End-to-End Audio Deepfake Detection Using Kolmogorov-Arnold Network

Anonymous submission to Interspeech 2025

Abstract

Majority of existing studies in speech processing employed Convolution Neural Networks (CNNs), until the discovery of Kolmogorov Arnold Networks (KANs). Both have one or another limitation, however in this study, we propose a new model, which has benefits of both CNNs and KANs. We employed End-to-End (E2E) speech-based model, which takes raw audio as input, and processes without extracting its acoustic features, which makes model more cost efficient and time efficient. For this particular study, we employed proposed model for Audio Deepfake Detection (ADD) task. When observed and compared the model with existing CNN model, our proposed model outperform existing CNN method by 81.56 %, and is 3.54 % more time-efficient than existing models. We provide detailed overview of the proposed model and also the relevant analysis of the proposed system. In order to evaluate the model efficiently, we also performed cross-dataset evaluation.

Index Terms: Fully Learnable Networks, KAN, CNN, MLP, ADD.

1. Introduction

Convolutional Neural Networks (CNNs) have undeniably transformed the landscape of deep learning, achieving remarkable success in areas, such as audio processing [1], image recognition [2], computer vision [3], and natural language processing [4]. Their ability to automatically learn hierarchical feature representations from raw data has made them the go-to architecture for a wide variety of applications. However, despite these achievements, CNNs are not without limitations that hinder their scalability and real-world performance. One significant challenge is the high computational cost associated with training deep CNN models, which require large amounts of labeled data and powerful hardware [3]. This makes their deployment in resource-constrained environments, such as mobile devices or real-time systems, a complex and costly task. Furthermore, CNNs often struggle to generalize well across different domains due to fixed hyperparameters, such as batch-size, and fixed activation function, leading to issues such as overfitting, where models perform well on training data but fail to adapt to unseen or diverse datasets [5]. The vulnerability to adversarial attacks—where small, imperceptible changes to input data can cause drastic misclassifications—poses a critical concern, particularly in safety-critical applications such as autonomous driving and healthcare. Additionally, CNNs are limited in their capacity to capture contextual or temporal information, making them less effective in tasks requiring long-term dependencies or nuanced understanding. These limitations highlight the need for more robust, efficient, and flexible architectures that can address the challenges faced by existing CNN models and extend their applicability across a broader range of tasks and environments.

Using CNNs for speech processing and audio security offers notable advantages, such as automatic feature extraction from raw audio, eliminating the need for manual (handcrafted) feature engineering. This enables improved accuracy in tasks such as speech recognition and speaker recognition. For this particular study we propose a solution to Audio Deepfake Detection (ADD) problem. Additionally, CNNs are relatively robust to noise, making them effective in environments, where background interference is present. However, they also come with disadvantages, such as high computational demands, which can limit their use in resource-constrained settings. Furthermore, CNNs struggle with capturing long-term temporal dependencies in audio signals, which are crucial for tasks such as emotion detection or context-aware speech processing. Existing works [6] and [7] have demonstrated the potential of CNNs in speech recognition and speaker identification, highlighting their ability to leverage raw audio data and time-frequency features for improved performance.

1.1. Related Works

Many existing works have explored CNN based networks for various speech processing tasks, such as (ADD). In [8], authors employed Mel Frequency Cepstral Coefficients (MFCC) based features in combination with joint CNN-LSTM classifiers and obtained 94.37 % accuracy. In [9], authors employed CNN with BiLSTM classifier and proposed an optimal model, however, authors lack the basic key backend understanding of CNNs. Most of the existing studies have employed 2-stage classification methods (acoustic feature extraction in first stage, and pattern classifier in second stage), however, this process is much complicated and time consuming. Also in order to obtain optimal features, we need to explore variety of features. As an alternative, this study propose an end-to-end (E2E) method for ADD. The main motivation of this study is provide new direction to existing approaches rather than to provide an optimal solution to a particular problem. Existing methods mostly used CNN-based classifiers, however, after recent discovery of Kolmogorov-Arnold-Networks (KANs) [10], the ability of Multilayer Perceptron (MLPs) and Universal Approximation Theorem (UAT) have been questioned majorly. This study proposes a method that modifies existing CNN methods, such that it also has benefits of KANs and thereby propose a new model architecture. Few early studies [11, 12, 13, 14, 15] have tried implementing KAN with MLPs, however, it failed to demonstrate mathematical and experimental analysis for speech problems. In [12], authors tried to embed KAN with audio processing, however, their work is restricted to wavelets (short term

waves) only, alternatively our proposed methodology provides solution irrespective of length of audios. Proposed study provides the following novelty:

- Kolmogorov-Arnold Network (KAN) for ADD.
- Cost and time efficient processing.
- E2E speech classification.
- CNN along with benefits of KANs.

2. Proposed Methodology

Upto the date, the structure of CNN consist of several convolution blocks, which are formed via combination of convolution layer, pooling layers, and an activation function. In proposed method, we replaced the lower dense layers of CNN [16] with KANLinear layers [17]. The key difference that occurs is, along with learnable weights, now the fixed activation function of CNN also becomes learnable. In this way, both weights and activation becomes learnable, resulting into a step closer to Fully Learnable Networks (FLNs).

2.1. KANLinear Layer

Kolmogorov-Arnold Representation Theorem (KART) [10, 18]: It states that if f is a multivariate continuous 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[10]. More specifically, for a smooth $f : [0, 1] \rightarrow \mathbb{R}$,

$$f(x) = \sum_{q=1}^{2k+1} \Phi_q \left(\sum_{p=1}^{n} \Phi_{q,p}(x_p) \right), \tag{1}$$

where $\Phi_{q,p} : [0, 1] \rightarrow \mathbb{R}, and \quad \Phi_q : \mathbb{R} \rightarrow \mathbb{R}$. Here, the multivariate function is addition and every other function is written using univariate functions and sum

2.1.1. Base Weight

The base weight is a trainable parameter of shape (out_features, in_features), responsible for learning linear relationships between inputs and outputs. It is initialized using Xavier uniform initialization to ensure balanced weight distribution, preventing vanishing or exploding gradients. This module acts similarly to a standard linear layer in neural networks but is enhanced by an additional spline-based component. Using a learnable weight matrix instead of fixed transformations allows the model to adaptively capture patterns in the data, making it more effective than static feature transformations. The transformation is performed as [10]:

$$Z_{\text{base}} = XW_{\text{base}}^{T} \quad , \tag{2}$$

where
$W_{\text{base}} \in \mathbb{R}^{\text{out\_features} \times \text{in\_features}}$ is the learnable weight matrix, initialized near identity.
$X \in \mathbb{R}^{\text{batch} \times \text{in\_features}}$ is the input feature vector.
$Z_{\text{base}} \in \mathbb{R}^{\text{batch} \times \text{out\_features}}$ is the output before applying activation
batch = Number of input audio signals.

2.1.2. Base Activation

The base activation function introduces non-linearity before applying the base weight transformation. By default, it is nn.PReLU(), a learnable activation that adapts its slope during training. This is more flexible than fixed activations like ReLU or Tanh, which may not always fit the data distribution well. Using an adaptive activation ensures better learning dynamics, helping the model capture subtle variations in the input data. This component complements the base weight, allowing it to model both simple and complex patterns effectively [19].

$$\text{PReLU}(x_i) = max(0, x_i) + a_i min(0, x_i). \tag{3}$$

Instead of directly applying the transformed values, a learnable nonlinear activation function is introduced:

$$W_{\text{base}}^{\text{act}} = \text{PReLU}(W_{\text{base}}), \tag{4}$$

where each weight element in W_{base} is individually activated using PReLU before being multiplied with the input. Thus, the final base transformation is:

$$Z_{\text{base}} = X{W_{\text{base}}^{\text{act}}}^{T}, \tag{5}$$

Resulting shape: (batch, out_features)

2.1.3. Spline Weight

The spline weight is a trainable parameter of shape (out_features, in_features, grid_size + spline_order), which enables non-linear feature transformations. Unlike traditional activation functions like ReLU, which apply fixed non-linear mappings, spline weights allow the model to learn data-driven, smooth, and continuous transformations. The weight matrix is initialized with small noise and converted into spline coefficients using least squares, ensuring stability and proper function approximation. This approach is superior to manually selecting non-linear functions, as it adapts to the input distribution dynamically.

2.1.4. Grid

The grid is a non-trainable buffer defining interpolation points for the B-spline transformation. It is computed based on the grid_size, spline_order, and a predefined range, ensuring structured and smooth non-linearity. Unlike conventional neural networks, which rely on predefined non-linearities, the grid allows KANLinear to use smooth basis functions for interpolation, leading to better generalization and interpretability. This structured approach prevents discontinuities in transformations, making it a preferred choice over randomly initialized non-linearity functions.
The B-spline basis functions are computed based on a **grid** that defines knot points for the splines. This grid is stored in:

`self.grid` (Shape: (in_features, total number of knots)),

where total number of knots in the grid equals to:

$$(grid\_size + (2 \times spline\_order) + 1) \tag{6}$$

This computation is recursive and refines the indicator functions into smooth basis functions.

2.1.5. B-Spline Interpolation

These modules compute B-spline basis functions (b_splines) and convert function values into spline coefficients using least squares (curve2coeff). The B-spline interpolation ensures smooth, continuous transformations, avoiding abrupt changes in activation values. Least squares regression helps determine optimal spline coefficients, ensuring that the learned function closely matches the input-output relationships. This approach is

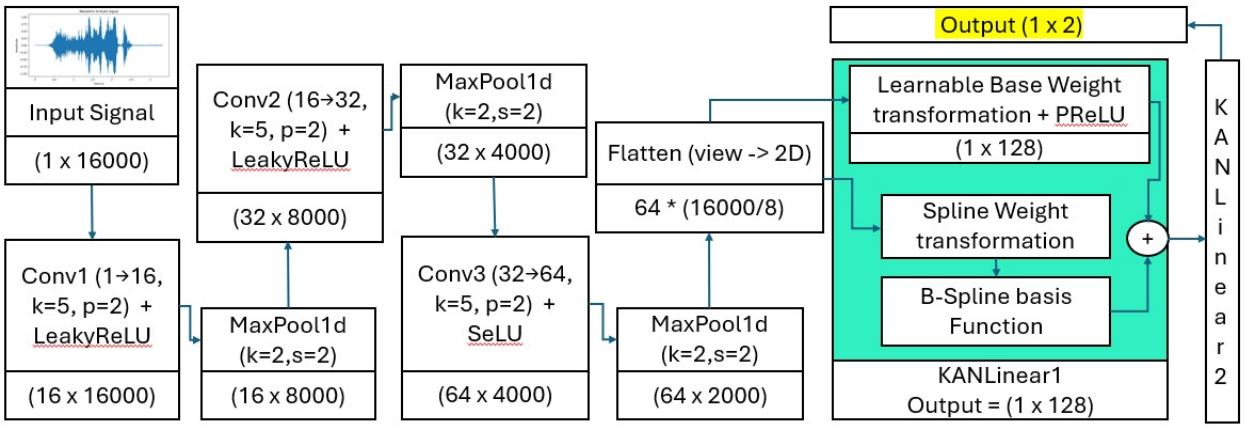

Figure 1: *Functional block diagram of proposed methodology for end-to-end ADD using KAN.*

more flexible than manually defined polynomial expansions or non-adaptive feature transformations, making it ideal for structured yet learnable function approximation. For each input feature (in_features features), a set of B-spline basis functions is computed using [20]:

$$B_{i,k}(x) = \frac{x - t_i}{t_{i+k} - t_i} B_{i,k-1}(x) + \frac{t_{i+k+1} - x}{t_{i+k+1} - t_{i+1}} B_{i+1,k-1}(x), \tag{7}$$

where
$B_{i,k}(x)$ is the basis function of order k.
t_i are the knot positions in the grid.
The number of B-spline basis functions equals to

$$\text{grid\_size} + \text{spline\_order}$$

In this experiment, the grid size is set to 5 and spline order is set to 3. For each sample in the batch, we perform a weighted sum of the B-spline basis function activations. This is computed as [10]:

$$\text{Spline Output} = \sum_{i=1}^{in\_features} \sum_{j=1}^{bf} B_{i,j} \cdot W_{o,i,j} \tag{8}$$

where bf equals to total number of B-spline basis functions $B_{i,j}$ represents the B-spline basis function activations (shape: $(\text{batch}, \text{in\_features}, bf)$). $W_{o,i,j}$ represents the learnable spline weight coefficients (shape: $(\text{out\_features}, \text{in\_features}, bf)$).
This operation is equivalent to performing a linear transformation where [20]:

$$\text{spline\_output} = \text{B-spline basis} \times \text{spline\_weight} \tag{9}$$

Resulting shape of spline output: $(\text{batch}, \text{out\_features})$

The architecture mentioned replaces traditional FC layer with a more expressive transformation KANLinear which learns nonlinear relationships in features via PReLU and enhances classification performance by making weight transformations adaptive and learnable.

2.1.6. KANLinear Output

The KANLinear output is computed as:

$$\text{output} = (\text{base weight} \times \text{activation}) + (\text{spline weight} \times \text{B-spline basis}),$$

where **Base Weight**: Standard linear transformation,
Spline Weight: A learnable function over grid points which approximates smooth nonlinear mappings,
Resulting output shape: (batch,out_features).

2.2. Motivation

One of the key problem in existing CNN is it learns weights however, the dense layers at end of CNN remains fixed. Replacing dense layers in CNNs [21] with KANLinear offers several advantages in terms of interpretability, efficiency, and robustness. Traditional dense layers function as black boxes, making it challenging to understand how models process information. In contrast, KANLinear provides an interpretable mapping, enabling researchers to analyze feature transformations explicitly. Additionally, dense layers introduce a large number of parameters, leading to high memory consumption, increased computational complexity, and potential overfitting. KANLinear mitigates these issues by leveraging a structured, low rank functional decomposition, reducing the number of learnable parameters while maintaining expressivity [22]. Another key advantage of KANLinear is its improved generalization ability. Dense layers often memorize patterns rather than learning the underlying function effectively, whereas KANLinear enforces structured representations that capture smooth functional mappings, leading to better performance on unseen data [23]. Moreover, KANLinear enhances robustness against adversarial perturbations. Traditional dense layers are highly sensitive to adversarial attacks and noise, however KANLinear's function-based decomposition provides a smoother, more robust mapping, making models less vulnerable to such perturbations [24]. Furthermore, the approach aligns with neuroscientific principles, as biological neural networks do not utilize fully connected layers in the same manner as artificial networks. Inspired by the Kolmogorov-Arnold representation theorem, KANLinear reflects the distributed and structured processing observed in human cognition. Lastly, KANLinear enables efficient learning with fewer data. Dense layers require a large amount of labeled data for effective generalization, whereas KANLinear structures the learning process to allow better feature learning with limited data. These advantages make KANLinear a compelling alternative to traditional dense layers in CNNs, improving both interpretability and model efficiency.

Despite of this benefits, KAN also have limitations, such as higher *time* and *space* complexity. KAN takes comparatively more time to train as compared to CNNs as it also learns activation function along with optimizing weights. Additionally CNNs need normalization functions after convolution blocks to prohibit overfitting when tested, however, when KANs are employed with CNN they do not require any normalization function to prohibit overfitting.

3. Experimental Setup

3.1. Dataset Details

In this work, Fake-or-Real (FoR) [25] and In-The-Wild [26] datasets are used. The dataset selection task was motivated by the shortage of deepfake dataset available openly, in which FoR being largest and freely available [25]. Refereed from previous works [27], the authors of that study have also used ASVspoof, which consist of 1,25,000 samples, compared to which FoR consist of 1,95,541 utterances of total [25]. This is the key reason to use FoR dataset instead of ASVSpoof dataset. For both the dataset used in the process of evaluation, the files were taken as mentioned in Table 1. Multiple models were tested on both the dataset for proper testing of the model.

3.1.1. FoR

The dataset contains around 87,000 fake utterances as well as 1,11,000 real utterances generated by more than 1200 speakers. The dataset has been further divided into 4 sub-datasets as described breif below. FoR-original is the version, which has original collected utterances. FoR-norm is the version of dataset that has samples which were collected after performing certain set of preprossessing on the original audio. FoR-2second contains the samples truncated to 2 second and at last forrerecorded consist of all utterance which were recorded again from the original audio with gadgets like speaker and microphone. We employed FoR-2Seconds dataset for the experiments in this study as it contains 2 seconds audio files. This particular smaller dataset was employed due to limited resources and time.

3.1.2. In-The-Wild

The test portion of In-The-Wild dataset has been used in this study in order to evaluate the model on realistic unseen data i.e., cross-database scenarios. The dataset consists of 37.9 hours of audio, which is mixture of 20.7 hours of original audio, and 17.2 hours of artificially generated audio [26]. There are about 23 minutes of bonafide and 18 minutes of fake audio per individual speaker. English language was preferred while creation of In-The-Wild dataset. The average time of audio samples is 4.3 seconds, and audios are of 16 kHz sampling rate.

Table 1: *Dataset Details [25], and [26]*

	Training		Testing		Validation	
	Fake	Real	Fake	Real	Fake	Real
FoR 2Seconds	6978	6978	544	544	1413	1413
In-The-Wild	-		11816	19963	-	

4. Experimental Results

We conducted experiments with CNN model and compared results with proposed combine model of CNN & KAN. Proposed model takes quiet more amount of time, and resources, it also gives better results as compared to CNN model. We in this study did not obtain much more accuracy as compared to existing works, however, we in this study broke the ice between classical CNN models and newly adopted KANs. For comparative results, we trained CNN and replaced last fully connected layer with KANLinear layers. Fig. 2 represents the variation in accuracy *w.r.t.* variation in number of epoch. It can be observed that even in the least epoch scenario, proposed method outperforms classical CNN model, representing significance of need to built FLNs.

Further, we developed an E2E mechanism, which takes raw audio as input and do not need to find any optimal features for classification. However, we further aim to optimize the features and then feed feature to proposed classifier to optimize accuracy of system. We also evaluated performance on cross dataset scenario, where we obtained 51.35 % accuracy with CNN classifier, and 54.47 % accuracy with proposed KAN and CNN classifier.

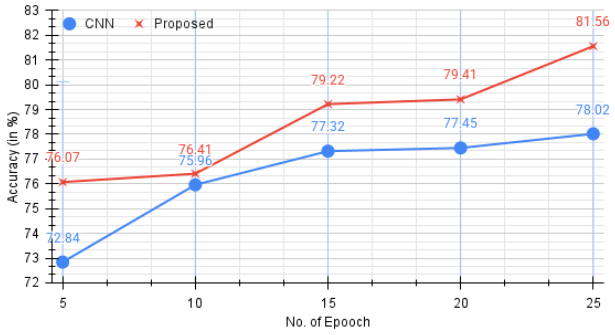

Figure 2: *Representation of epochwise training on FoR-2Second dataset.*

Table 2 represents the confusion matrix of classical CNN, and proposed CNN + KAN model. It can be observed that when tested on CNN, more real samples are predicted as fake, which happens due to unavailability of model to capture key characteristics of real audio from original speech signal.

Table 2: *Confusion Matrix of 25 Epoch Experiments*

	CNN	**Real**	**Fake**	**KAN+CNN**	**Real**	**Fake**
Actual	**Real**	1888	792	**Real**	2094	586
	Fake	377	2305	**Fake**	403	2279
	Predicted					

5. Summary and Conclusions

In this study, we aim to break the ice between classical CNNs, and recently proposed KANs. We in this study propose a model, which is combination of both classical CNN and KANs. The proposed network paths a new way, which is close to form a fully learnable network. We analyzed and compared the behavior of CNN with proposed method, which resulted in better accuracy of model. We proposed a E2E method which saves time for finding optimal acoustic feature, as it takes raw audio as an input. Further, results can be increased by fine-turning model using optimal selective features. Future works include optimizing the proposed model by learning the activation functions in each convolution blocks. Deeper Understanding of theoretical results of KAN with respect to ADD task, and machine learning, in general, remains an open research problem.

6. References

[1] Mustaqeem and S. Kwon, "A cnn-assisted enhanced audio signal processing for speech emotion recognition," *Sensors*, vol. 20, no. 1, p. 183, 2019.

[2] T. Guo, J. Dong, H. Li, and Y. Gao, "Simple convolutional neural network on image classification," in *IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, 2017, Beijing, China, pp. 721–724.

[3] D. Bhatt, C. Patel, H. Talsania, J. Patel, R. Vaghela, S. Pandya, K. Modi, and H. Ghayvat, "Cnn variants for computer vision: History, architecture, application, challenges and future scope," *Electronics*, vol. 10, no. 20, p. 2470, 2021.

[4] W. Wang and J. Gang, "Application of convolutional neural network in natural language processing," in *IEEE International Conference on Information Systems and Computer Aided Education (ICISCAE)*, 2018, Changchun, China, pp. 64–70.

[5] Á. Zarándy, C. Rekeczky, P. Szolgay, and L. O. Chua, "Overview of cnn research: 25 years history and the current trends," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015, Lisbon, Portugal, pp. 401–404.

[6] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold *et al.*, "Cnn architectures for large-scale audio classification," in *IEEE international conference on acoustics, speech and signal processing (icassp)*, 2017, New Orleans, Louisiana, pp. 131–135.

[7] C. Zhang, K. Koishida, and J. H. Hansen, "Text-independent speaker verification based on triplet convolutional neural network embeddings," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 9, pp. 1633–1644, 2018.

[8] M. Chitale, A. Dhawale, M. Dubey, and S. Ghane, "A hybrid cnn-lstm approach for deepfake audio detection," in *2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT)*. IEEE, 2024, pp. 1–6.

[9] T. M. Wani, S. A. A. Qadri, D. Comminiello, and I. Amerini, "Detecting audio deepfakes: Integrating cnn and bilstm with multi-feature concatenation," in *Proceedings of the 2024 ACM Workshop on Information Hiding and Multimedia Security*, 2024, pp. 271–276.

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

[11] Y. Cang, L. Shi *et al.*, "Can kan work? exploring the potential of kolmogorov-arnold networks in computer vision," *arXiv preprint arXiv:2411.06727*, 2024, {Last Accessed Date: $2^{nd} February, 2025$}.

[12] Z. Bozorgasl and H. Chen, "Wav-kan: Wavelet kolmogorov-arnold networks," *arXiv preprint arXiv:2405.12832*, 2024, {Last Accessed Date: $2^{nd} February, 2025$}.

[13] C. J. Vaca-Rubio, L. Blanco, R. Pereira, and M. Caus, "Kolmogorov-arnold networks (kans) for time series analysis," *arXiv preprint arXiv:2405.08790*, 2024, {Last Accessed Date: $2^{nd} February, 2025$}.

[14] B. Azam and N. Akhtar, "Suitability of kans for computer vision: A preliminary investigation," *arXiv preprint arXiv:2406.09087*, 2024, {Last Accessed Date: $2^{nd} February, 2025$}.

[15] J. Schmidt-Hieber, "The kolmogorov–arnold representation theorem revisited," *Neural networks*, vol. 137, pp. 119–126, 2021.

[16] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[17] Y. Cao, Z. Yuan, H. Feng, T. Wang, B. Zhang, and C. Li, "Kanlinear: A lightweight model for multi-step feedforward prediction," in *China Automation Congress (CAC)*, 2024, Qingdao, China, pp. 3335–3340.

[18] V. I. Arnold, "On the representation of functions of several variables as a superposition of functions of a smaller number of variables," *Collected works: Representations of functions, celestial mechanics and KAM theory, 1957–1965*, pp. 25–46, 2009.

[19] T. Jiang and J. Cheng, "Target recognition based on cnn with leakyrelu and prelu activation functions," in *International Conference on Sensing, Diagnostics, Prognostics, and Control (SDPC)*, 2019, Beijing, China, pp. 718–722.

[20] A. Chaudhuri, "B-splines," *arXiv preprint arXiv:2108.06617*, 2021, {Last Accessed Date: $2^{nd} February, 2025$}.

[21] M. Meftahul Ferdaus, M. Abdelguerfi, E. Ioup, D. Dobson, K. N. Niles, K. Pathak, and S. Sloan, "Kanice: Kolmogorov-arnold networks with interactive convolutional elements," *arXiv e-prints*, pp. arXiv–2410, 2024, {Last Accessed Date: $2^{nd} February, 2025$}.

[22] S. SS, K. AR, A. KP *et al.*, "Chebyshev polynomial-based kolmogorov-arnold networks: An efficient architecture for nonlinear function approximation," *arXiv preprint arXiv:2405.07200*, 2024, {Last Accessed Date: $2^{nd} February, 2025$}.

[23] J. Braun and M. Griebel, "On a constructive proof of kolmogorov's superposition theorem," *Constructive approximation*, vol. 30, pp. 653–675, 2009.

[24] R. Zimbres, "https://medium.com/%40rubenszimbres/kolmogorov-arnold-networks-a-critique-2b37fea2112e," Medium, Tech. Rep., 2024.

[25] R. Reimao and V. Tzerpos, "For: A dataset for synthetic speech detection," in *2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, 2019, Timișoara, Romania, pp. 1–10.

[26] N. M. Müller, P. Czempin, F. Dieckmann, A. Froghyar, and K. Böttinger, "Does audio deepfake detection generalize?" *arXiv preprint arXiv:2203.16263*, 2022, {Last Accessed : 22 January, 2024}.

[27] P. Kawa, M. Plata, M. Czuba, P. Szymański, and P. Syga, "Improved deepfake detection using whisper features," *arXiv preprint arXiv:2306.01428*, 2023, {Last Accessed : 22 January, 2024}.