605.16	**45.73**	154.64	139.93
HouseholdPowerConsumption2	68.80	47.01	**34.24**	45.65	40.75
IEEEPPG	49.29	76.94	**34.01**	35.27	38.64
LiveFuelMoistureContent	39.76	55.23	**33.65**	44.22	35.32
NewsHeadlineSentiment	**0.14**	**0.14**	0.35	**0.14**	**0.14**
NewsTitleSentiment	**0.14**	**0.14**	1.40	**0.14**	**0.14**
PPGDalia	18.67	23.02	14.10	18.43	**13.71**

To assess the statistical significance of performance differences across the TSER datasets, we performed a Nemenyi post-hoc test. This test for TSER (Figure 4) indicates that FCN achieved the best average rank (2.5263) among the evaluated models. It forms a statistically indistinguishable group with the all other models (MLP, KAN, FCKAN, Hybrid FCN-KAN), but it is clear that MLP and KAN occupy worse average ranks.

To further compare performance, we computed pairwise Win/Tie/Loss counts for each model across the 19 TSER datasets, as presented in Table 5. Each entry quantifies how many datasets a model won, tied, or lost against another. An examination of these counts reveals distinct behavioral patterns. Models without explicit temporal recognition, namely MLP and KAN, demonstrate a relatively balanced win/loss dynamic when compared against each other. However, a clear disadvantage emerges for these non-temporal

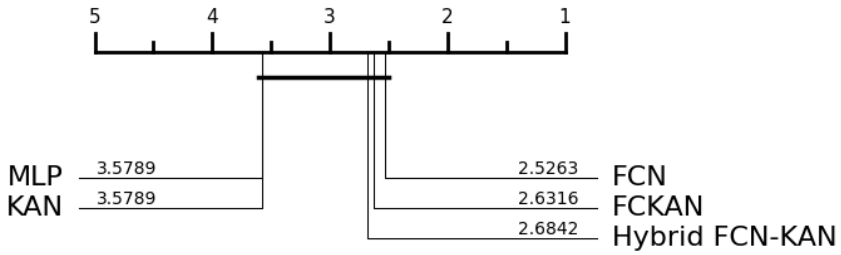

Figure 4. Critical Difference for TSER. Horizontal black bar indicates groups of models with no statistically significant difference (closer to 1, the better).

models when they are pitted against architectures designed specifically with temporal feature extraction capabilities. In such comparisons, MLP and KAN consistently incur a higher number of losses.

Conversely, models incorporating temporal recognition (FCN, FCKAN, Hybrid FCN-KAN) exhibit a similar competitive balance among themselves. More significantly, these temporal models achieve a substantial majority of wins when compared directly to their non-temporal counterparts. This difference in performance between models with and without temporal feature handling decisively testifies to the importance of effectively capturing temporal characteristics for achieving better results in time series analysis tasks.

Table 5. Pairwise Win/Tie/Loss counts between all model pairs across 19 TSER datasets. Each cell reports results for the model in the row compared against the model in the column.

	MLP	**KAN**	**FCN**	**Hybrid FCN-KAN**	**FCKAN**
MLP	-	10/0/9	6/0/13	7/0/12	4/0/15
KAN	-	-	5/0/14	5/0/14	8/0/11
FCN	-	-	-	10/0/9	10/0/9
Hybrid FCN-KAN	-	-	-	-	9/0/10
FCKAN	-	-	-	-	-

6. Concluding Remarks and Future Work

This paper introduced FCKAN, a novel architecture combining FCNs with KANs for TSC and TSER tasks. Motivated by FCNs' strength in temporal feature extraction and KANs' high interpretability via learnable spline functions, we proposed two integration strategies: a hybrid two-stage model using a pre-trained FCN as a robust feature extractor followed by a KAN, and an end-to-end FCKAN trained jointly.

Experiments on the UCR TSC Archive and the Monash, UEA & UCR TSER Repository revealed that for TSC, Hybrid FCN-KAN outperformed standalone FCNs and other baselines, suggesting that pre-training FCNs for temporal features before applying KAN yields significant benefits. The end-to-end FCKAN was statistically comparable but did not consistently exceed the hybrid model, likely due to joint training complexities

and optimization challenges. Both novel models outperformed non-temporal baselines, highlighting the importance of temporal inductive biases.

For TSER, temporal feature extraction remained crucial. Although FCN achieved the best average rank, statistical tests found no significant difference among models. Nonetheless, temporal models (FCN, Hybrid FCN-KAN, FCKAN) consistently outperformed non-temporal baselines, confirming the value of capturing temporal dynamics. While KAN integrations did not surpass FCN statistically in TSER, their competitive performance indicates promise, especially with further regression fine-tuning.

Future work will focus on investigating the gradient flow issues observed during end-to-end FCKAN training, aiming to identify and resolve problems in joint optimization. Additionally, we plan to integrate KAN with other temporal encoders, such as Transformers and recurrent networks, to improve performance and extend applicability.

7. Acknowledgments

The authors gratefully acknowledge the efforts of all contributors to the UCR TSC Archive and the UEA & UCR TSER Repository, whose dedication has significantly advanced time series research. This research was supported by the São Paulo Research Foundation (FAPESP), grant numbers $2024/07047-7$ and $2024/14856-9$.

References

Ang, L.-M. and Seng, K. P. (2016). Big sensor data applications in urban environments. *Big Data Research*, 4:1–12.

Bagnall, A., Bostrom, A., Large, J., and Lines, J. (2016). The great time series classification bake off: An experimental evaluation of recently proposed algorithms. extended version.

Dau, H. A., Keogh, E., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., Yanping, Hu, B., Begum, N., Bagnall, A., Mueen, A., and Batista, G. (2018). The ucr time series classification archive. `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`.

Dong, C., Zheng, L., and Chen, W. (2024). Kolmogorov-arnold networks (kan) for time series classification and robust analysis.

El Maachi, I., Bilodeau, G.-A., and Bouachir, W. (2020). Deep 1d-convnet for accurate parkinson disease detection and severity prediction from gait. *Expert Systems with Applications*, 143:113075.

Ismail-Fawaz, A., Devanne, M., Weber, J., and Forestier, G. (2022). Deep learning for time series classification using new hand-crafted convolution filters. pages 972–981.

Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., and Muller, P.-A. (2019). Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963.

Ismail Fawaz, H., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D. F., Weber, J., Webb, G. I., Idoumghar, L., Muller, P.-A., and Petitjean, F. (2020). Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962.

Lim, B. and Zohren, S. (2021). Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194):20200209.

Liu, Z., Wang, Y., Vaidya, S., Ruehle, F., Halverson, J., Soljačić, M., Hou, T. Y., and Tegmark, M. (2024). Kan: Kolmogorov-arnold networks.

Middlehurst, M., Ismail-Fawaz, A., Guillaume, A., Holder, C., Guijo-Rubio, D., Bulatova, G., Tsaprounis, L., Mentel, L., Walter, M., Schäfer, P., and Bagnall, A. (2024). aeon: a python toolkit for learning from time series. *Journal of Machine Learning Research*, 25(289):1–10.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Tan, C. W., Bergmeir, C., Petitjean, F., and Webb, G. I. (2021a). Time series extrinsic regression. *Data Mining and Knowledge Discovery*, pages 1–29.

Tan, C. W., Bergmeir, C., Petitjean, F., and Webb, G. I. (2021b). Time series extrinsic regression. *Data Mining and Knowledge Discovery*, 35(3):1032–1060.

Wang, Z., Yan, W., and Oates, T. (2016). Time series classification from scratch with deep neural networks: A strong baseline.