"KAN you hear me?" Exploring Kolmogorov-Arnold Networks for Spoken Language Understanding

Interspeech 2025
17-21 August 2025, Rotterdam, The Netherlands

"KAN you hear me?" Exploring Kolmogorov-Arnold Networks for Spoken Language Understanding

Alkis Koudounas[†], *Moreno La Quatra*[‡], *Eliana Pastor*[†], *Sabato Marco Siniscalchi*[⋆], *Elena Baralis*[†]

[†]Politecnico di Torino, Turin, Italy
[‡]Kore University of Enna, Enna, Italy
[⋆]Università degli Studi di Palermo, Palermo, Italy

alkis.koudounas@polito.it, moreno.laquatra@unikore.it

Abstract

Kolmogorov-Arnold Networks (KANs) have recently emerged as a promising alternative to traditional neural architectures, yet their application to speech processing remains under explored. This work presents the first investigation of KANs for Spoken Language Understanding (SLU) tasks. We experiment with 2D-CNN models on two datasets, integrating KAN layers in five different configurations within the dense block. The best-performing setup, which places a KAN layer between two linear layers, is directly applied to transformer-based models and evaluated on five SLU datasets with increasing complexity. Our results show that KAN layers can effectively replace the linear layers, achieving comparable or superior performance in most cases. Finally, we provide insights into how KAN and linear layers on top of transformers differently attend to input regions of the raw waveforms.

Index Terms: Kolmogorov-Arnold Networks, spoken language understanding, speech recognition, transformers

1. Introduction

Kolmogorov-Arnold Networks (KANs) have recently emerged as an alternative to traditional neural architectures, offering advantages in modeling complex, nonlinear relationships through learnable activation functions [1]. Inspired by the Kolmogorov-Arnold representation theorem [2–4], KANs replace fixed activation functions with learnable univariate functions, which enables them to approximate multivariate continuous functions. Preliminary works have demonstrated KAN's both advantages and limitations across various domains [5, 6]. However, recent studies suggest that they can offer superior modeling capabilites in specific applications, including computer vision [7, 8], medical image segmentation [9], and time-series forecasting [10, 11].

Despite their success in these areas, the application of KANs to speech-processing tasks remains largely underexplored. Spoken Language Understanding (SLU), in particular, is a fundamental component of human-computer interaction, enabling systems to extract meaning from spoken input and interpret user intentions [12, 13]. SLU plays a critical role in applications such as voice assistants, automated customer service, and voice-controlled smart devices, where accurate interpretation of natural language is essential for seamless interaction [14–16]. Although traditional neural networks have been widely adopted for SLU, the potential of KANs in this domain has yet to be investigated. We aim to bridge this gap by assessing whether the unique properties of KANs can be used to enhance performance on SLU tasks.

Recent studies have begun to explore the use of KANs in speech-related applications. For instance, [17] applied KANs to keyword spotting, highlighting their effectiveness in modeling high-level features in lower-dimensional space, while [18] introduced KANs for speech enhancement tasks, demonstrating their ability to improve speech quality in both time and frequency domains. However, these studies have focused primarily on low-level acoustic tasks, leaving the application of KANs to higher-level semantic tasks, such as SLU, unexamined.

In this work, we present the first investigation of KANs for SLU tasks. We consider the following exploration dimensions.

- **Integration options.** We experiment with five different configurations, integrating KAN layers within the final dense block of a 2D-CNN-based model operating on spectrogram inputs. These evaluations are carried out on two datasets: FLUENT SPEECH COMMANDS (FSC) [19] and TIMERS AND SUCH [20]. We also compare several function approximations within the KAN layer, finding that B-spline functions outperform Radial Basis Functions (RBF) [21, 22], Reflectional SWitch Activation Function (RSWAF) [23], Chebyshev [24], and Group-Rational KAN (GR-KAN) [25] alternatives.
- **Task complexity.** We apply the best-performing setup to two transformer-based models, wav2vec 2.0 [26] and XLS-R [27], and evaluate their performance across five datasets of increasing complexity: FSC, TIMERS AND SUCH, SLURP [28], ITALIC [29] in Italian, and SPEECH-MASSIVE [30] in German and French. Our experiments demonstrate that KAN layers can replace linear layers in SLU models, achieving comparable or superior performance in most cases without increasing model size or training time.
- **Error analysis.** Transformers equipped with KAN and linear layers differently attend to input regions of the raw waveform. We investigate cases where introducing a KAN layer corrects predictions, revealing patterns of behavior aligned with human reasoning.

Through this work, we aim to inspire further research into alternative neural architectures for speech processing, particularly in SLU applications.

2. Preliminaries

A KAN layer $L(x)$ models continuous functions by decomposing them into learnable univariate transformations. Formally, for an input vector $x \in \mathbb{R}^I$ and output dimension O, it can be expressed as:

$$L(x) = \Phi \circ x = \prod_{i=1}^{I} \phi_{i,1}(x_i) \cdots \prod_{i=1}^{I} \phi_{i,O}(x_i) \qquad (1)$$

where Φ is a matrix of learnable scalar functions $\phi_{i,o}(\cdot)$, with i indexing input dimensions and o indexing output dimensions.

10.21437/Interspeech.2025-1612

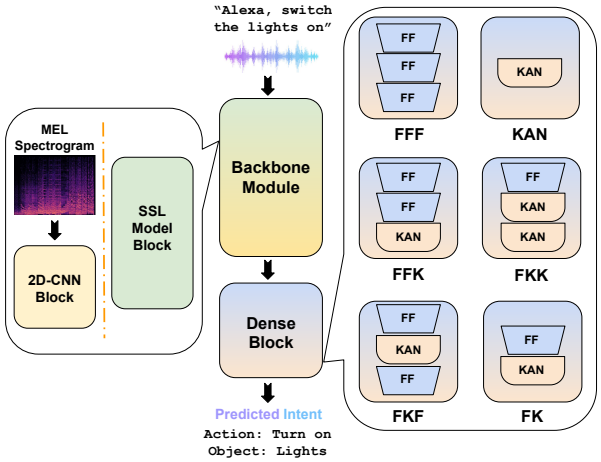


Figure 1: ***Proposed configurations' overview.*** FFF *is the baseline with a fully-connected MLP. The other configurations show five alternatives differently integrating linear and KAN layers.*

For efficiency, a KAN layer is typically approximated using a combination of a basis function (x) (e.g., Swish) and a *B-spline* function, which enables smooth curve fitting with lower-degree polynomials:

$$L(x) = w_1 b(x) + w_2 \text{spline}(x) \tag{2}$$

where w_1 and w_2 are learnable scalars. This formulation enhances flexibility while maintaining computational efficiency. Several alternatives have been proposed to better approximate functions in KAN layers. RBF [21, 22] provides smooth approximations by weighting basis functions centered at different points based on their distance from the input. Chebyshev polynomials [24] are a sequence of orthogonal polynomials defined recursively and commonly used in approximation theory due to their minimization of the maximum error, making them effective for function interpolation. RSWAF [23] approximates B-splines using a modified version of the Switch Activation Function having Reflectional symmetry. Group-Rational KAN (GR-KAN) [25] replaces B-splines with rational functions, aiming to enhance the expressiveness of the approximation through learnable rational transformations.

3. KAN in SLU

We first investigate the impact of architectural modifications on the dense classification block of a 2D-CNN, and then extend our analysis to transformer-based models.

The baseline architecture (FFF in Fig. 1) employs a 2D-CNN feature extractor followed by a dense block comprising a 3-layer Multi-Layer Perceptron (MLP). Each Feed-Forward (FF) layer within the MLP incorporates GELU activation, dropout regularization, and batch normalization. In formulas:

$$y = \rho(W_3 \cdot \text{Dropout}(\rho(W_2 \cdot \text{Dropout}(\rho(W_1 \cdot x + b_1)) + b_2)) + b_3)$$

where $x \in \mathcal{X}$ represents the input vector from the CNN feature extractor, $W_i \in \mathbb{R}^{n_i \times n_{i-1}}$ and $b_i \in \mathbb{R}^{n_i}$ are the weights and biases of the i-th layer, respectively, $\rho(\cdot)$ denotes the GELU activation function, and $y \in \mathcal{Y}$ is the output. This architecture serves as the basis for comparison with five configurations (Fig. 1) designed to explore the interplay between linear (FF) and KAN layers. We hypothesize that KAN layers can learn more efficient representations than linear layers by capturing nonlinear patterns directly. Through different configurations of FF and KAN layers, we aim to identify which structures best leverage their complementary strengths.

KAN: KAN-Only. This configuration replaces the entire 3-FF-layer MLP with a single KAN layer. While keeping the model parameters constrained, this simplification evaluates the raw representational capacity of a KAN layer. By directly mapping CNN-extracted features to the output space, we remove intermediate linear transformations, isolating the KAN layer's ability to capture complex, non-linear relationships in the data.

FFK and FKK: Varying KAN Placement. In configuration FFK, we position a single KAN layer at the end of the block, preceded by two FF layers. In contrast, FKK has one FF layer followed by two KAN layers. These designs investigate whether the KAN layer benefits from operating on a feature representation pre-processed by linear transformations. The FF layers might learn to disentangle or project the input into a space more tractable to the KAN-specific basis functions.

FKF: Embedded KAN Layer. Configuration FKF embeds a single KAN layer within two FF layers. We hypothesize that this arrangement can offer a balance between the global representation learning capabilities of FF layers and the localized, non-linear feature extraction of the KAN layer. The initial FF layer might perform a coarse-grained feature extraction, preparing the input for the KAN layer to focus on specific, higher-order relationships. The final FF layer can then integrate these non-linear features with the original linear projections, potentially leading to a richer and more discriminative representation.

FK: Minimal Hybrid Configuration. Finally, FK represents the most compact hybrid architecture, comprising a single FF layer followed by a single KAN layer. This configuration serves as a baseline for evaluating the effectiveness of combining even just one FF and one KAN layer. It allows us to determine if the benefits of incorporating a KAN layer can be realized even with a minimal increase in architectural complexity.

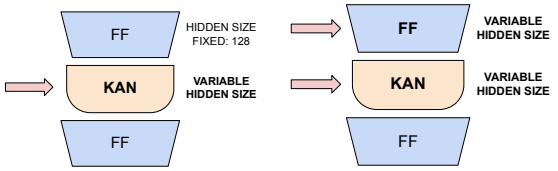
A key challenge in the original KAN implementation [1] arises from the computational demands of expanding intermediate variables. For a layer with input dimension d_{in} and output dimension d_{out}, the original approach requires expanding the input to a tensor of shape (B, d_{out}, d_{in}), to apply the activation functions, where B is the batch size. This expansion incurs significant memory overhead. However, recognizing that all activation functions within KAN are linear combinations of a fixed set of basis functions (e.g., B-splines), we exploit the more efficient recomputation available here[1]. Instead of expanding the input, we activate it with different basis functions and subsequently combine the results linearly. This drastically reduces memory consumption and transforms the computation into a straightforward matrix multiplication, naturally compatible with both forward and backward passes.

We evaluate different function approximations within KAN layers, comparing B-splines against RBF [22], Chebyshev [24] and RSWAF [23] alternatives. After identifying the best configurations from our initial experiments on CNNs, we extend these setups to transformer-based models. This allows us to evaluate their generalization capabilities across different architectures. By applying the same architectural modifications, we assess whether the observed improvements hold beyond CNNs.

[1] https://github.com/Blealtan/efficient-kan

Table 1: ***2D-CNN Performance***. *Performance comparison of several configurations with FF and KAN layers (as shown in Fig. 1) on two SLU datasets. The baseline with a fully-connected MLP is highlighted in light-blue. Best results in* ***bold****, second-best* underlined.

Config	Size	Training Time	FSC		Timers and Such	
			Accuracy	F1 Macro	Accuracy	F1 Macro
FFF	6.2M	1.82it/s	.555±.020	.549±.022	.471±.032	.455±.042
KAN	17.6M	1.71it/s	.452±.007	.445±.008	.416±.010	.364±.019
FFK	6.2M	1.96it/s	.559±.012	.550±.009	.565±.017	.533±.025
FKK	6.4M	1.75it/s	.533±.009	.527±.010	.553±.007	.515±.009
FKF	6.3M	1.95it/s	**.583±.006**	**.575±.002**	**.571±.011**	**.545±.022**
FK	6.2M	1.86it/s	.526±.015	.521±.013	.533±.029	.497±.032



(a) Fixed hidden size. (b) Variable hidden size.

Figure 2: ***Ablation on hidden size.*** *Fixed (a) and variable (b) hidden size study on* FKF *configuration.*

4. Experimental Setup

Datasets. We evaluate our approach on five publicly available intent classification datasets: FSC [19], Timers and Such [20], and SLURP [28] for English, ITALIC [29] for Italian, and SPEECH-MASSIVE [30] for German and French. The first two datasets are relatively simple, containing 31 and 4 intents, respectively. In contrast, SLURP, ITALIC, and SPEECH-MASSIVE are significantly larger, comprising 60 intents and offering greater linguistic diversity. ITALIC and SPEECH-MASSIVE can be considered multilingual extensions of SLURP, covering Italian, German, and French, respectively.

Among these datasets, TIMERS AND SUCH is unique in that it consists of spoken commands specifically designed for common voice control scenarios involving numerical inputs.

Models. We first train a 2D-CNN using Mel spectrograms as input. The architecture consists of four convolutional layers with progressively increasing output channels (16, 32, 64, and 128). Each layer employs a kernel size of 3, a stride of 1, and a padding of 2. Following each convolution, we apply two-dimensional batch normalization, GeLU activation function, and dropout. Max pooling is applied after the second and fourth convolutional layers to downsample the feature maps. A fully connected classification block follows the convolutional layers, comprising three linear layers, each followed by layer normalization, GeLU, and dropout. When replacing a feed-forward with a KAN layer, we substitute the entire block consisting of the linear layer, normalization, activation function, and dropout. Mel spectrograms are computed using 400 FFT size, a window length of 400, a hop length of 160, and 64 Mel filters. The model is trained for a maximum of 20 epochs with an early stopping criterion of 5 epochs. We use a batch size of 256, an initial learning rate of 1e-4, and AdamW optimizer with weight decay. Learning rate adjustments are handled via a plateau scheduler based on validation accuracy.

We also evaluate transformer models, specifically the monolingual wav2vec 2.0[2] and the multilingual XLS-R[3]. These models utilize the same dense classification block as the 2D-CNN. Their training setup follows the same configuration as described before, with a reduced batch size of 64, 5e-5 learning rate, and a maximum of 50 epochs.

The code to reproduce our experiments is available at `https://github.com/koudounasalkis/SLU-KAN`.

Table 2: ***Ablation on hidden size***. *Performance comparison of* FFF *and* FKF *configurations, the latter with fixed hidden size (as shown in Fig.2(a)) and variable hidden size (Fig.2(b)). The baseline is highlighted in light-blue.*

Config	Hidden Size	Model Size	Training Time	FSC	
				Accuracy	F1 Macro
FFF	128	6.2M	1.82it/s	.555±.020	.549±.022
FKF(a)	32	6.2M	1.97it/s	.554±.022	.533±.025
FKF(a)	64	6.2M	1.97it/s	.565±.011	.554±.012
FKF(a)	128	6.3M	1.95it/s	**.583±.006**	**.575±.002**
FKF(a)	256	6.5M	1.91it/s	.578±.019	.562±.023
FKF(a)	512	6.9M	1.89it/s	.552±.013	.544±.010
FKF(a)	1024	7.7M	1.85it/s	.547±.010	.543±.009
FKF(b)	32	1.6M	3.01it/s	.513±.013	.493±.014
FKF(b)	64	3.2M	2.69it/s	.533±.007	.520±.004
FKF(b)	128	6.3M	1.95it/s	.583±.006	.575±.002
FKF(b)	256	12.9M	1.80it/s	.586±.013	.584±.017
FKF(b)	512	27.4M	1.71it/s	**.631±.014**	**.628±.016**
FKF(b)	1024	61.2M	1.52it/s	.616±.029	.579±.032

5. Results

In the following, we present the wide variety of experiments performed to explore the behavior of KAN network integration.

5.1. Experimental results on 2D-CNN

Table 1 shows the results on the 2D-CNN. FFF serves as the baseline with only FF layers, providing a reference point for evaluating KAN-based configurations. KAN, which fully replaces FF layers with a single KAN layer, shows a notable drop in performance while increasing the number of parameters. This suggests that KAN alone struggles to stabilize feature transformations, and FF layers remain fundamental for structured representation learning. FFK, where a KAN layer is placed at the end of the FF stack, slightly improves performance, indicating that linear feature extraction first, followed by a nonlinear transformation with KAN, is beneficial. FKK, where 2 KAN layers follow a single FF layer, performs slightly worse than FFK, suggesting that KAN provides a very effective intermediate layer, but is less effective as final layer. FKF, which places KAN between two FF layers, achieves the best accuracy and F1 scores across both datasets while also converging slightly faster, showing that a balance between structured transformations from FF layers and flexible representations from KAN is optimal for SLU tasks. FK, where KAN is placed after a single FF layer, leads to lower performance similar to FKK. Overall, KAN layers provide improvements when properly integrated with FF layers, as seen in FKF, while full replacement, as in KAN, proves ineffective.

Analysis of hidden size. We analyze how the hidden size im-

[2] `huggingface.co/facebook/wav2vec2-base`
[3] `huggingface.co/facebook/wav2vec2-xls-r-300m`

Table 3: ***Ablation on approximation function***. *Comparison of* FFF *and* FKF *configurations, the latter with different approximation functions. The baseline is highlighted in light-blue .*

Config	Approx. Function	Model Size	Training Time	FSC	
				Accuracy	F1 Macro
FFF	-	6.2M	1.82it/s	.555±.020	.549±.022
FKF	B-Spline	6.3M	1.95it/s	**.583±.006**	**.575±.002**
FKF	RBF	6.3M	1.98it/s	.554±.005	.540±.007
FKF	RSWAF	6.3M	1.98it/s	.538±.009	.526±.011
FKF	Chebyshev	6.2M	1.65it/s	.569±.010	.564±.009
FKF	GR-KAN	6.2M	1.65it/s	.563±.009	.554±.011

Table 4: ***Transformer models performance***. *Comparison in terms of F1 macro of* FFF*,* FFK*, and* FKF *configurations on five SLU datasets. Monolingual wav2vec 2.0 model for the first block of datasets, multilingual XLS-R for the second. The baseline is highlighted in light-blue .*

Config	Size	FSC	Timers and Such	SLURP
FFF	94.6M	.990±.002	.976±.006	**.539±.010**
FFK	94.7M	.994±.001	.985±.003	.534±.006
FKF	95.4M	**.995±.001**	**.992±.004**	.531±.001

Config	Size	ITALIC	SPEECH-MASSIVE	
			de-DE	fr-FR
FFF	315.8M	.747±.004	.659±.011	.672±.008
FFK	315.9M	.734±.007	.642±.012	.678±.010
FKF	316.5M	**.749±.006**	**.667±.009**	**.679±.004**

pacts the final performance in Table 2. FKF(a) maintains a fixed hidden size of 128 for the initial FF layer while varying the KAN layer's output size. The results show that performance improves as the KAN output size increases up to 128, where accuracy and F1-score peak. Beyond this, performance shows a slight decline, indicating that very large KAN outputs might affect feature learning stability. Training time slightly decreases with smaller KAN output sizes but remains stable across configurations. FKF(b), where both FF and KAN hidden sizes are variable, exhibits a different trend. With small hidden sizes (32 and 64), performance is significantly lower, showing that a minimal feature space limits the network's ability to model complex speech patterns. As the hidden size increases, accuracy and F1-score improve, surpassing FKF(a) at 512 hidden units, where it achieves the highest scores. However, performance slightly drops at 1024, potentially due to overfitting or inefficient representation scaling. Model size and training time increase notably in FKF(b), with the largest configuration reaching over 60M params, suggesting that while increasing model capacity enhances performance, it comes with higher computational costs.

Analysis of approximation function. Table 3 analyzes diverse approximation functions. Among the different approximation functions tested in FKF, B-Spline achieves the highest scores. RBF and RSWAF show lower performance, indicating that not all approximation functions contribute positively in this context. Chebyshev and GR-KAN perform better than RBF and RSWAF, but both remain below B-Spline. The choice of approximation function thus significantly impacts performance, with B-Spline resulting the most effective in this setup.

5.2. Experimental results with transformers

Table 4 presents a performance comparison of three configurations (baseline FFF, FFK, and FKF) using transformer-based models across five SLU datasets. The first block corresponds to monolingual wav2vec 2.0 models, while the second block includes multilingual XLS-R models. The results show that FFK and FKF consistently outperform FFF in most datasets, though the differences are often marginal. For FSC and TIMERS AND SUCH, FFK and FKF reach near-perfect F1 scores. In SLURP, all models perform similarly, suggesting that the dataset may pose intrinsic challenges that are not addressed by architectural modifications. In the multilingual setting, performance gaps are slightly less pronounced. For ITALIC and SPEECH-MASSIVE, FKF achieves the best results, suggesting better generalization across languages. Despite these improvements, training times are nearly identical across configurations, indicating that the modifications in FFK and FKF do not introduce significant computational overhead. The results demonstrate that the configuration identified for 2D-CNN models and directly applied to transformer-based models generalizes well across datasets. This suggests that the architectural refinements in FKF are robust and transferable across different model types.

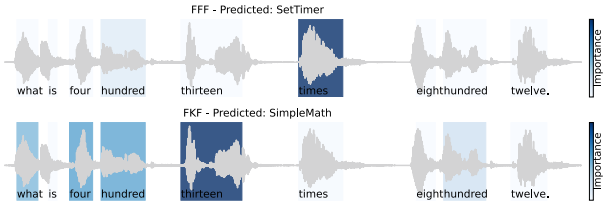


Figure 3: ***Example of word-level explanations*** *for* FFF *and* FKF *predictions;* TIMERS AND SUCH*, SimpleMath intent.*

5.3. Attention to input regions

We investigate whether introducing a KAN layer affects how the model attends to input data. We focus on transformer models and TIMERS AND SUCH, where the FKF configuration shows a marked improvement in performance. We analyze differences in how FFF and FKF models process inputs, particularly in cases where FFF makes incorrect predictions that FKF corrects. To explore this, we use an explanation technique that assigns relevance scores to word-level segments, indicating how much each spoken word impacts the prediction [31]. Fig. 3 provides an example: FFF incorrectly classifies an utterance to 'SetTimer', with its explanation showing that the model attends to the word "*times*". In contrast, FKF correctly classifies it to the 'SimpleMath' intent, focusing on numerical cues. This pattern holds across other misclassified samples, where FFF consistently assigns relevance to "*times*", suggesting it as a source of ambiguity, while FKF attends to more pertinent words. FKF's explanations align more closely with human reasoning, making them plausible [32] and suggesting KAN layers can be a valid alternative to linear ones.

6. Conclusion

This work explores the integration of KAN layers in SLU tasks, demonstrating their effectiveness across different model architectures and languages. Our experiments show that strategic placement of KAN layers between FF layers achieves optimal performance while maintaining computational efficiency comparable to traditional approaches. Multilingual generalization and the plausibility of model explanation suggest promising directions for future research in semantic speech processing.

7. Acknowledgments

This work is partially supported by the FAIR - Future Artificial Intelligence Research (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.3 – D.D. 1555 11/10/2022, PE00000013) and the spoke "FutureHPC & BigData" of the ICSC - Centro Nazionale di Ricerca in High-Performance Computing, Big Data and Quantum Computing, both funded by the European Union - NextGenerationEU, and by the "D.A.R.E. – Digital Lifelong Prevention" project (code: PNC0000002, CUP: B53C22006450001), co-funded by the Italian Complementary National Plan PNC-I.1 Research initiatives for innovative technologies and pathways in the health and welfare sector (D.D. 931 of 06/06/2022) and by the European Union – Next Generation EU under the National Recovery and Resilience Plan (PNRR) – M4 C2, Investment 1.1: Fondo per il Programma Nazionale di Ricerca e Progetti di Rilevante Interesse Nazionale (PRIN) - PRIN 2022 - "SHAPE-AD" (CUP: J53D23007240008). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

8. References

[1] Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljacic, T. Y. Hou, and M. Tegmark, "KAN: Kolmogorov–arnold networks," in *The Thirteenth International Conference on Learning Representations*, 2025.

[2] A. N. Kolmogorov, *On the representation of continuous functions of several variables by superpositions of continuous functions of a smaller number of variables*. American Mathematical Society, 1961.

[3] ——, "On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition," in *Doklady Akademii Nauk*, vol. 114, no. 5. Russian Academy of Sciences, 1957, pp. 953–956.

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

[5] R. Yu, W. Yu, and X. Wang, "Kan or mlp: A fairer comparison," *arXiv preprint arXiv:2407.16674*, 2024.

[6] E. Poeta, F. Giobergia, E. Pastor, T. Cerquitelli, and E. Baralis, "A benchmarking study of kolmogorov-arnold networks on tabular data," in *2024 IEEE 18th International Conference on Application of Information and Communication Technologies (AICT)*, 2024.

[7] B. Azam and N. Akhtar, "Suitability of kans for computer vision: A preliminary investigation," *arXiv preprint arXiv:2406.09087*, 2024.

[8] D. Rege Cambrin, E. Poeta, E. Pastor, T. Cerquitelli, E. Baralis, and P. Garza, "Kan you see it? kans and sentinel for effective and explainable crop field segmentation," in *Computer Vision – ECCV 2024 Workshops*. Cham: Springer Nature Switzerland, 2025, pp. 115–131.

[9] C. Li, X. Liu, W. Li, C. Wang, H. Liu, Y. Liu, Z. Chen, and Y. Yuan, "U-kan makes strong backbone for medical image segmentation and generation," *arXiv preprint arXiv:2406.02918*, 2024.

[10] K. Xu, L. Chen, and S. Wang, "Kolmogorov-arnold networks for time series: Bridging predictive power and interpretability," *arXiv preprint arXiv:2406.02496*, 2024.

[11] X. Han, X. Zhang, Y. Wu, Z. Zhang, and Z. Wu, "Kan4tsf: Are kan and kan-based models effective for time series forecasting?" *arXiv preprint arXiv:2408.11306*, 2024.

[12] G. Tur and R. De Mori, *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons, 2011.

[13] A. Koudounas, F. Giobergia, E. Pastor, and E. Baralis, "A contrastive learning approach to mitigate bias in speech models," in *Proc. INTERSPEECH 2024*, 2024, pp. 827–831.

[14] D. Serdyuk, Y. Wang, C. Fuegen, A. Kumar, B. Liu, and Y. Bengio, "Towards end-to-end spoken language understanding," in *ICASSP*, 2018.

[15] A. Koudounas, E. Pastor, G. Attanasio, L. de Alfaro, and E. Baralis, "Prioritizing data acquisition for end-to-end speech model improvement," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 7000–7004.

[16] A. Saade, J. Dureau, D. Leroy, F. Caltagirone, A. Coucke, A. Ball, C. Doumouro, T. Lavril, A. Caulier, T. Bluche, T. Gisselbrecht, and M. Primet, "Spoken language understanding on the edge," in *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS Edition (EMC2-NIPS)*, 2019.

[17] A. Xu, B. Zhang, S. Kong, Y. Huang, Z. Yang, S. Srivastava, and M. Sun, "Effective integration of kan for keyword spotting," in *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2025, pp. 1–5.

[18] H. Li, Y. Hu, C. Chen, and E. S. Chng, "An investigation on the potential of kan in speech enhancement," *arXiv preprint arXiv:2412.17778*, 2024.

[19] L. Lugosch, M. Ravanelli, P. Ignoto, V. S. Tomar, and Y. Bengio, "Speech model pre-training for end-to-end spoken language understanding," in *Proc. INTERSPEECH*, 2019.

[20] L. Lugosch, P. Papreja, M. Ravanelli, A. HEBA, and T. Parcollet, "Timers and such: A practical benchmark for spoken language understanding with numbers," in *NeurIPS Datasets and Benchmarks Track*, 2021.

[21] M. J. Orr *et al.*, "Introduction to radial basis function networks," 1996.

[22] Z. Li, "Kolmogorov-arnold networks are radial basis function networks," *arXiv preprint arXiv:2405.06721*, 2024.

[23] A. Delis, "Fasterkan," https://github.com/AthanasiosDelis/faster-kan/, 2024.

[24] N. Karjanto, "Properties of chebyshev polynomials," *arXiv preprint arXiv:2002.01342*, 2020.

[25] X. Yang and X. Wang, "Kolmogorov-arnold transformer," *arXiv preprint arXiv:2409.10594*, 2024.

[26] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *NeurIPS*, 2020.

[27] A. Babu, C. Wang, A. Tjandra, K. Lakhotia, Q. Xu, N. Goyal, K. Singh, P. von Platen, Y. Saraf, J. Pino *et al.*, "Xls-r: Self-supervised cross-lingual speech representation learning at scale," *Interspeech*, 2022.

[28] E. Bastianelli, A. Vanzo, P. Swietojanski, and V. Rieser, "SLURP: A spoken language understanding resource package," in *EMNLP*, 2020.

[29] A. Koudounas, M. La Quatra, L. Vaiani, L. Colomba, G. Attanasio, E. Pastor, L. Cagliero, and E. Baralis, "ITALIC: An Italian Intent Classification Dataset," in *Proc. INTERSPEECH 2023*, 2023, pp. 2153–2157.

[30] B. Lee, I. Calapodescu, M. Gaido, M. Negri, and L. Besacier, "Speech-massive: A multilingual speech dataset for slu and beyond," in *Proc. INTERSPEECH*, 2024, pp. 817–821.

[31] E. Pastor, A. Koudounas, G. Attanasio, D. Hovy, and E. Baralis, "Explaining speech classification models via word-level audio segments and paralinguistic features," in *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*. St. Julian's, Malta: Association for Computational Linguistics, Mar. 2024, pp. 2221–2238.

[32] A. Jacovi and Y. Goldberg, "Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness?" in *ACL*, 2020.