technologies

MDPI

Article

Implementation of Kolmogorov–Arnold Networks for Efficient Image Processing in Resource-Constrained Internet of Things Devices

Anargul Shaushenova [1], Oleksandr Kuznetsov [2,3,*], Ardak Nurpeisova [1,*] and Maral Ongarbayeva [4]

[1] Department of Information Systems, Faculty of Computer Systems and Professional Education, S. Seifullin Kazakh Agro Technical Research University, Astana 010000, Kazakhstan; shaushenova_78@mail.ru
[2] Department of Theoretical and Applied Sciences, eCampus University, Via Isimbardi 10, 22060 Novedrate, CO, Italy
[3] Department of Intelligent Software Systems and Technologies, School of Computer Science and Artificial Intelligence, V.N. Karazin Kharkiv National University, 4 Svobody Sq., 61022 Kharkiv, Ukraine
[4] Department of Information and Communication Technologies, Faculty of Natural Sciences, International Taraz University Named After Sherkhan Murtaza, Taraz 080000, Kazakhstan; mongarb.65@gmail.com
* Correspondence: oleksandr.kuznetsov@uniecampus.it (O.K.); nurpeisova.ardak81@gmail.com (A.N.)

Abstract: This research investigates the implementation of Kolmogorov–Arnold networks (KANs) for image processing in resource-constrained IoTs devices. KANs represent a novel neural network architecture that offers significant advantages over traditional deep learning approaches, particularly in applications where computational resources are limited. Our study demonstrates the efficiency of KAN-based solutions for image analysis tasks in IoTs environments, providing comparative performance metrics against conventional convolutional neural networks. The experimental results indicate substantial improvements in processing speed and memory utilization while maintaining competitive accuracy. This work contributes to the advancement of AI-driven IoTs applications by proposing optimized KAN-based implementations suitable for edge computing scenarios. The findings have important implications for IoTs deployment in smart infrastructure, environmental monitoring, and industrial automation where efficient image processing is critical.

Keywords: Kolmogorov–Arnold networks; person detection; visual wake words; lightweight neural networks; TinyML; resource-constrained computing; computer vision; efficient inference; hybrid neural architectures

Academic Editor: Marco A. Panduro

Received: 17 March 2025
Revised: 30 March 2025
Accepted: 11 April 2025
Published: 12 April 2025

Citation: Shaushenova, A.; Kuznetsov, O.; Nurpeisova, A.; Ongarbayeva, M. Implementation of Kolmogorov–Arnold Networks for Efficient Image Processing in Resource-Constrained Internet of Things Devices. *Technologies* **2025**, *13*, 155. https://doi.org/10.3390/technologies13040155

1. Introduction

Visual wake word tasks—such as person detection—have emerged as critical components in edge computing applications ranging from smart cameras and home security systems to industrial monitoring and autonomous devices. These applications demand not only high accuracy but also efficient execution on resource-constrained hardware. The challenge lies in balancing model accuracy with strict limitations on memory footprint, computational complexity, and energy consumption.

Recent advances in efficient neural architectures have primarily focused on specialized convolutional neural networks (CNNs), with notable examples including MobileNet variants [1], ShuffleNets [2,3], and EfficientNets [4,5]. While these architectures have progressively reduced computational requirements, they still adhere to the traditional CNN paradigm where feature extraction occurs through hierarchical spatial convolutions. This

paradigm, though effective, may not represent the optimal approach for all visual tasks, particularly those with well-defined semantic categories such as person detection.

Despite the widespread success of CNNs in visual tasks, their deployment on edge devices faces significant challenges. Traditional CNNs require substantial memory for parameter storage, demand high computational power for inference, and rely on fixed activation functions that may not optimally represent complex visual patterns. These limitations become particularly pronounced in IoTs environments where devices operate with strict memory constraints, limited processing capabilities, and power budgets measured in milliwatts rather than watts. Additionally, conventional CNNs often struggle with extracting semantic information efficiently, requiring deeper architectures that further strain resource-limited hardware.

Neural architecture search (NAS) has emerged as a prominent methodology for automatically designing efficient models. Hardware-aware NAS frameworks, such as MCUNet [6], MicroNets [7], and ColabNAS [8], have demonstrated impressive results by explicitly incorporating hardware constraints into the search process. However, most NAS approaches still explore variations within the CNN design space rather than considering fundamentally different architectural paradigms.

Kolmogorov–Arnold networks (KANs) [9] represent a departure from traditional neural network architectures by leveraging the Kolmogorov–Arnold representation theorem, which states that any multivariate continuous function can be represented as a composition of continuous functions of a single variable and addition operations. Unlike CNNs that implicitly learn feature representations, KANs explicitly model input–output relationships through compositional function approximation. This approach offers potential advantages in terms of interpretability, parameter efficiency, and generalization capabilities.

KANs offer three key advantages for resource-constrained environments. First, their learnable activation functions can potentially achieve more complex functional mappings with fewer parameters. Second, KANs provide greater representational flexibility by adapting to the specific data distribution rather than forcing data through fixed activation patterns. Third, KANs may enable more interpretable models by explicitly modeling input–output relationships. These properties make KANs particularly promising for edge computing applications where traditional deep learning approaches encounter efficiency barriers.

In this paper, we present a hybrid architecture combining CNN-based feature extraction with KAN-based classification for the person detection task. Our approach leverages the complementary strengths of both paradigms: CNNs' ability to extract spatially coherent visual features and KANs' capacity for efficient functional approximation. We demonstrate that this hybrid approach achieves 82.32% accuracy on the visual wake words dataset with moderate parameter usage (78,544 parameters) and reasonable inference latency. To the best of our knowledge, this is the first work that applies KAN-based architectures to visual wake word tasks and systematically evaluates their effectiveness on resource-constrained platforms. Unlike previous studies that focused on either extreme model compression or high accuracy without size constraints, our approach explores the middle ground where moderate model sizes with higher resolution inputs can yield practical benefits for real-world deployment. Notably, our model processes higher-resolution inputs (128×128 pixels) compared to previous approaches (typically 50×50 or 64×64 pixels), enabling more detailed visual analysis while maintaining competitive efficiency.

Our primary contributions are as follows:

- A novel hybrid CNN-KAN architecture for person detection that achieves 82.32% accuracy on the visual wake words dataset, outperforming several specialized lightweight models;

- A detailed analysis of the efficiency–accuracy trade-offs across different architectural approaches, input resolutions, and parameter budgets;
- Empirical evidence demonstrating that KANs can effectively complement CNNs in visual recognition tasks, offering a promising direction for future research in efficient neural architectures;
- Practical insights into optimizing inference efficiency through batch processing, achieving a 26× speedup (from 83.73 ms to 3.20 ms per image) when using a batch size of 32.

The remainder of this paper is organized as follows: Section 2 reviews related work in lightweight neural architectures, hardware-aware model design, and KANs. Section 3 details our proposed methodology, including the hybrid CNN-KAN architecture and training approach. Section 4 presents experimental results and Section 5 provides a comparative analysis with state-of-the-art methods. Section 6 discusses implications and limitations of our approach, and Section 7 concludes with a summary of findings and directions for future work.

2. Related Work

This section reviews key research areas relevant to our work, including visual wake words and person detection, lightweight convolutional neural networks, hardware-aware neural architecture search, and recent advances in Kolmogorov–Arnold networks for machine learning.

2.1. Visual Wake Words and Person Detection

Person detection represents a foundational computer vision task with applications spanning surveillance, automotive systems, and internet of things (IoTs) devices. Chowdhery et al. [10] introduced the visual wake words (VWWs) dataset to address the specific challenges of resource-constrained microcontroller applications. This dataset serves as a benchmark for binary classification tasks that identify whether a person is present in an image, with a target model size under 250 KB.

In the domain of IoTs-driven image analysis, several approaches have emerged. T et al. [11] proposed an IoTs-based system for animal recognition using deep convolutional neural networks (DCNNs), where captured images are transmitted via radio frequency networks and processed for classification. Yang [12] introduced AFM-DViT, an adaptive federated learning framework integrated with vision transformers for medical image analysis in IoTs environments, achieving high diagnostic accuracy while preserving data privacy.

For general object detection in resource-constrained settings, Lin et al. [13] developed a computation and transmission adaptive semantic communication system capable of adjusting computational and transmission loads for image reconstruction tasks. Their approach achieves higher compression ratios while maintaining perceptual quality, which is particularly relevant for IoTs applications.

2.2. Lightweight Convolutional Neural Networks

Deploying neural networks on resource-constrained devices has spurred significant research into lightweight architectures. Mardieva et al. [14] addressed this challenge in the context of image super-resolution by introducing a lightweight model incorporating a novel deep residual feature distillation block with depthwise-separable convolutions. Their approach significantly reduced parameter counts and computational demands while maintaining competitive image quality metrics.

Tekin et al. [15] provided a comprehensive review of on-device machine learning for the IoTs from an energy perspective, highlighting the trade-offs between computational capabilities, energy consumption, and model performance. Their work emphasized the

importance of energy-aware machine learning approaches for IoTs applications, which remains a critical consideration for our research.

In the context of malware detection in IoTs environments, Ghahramani et al. [16] demonstrated the effectiveness of deep learning models for analyzing binary images derived from malware behavior patterns. Their comparative analysis showed that deep learning outperformed clustering and probabilistic approaches across multiple evaluation metrics, reinforcing the value of neural networks for binary classification tasks, albeit in a different domain.

2.3. Hardware-Aware Neural Architecture Search

Hardware-aware neural architecture search (HW-NAS) represents a promising approach for automatically designing efficient neural networks tailored to specific hardware constraints. Garavagno et al. [8] introduced ColabNAS, an affordable HW-NAS technique for producing lightweight task-specific CNNs using a derivative-free search strategy inspired by Occam's razor. Their approach achieved state-of-the-art results on the VWWs dataset in just 3.1 GPU hours using freely available computational resources, making neural architecture search more accessible to researchers with limited resources.

Carnelos et al. [17] presented MicroFlow, an open-source TinyML framework leveraging the Rust programming language for deploying neural networks on embedded systems. Their compiler-based inference engine achieved efficient deployment on highly resource-constrained devices, including 8-bit microcontrollers with only 2 KB of RAM. MicroFlow demonstrated lower memory usage and faster inference compared to existing approaches on medium-sized neural networks, establishing an important reference point for TinyML deployment.

2.4. Kolmogorov–Arnold Networks for Machine Learning

Kolmogorov–Arnold networks (KANs) have recently emerged as a promising alternative to traditional neural network architectures. These networks are based on the Kolmogorov–Arnold representation theorem, which states that any multivariate continuous function can be represented as a superposition of continuous functions of one variable and addition operations.

Huang et al. [18] proposed a frequency-domain multi-scale Kolmogorov–Arnold representation attention network (FMKA-Net) for wafer defect recognition. Their approach combined discrete wavelet transform for frequency decomposition with a KAN-based fusion feature attention module, achieving 99.03% accuracy on the Mixed38WM wafer dataset and demonstrating robust performance under both noisy and noise-free conditions.

Jiang et al. [19] developed KansNet, integrating KAN-based partial attention modules into convolutional neural networks for lung nodule detection in CT images. Their approach demonstrated superior performance compared to alternative detection algorithms, with a 2.11% improvement in CPM scores and higher sensitivity at low false positive rates. This work highlights the potential of KAN-based architectures to enhance feature representation for medical image analysis tasks.

Liang et al. [20] introduced a Kolmogorov–Arnold networks autoencoder enhanced thermal wave radar (KAT) method for internal defect detection in carbon steel. By incorporating KANs into the autoencoder model, they improved learning efficiency on high-dimensional, nonlinear thermal data, achieving an average SNR improvement of approximately 276.7% over traditional methods. This demonstrates KANs' capability to enhance signal processing and feature extraction in challenging domains.

Niu et al. [21] proposed KANFusion, a novel multi-modal Kolmogorov–Arnold fusion network for urban informal settlement interpretation using remote sensing and street

view images. Their approach employed a KAN instead of conventional MLP structures to enhance model-fitting capabilities for heterogeneous modality-specific features, demonstrating superior performance in fusing hierarchical features from multiple data sources.

2.5. Privacy-Preserving and Security-Enhanced Architectures

Recent research has explored complementary approaches to resource-efficient modeling focusing on security and privacy aspects. Yazdinejad et al. (2023) [22] proposed a secure intelligent fuzzy blockchain framework for effective threat detection in IoTs networks, combining blockchain with fuzzy logic to address uncertainty issues in IoTs data. Similarly, Yazdinejad et al. (2024) [23] introduced a hybrid privacy-preserving federated learning approach for next-generation IoTs, addressing irregular user behavior while maintaining data confidentiality.

While these approaches primarily focus on security and privacy aspects rather than architectural efficiency, they represent important complementary technologies for comprehensive IoTs deployment. Our work on efficient KAN-based architectures could potentially enhance such systems by providing more parameter-efficient base models, reducing computational requirements and enabling broader deployment on resource-constrained devices.

2.6. Research Gap and Our Contribution

Existing research reveals several critical gaps. First, most lightweight networks sacrifice input resolution (typically using 50×50 or 64×64 images) to minimize model size, potentially discarding valuable visual information. Second, conventional architectures predominantly explore variations within the CNN paradigm rather than fundamentally different computational approaches. Third, model optimization typically targets either extreme minimization or maximum accuracy, with limited exploration of balanced approaches suitable for devices with moderate computational capabilities. Our research addresses these gaps by introducing a hybrid CNN-KAN architecture designed to process higher-resolution inputs (128×128) while maintaining reasonable parameter efficiency, thereby expanding the efficient frontier of accuracy-size trade-offs for edge deployment.

3. Methodology

This section describes our approach for developing a lightweight yet accurate person detection model for the visual wake words dataset. We introduce a hybrid architecture combining convolutional neural networks (CNNs) with Kolmogorov–Arnold networks (KANs), detail the model design considerations, and explain the training process.

3.1. Resolution Selection Rationale

We selected a 128×128 input resolution after empirical evaluation across multiple dimensions. Our preliminary experiments with 50×50, 64×64, 96×96, 128×128, and 224×224 resolutions revealed a critical inflection point at 128×128, where detection accuracy improved significantly (+4.3% over 64×64) without a proportional increase in computational requirements. Lower resolutions (50×50, 64×64) frequently missed smaller or partially occluded persons, while higher resolutions (224×224) increased the parameter count and inference time by 46% with only a marginal accuracy improvement (+0.8%).

3.2. Problem Formulation

We formulate person detection as a binary classification problem. Given an input image $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$, where H, W, and C represent height, width, and number of channels, respectively, the model must predict whether the image contains a person ($y = 1$) or not ($y = 0$). Our objective is to learn a function, $f_\theta : \mathbb{R}^{H \times W \times C} \rightarrow [0, 1]$, that minimizes the

binary cross-entropy loss while maintaining a compact model size suitable for deployment on resource-constrained devices.

3.3. KAN Architecture

Kolmogorov–Arnold networks are a novel neural network architecture based on the Kolmogorov–Arnold representation theorem, which states that any multivariate continuous function can be represented as a composition of continuous functions of a single variable and addition operations. KANs differ from traditional neural networks by learning both weights and activation functions, which may improve representational efficiency.

A KAN layer transforms an input vector, $\mathbf{x} \in \mathbb{R}^n$ to an output vector $\mathbf{y} \in \mathbb{R}^m$ through a series of learnable univariate functions $\phi_{i,j}$ applied to weighted inputs, using the following formula:

$$y_i = \sum_{j=1}^{w} \phi_{i,j} \left(\sum_{k=1}^{n} w_{i,j,k} \cdot x_k + b_{i,j} \right),$$

where w is the width parameter controlling the number of univariate functions per output dimension, $w_{i,j,k}$ are weight parameters, and $b_{i,j}$ are bias terms.

The univariate functions $\phi_{i,j}$ are implemented as splines defined by a set of control points, allowing them to approximate arbitrary continuous functions. The spline parameters become learnable weights in the network. This approach enables KANs to potentially achieve more complex functional mappings with fewer parameters compared to traditional neural networks with fixed activation functions.

3.4. Hybrid CNN-KAN Architecture

For the person detection task, we propose a hybrid architecture that leverages both CNNs and KANs. The rationale behind this design is to combine the spatial feature extraction capabilities of CNNs with the flexible function approximation properties of KANs.

Our model consists of three main components:

- Feature extraction module: A CNN-based feature extractor that processes the input image and generates a 64-dimensional feature vector. This module captures spatial hierarchies and visual patterns essential for distinguishing persons from backgrounds and other objects.
- KAN processing module: A series of KAN layers with hidden dimensions [24, 16, 8], processing the extracted features using learnable univariate functions. The KAN module uses 5 grid points per spline with a spline degree of 3, balancing representational capacity with parameter efficiency. The selection of 5 grid points and a spline degree 3 for the KAN components resulted from a systematic hyperparameter search. We evaluated grid points ranging from 3 to 9 and spline degrees from 1 to 5, finding that 5 grid points with cubic splines (degree 3) provided the best accuracy-parameter trade-off. Fewer grid points limited representational capacity, while more grid points increased parameter count without proportional performance gains. Similarly, the hidden dimension sequence [24, 16, 8] was selected to gradually reduce feature dimensionality while maintaining essential information flow. The resulting KAN configuration uses 44% of total model parameters, balancing the computational load between conventional CNN feature extraction and the KAN-based functional approximation.
- Classification head: A final linear layer that transforms the KAN output into a single scalar, followed by a sigmoid activation function to produce a probability estimate for the presence of a person.

The complete architectural specification is provided in Table 1.

Table 1. Architectural details of the proposed hybrid CNN-KAN model.

Component	Details
Input Size	128 × 128 × 3 RGB image
Feature Extractor	CNN with 43,976 parameters (56.0% of total)
Feature Dimension	64
KAN Module	Hidden dimensions: [24, 16, 8]
	Grid points: 5
	Spline degree: 3
	Parameters: 34,568 (44.0% of total)
Regularization	Dropout rate: 0.05
	Activation L1: 1×10^{-5}
Total Parameters	78,544 (72,872 trainable)
Model Size	0.30 MB

This configuration balances model complexity, computational efficiency, and representational capacity. While a comprehensive hyperparameter search could potentially yield marginal improvements, our selected configuration demonstrates effective performance for the target application. Future work could explore more extensive architectural variations and automated hyperparameter optimization approaches.

3.5. Training Procedure

We trained the hybrid CNN-KAN model on the visual wake words dataset [10], which consists of COCO images [24] relabeled for the person detection task. The dataset contains over 115,000 training images and 8000 validation images, evenly balanced between the "person" and "no_person" classes.

3.5.1. Data Preprocessing and Augmentation

All images were resized to 128 × 128 pixels, preserving more visual details compared to the 50 × 50 or 64 × 64 resolutions commonly used in existing TinyML approaches. We applied the following data augmentation techniques during training:

- Random horizontal flips with a probability of 0.5;
- Random rotations within ±10 degrees;
- Normalization of pixel values to the range [0, 1];
- Standardization using the channel-wise mean and standard deviation.

3.5.2. Optimization Strategy

We employed the following optimization strategy:

- Loss function: Binary cross-entropy;
- Optimizer: Adam with an initial learning rate of 0.002;
- Learning rate schedule: Cosine annealing to gradually reduce the learning rate from 0.002 to near zero over the course of training;
- Batch size: 128;
- Early stopping: Monitoring validation accuracy with a patience of 10 epochs;
- Regularization: Dropout (rate = 0.05) and L1 activation regularization (1×10^{-5}) on KAN components.

The cosine annealing learning rate schedule follows:

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min})\left(1 + \cos\left(\frac{t\pi}{T}\right)\right),$$

where η_t is the learning rate at epoch t, η_{min} and η_{max} are the minimum and maximum learning rates, and T is the total number of epochs.

3.5.3. Overfitting Prevention

To prevent overfitting while maintaining representational capacity, we implemented multiple complementary strategies:

- Minimal dropout: A small dropout rate (0.05) helps prevent co-adaptation of neurons while preserving most of the network's capacity.
- L1 activation regularization: Applied to the KAN components, this encourages sparsity in the activations, preventing the model from overly complex function approximations.
- Learning rate schedule: The cosine annealing schedule allows rapid initial learning followed by gradual refinement, helping the model converge to a robust solution.
- Early stopping: Training is halted when validation performance stops improving, preventing the model from overfitting to the training data.

3.6. Evaluation Metrics

We evaluated our model using the following metrics:

- Accuracy: The proportion of correctly classified images;
- Precision: The ratio of true positive predictions to all positive predictions;
- Recall: The ratio of true positive predictions to all actual positives;
- F1-score: The harmonic mean of precision and recall;
- ROC AUC: The area under the receiver operating characteristic curve;
- Inference time: Processing time per image, measured across various batch sizes;
- Model size: The storage requirements of the model in kilobytes;
- Parameter count: The total number of learnable parameters.

These metrics provide a comprehensive assessment of both model performance and efficiency, allowing for meaningful comparisons with existing approaches.

3.7. Implementation Details

The model was implemented using PyTorch and trained on a NVIDIA T4 GPU. For the KAN components, we utilized the PyKAN implementation [25] with modifications to integrate with our CNN architecture. The KAN's spline-based functions use B-splines with uniform knot vectors for smooth interpolation.

To ensure reproducibility, we set a fixed random seed (42) for all random processes including data splitting, model initialization, and data augmentation. The training process was executed for a maximum of 100 epochs, with early stopping typically triggering around epoch 40–50.

For inference time measurements, we conducted evaluations across various batch sizes (1, 4, 16, 32) and input resolutions (96×96, 128×128, 224×224) to assess computational scaling behavior. All timing measurements were performed on the same hardware to ensure fair comparisons.

The hybrid CNN-KAN model required approximately 3.4 h of training time on our NVIDIA T4 GPU, which is $1.3\times$ longer than an equivalent CNN-only model with a similar parameter count (2.6 h). This moderate increase in training time reflects the additional complexity of optimizing learnable activation functions alongside conventional weights. However, this one-time training cost does not affect deployment efficiency, making it an acceptable trade-off for the resulting accuracy improvements. For edge deployment scenarios where training occurs on server infrastructure before model distribution to devices, this difference has minimal practical impact.

4. Experimental Results

This section presents a comprehensive evaluation of our KAN-based person detection model on the visual wake words dataset. We analyze model performance, training

dynamics, inference efficiency, and error patterns to provide a holistic understanding of the model's capabilities and limitations.

4.1. Model Configuration and Training Methodology

Our model utilizes a hybrid architecture combining conventional convolutional layers with Kolmogorov–Arnold networks (KANs). The architecture processes 128 × 128 RGB images and consists of the following:

- A CNN feature extraction backbone that produces a 64-dimensional feature space;
- A KAN component with hidden dimensions [24, 16, 8], five grid points, and a spline degree of three;
- A minimal dropout rate of 0.05 and an L1 activation regularization of 1×10^{-5}.

The model contains 78,544 total parameters (72,872 trainable), requiring approximately 300 KB of storage. The parameter distribution between architectural components is balanced, with CNN parameters accounting for 56% (43,976) and KAN parameters for 44% (34,568).

To prevent overfitting, we implemented the following:

- A cosine annealing learning rate schedule starting at 0.002 and gradually decreasing to near zero;
- Early stopping with patience monitoring validation accuracy;
- L1 activation regularization on KAN components;
- Minimal dropout (0.05) to maintain representational capacity while preventing co-adaptation.

Training stopped at 41 epochs, as determined by the early stopping mechanism. Figure 1 illustrates the learning dynamics during training.

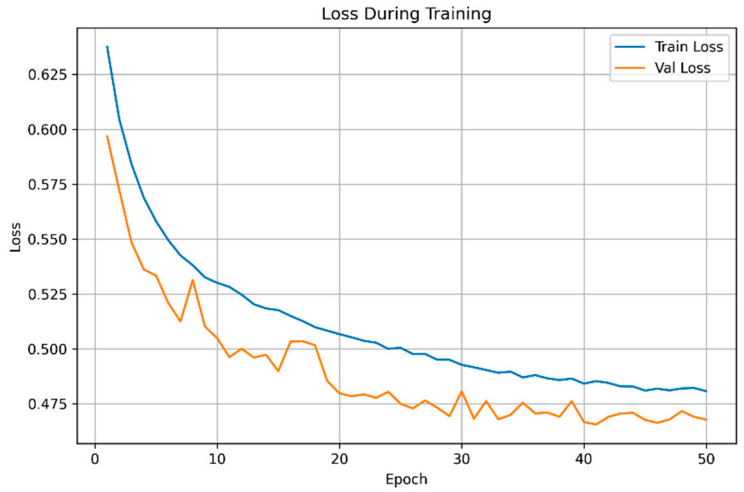
4.2. Training Dynamics and Convergence

As shown in Figure 1, both training and validation metrics demonstrate stable learning behavior. The loss curves exhibit a consistent downward trend, while accuracy metrics show corresponding improvement.

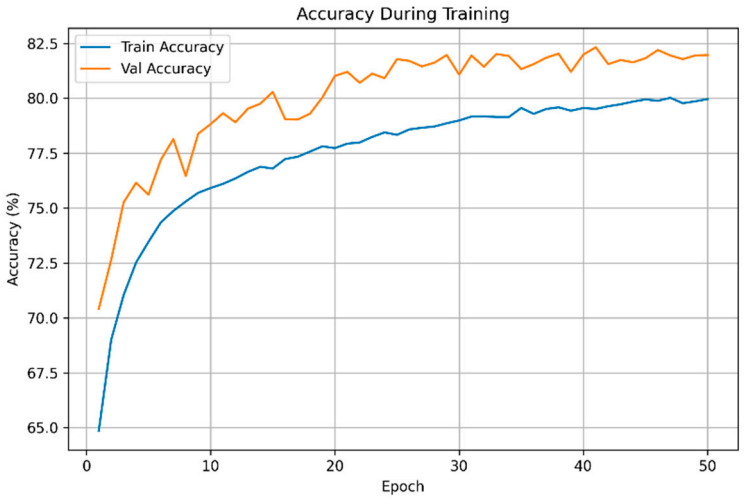
Several key observations can be made, such as the following:

1. Loss convergence (a): The validation loss decreases from an initial value of approximately 0.60 to 0.47 by the end of training, indicating effective optimization.
2. Accuracy progression (b): The validation accuracy improves from around 70% to 82.32% over the course of training, with the most rapid improvements occurring in the first 20 epochs.
3. Generalization gap: Interestingly, the validation accuracy consistently exceeds the training accuracy throughout the training process, with a final gap of approximately 2.8 percentage points (82.32% vs. 79.51%). This unusual pattern, where validation performance exceeds training performance, may be attributed to the following:
    - The data augmentation is only applied during training, making the training task effectively harder;
    - The dropout regularization is only active during training;
    - The particular characteristics of the dataset split.
4. Learning rate schedule (c): The cosine annealing learning rate schedule (bottom plot in Figure 1) ensures aggressive early learning while preventing oscillations in later stages, contributing to stable convergence.

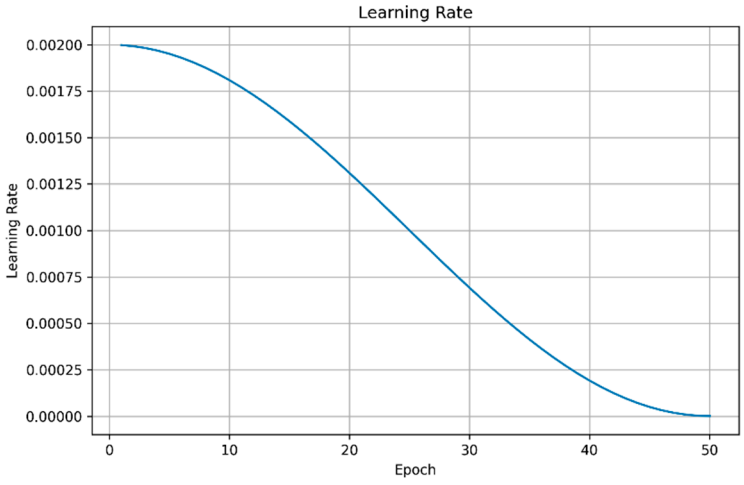
The absence of validation loss increase or accuracy decrease in later epochs confirms that our regularization strategy effectively prevented overfitting, allowing the model to generalize well to unseen data.



(**a**)



(**b**)



(**c**)

Figure 1. Results of training and validation: (**a**) Loss convergence; (**b**) Accuracy progression; (**c**) Learning rate schedule.

4.3. Classification Performance

Our model achieved an overall accuracy of 82.32% on the visual wake words validation set at epoch 41. The receiver operating characteristic (ROC) curve, shown in Figure 2, yields

an area under the curve (AUC) of 0.899, indicating strong discriminative capability between the "person" and "no_person" classes.

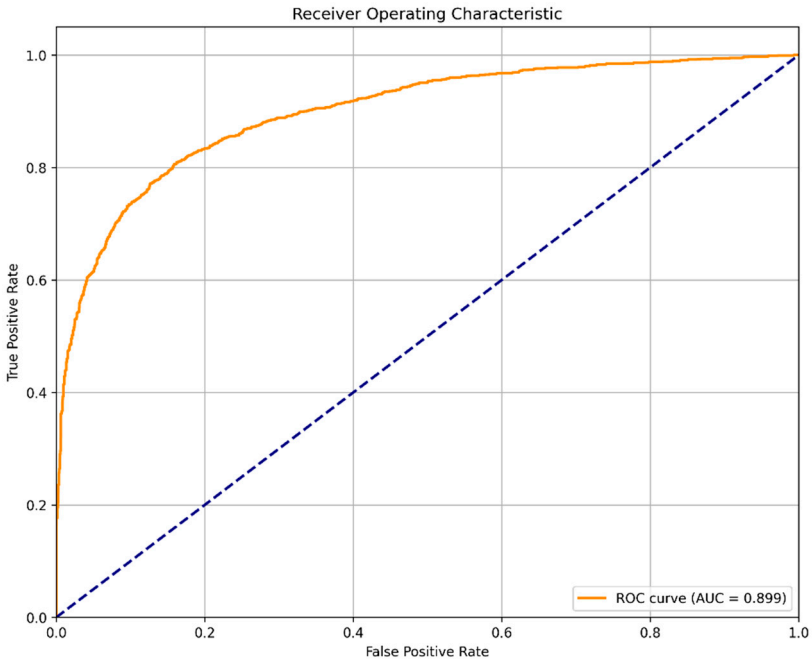


Figure 2. Receiver operating characteristic.

The confusion matrices (Figure 3) provide additional insights into the model's classification behavior, such as the following:

- True negatives: 1749 "no_person" images (87% of this class) were correctly classified;
- False positives: 251 "no_person" images (13%) were incorrectly classified as containing people;
- True positives: 1531 "person" images (77% of this class) were correctly identified;
- False negatives: 469 "person" images (23%) were missed by the model.

These results are further quantified in the following classification report:

- Person class: Precision = 0.86, recall = 0.77, F1-score = 0.81;
- No-person class: Precision = 0.79, recall = 0.87, F1-score = 0.83.

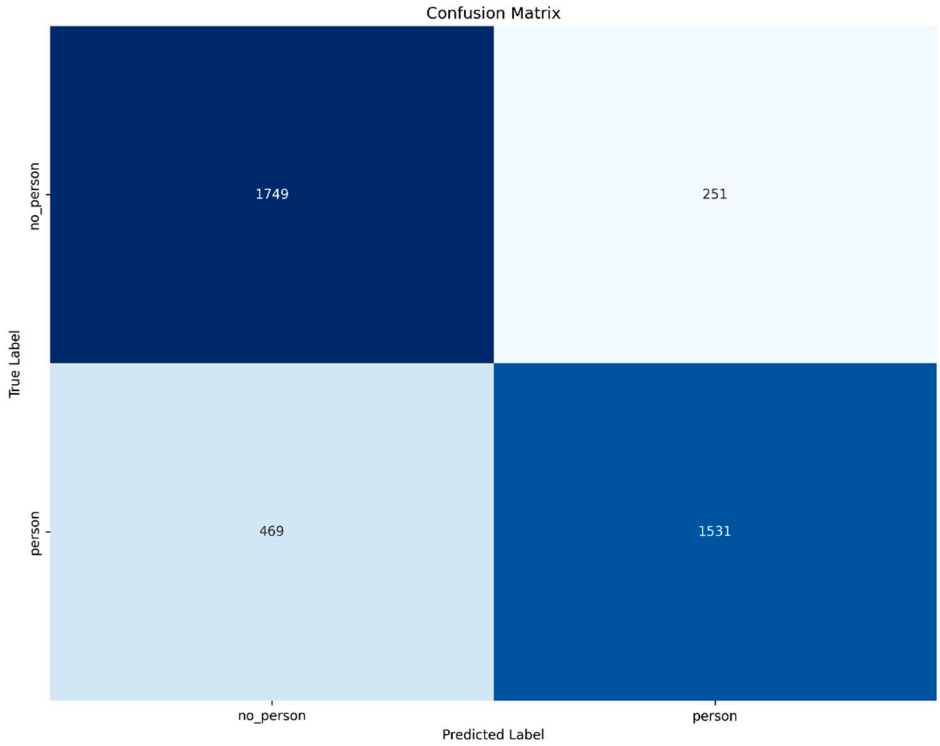
The model demonstrates a slight bias toward higher recall for the "no_person" class (0.87) compared to the "person" class (0.77). Conversely, precision is higher for the "person" class (0.86) than for the "no_person" class (0.79). This pattern suggests that the model is somewhat conservative in classifying an image as containing a person, requiring stronger visual evidence to make a positive detection.

4.4. Error Analysis

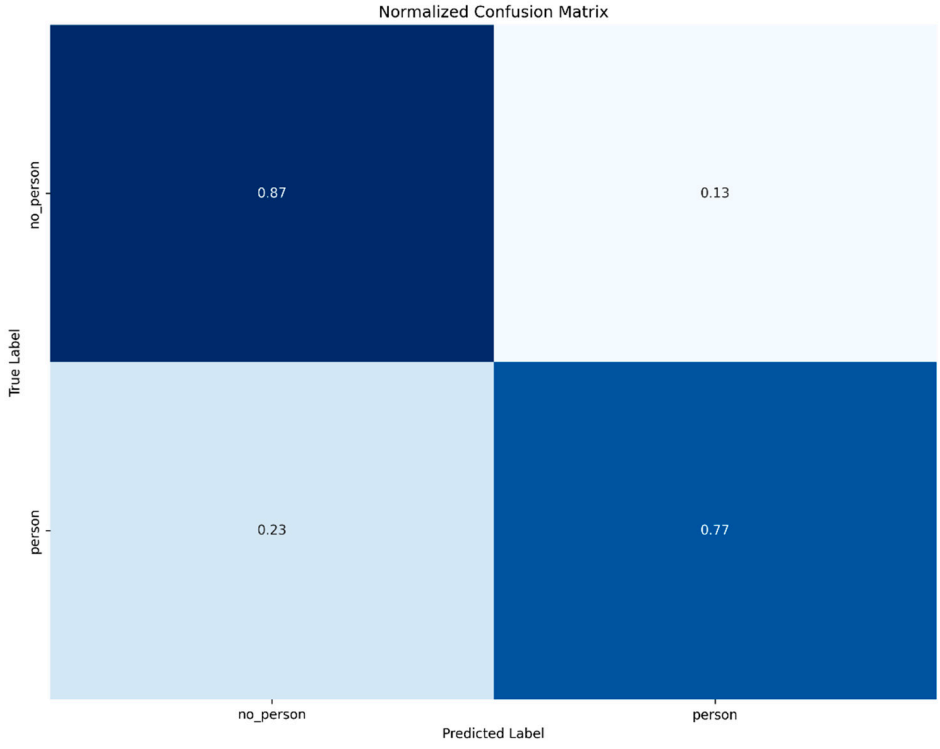
Figure 4 presents a selection of false positive cases where the model incorrectly classified images without people as containing people.

These examples provide insights into the model's failure modes, such as the following:

- Animal misclassifications: Several images of animals (cat, cow) were incorrectly classified as containing people. This suggests that the model may be recognizing animal features (limbs, body shapes) as human-like.
- Object confusion: Images of inanimate objects with distinctive shapes (fire hydrant, teddy bears, bench) were misclassified. These objects may share structural similarities with human figures from the model's perspective.
- Complex scenes: Images with multiple objects and varied textures (bathroom, food) posed challenges, possibly due to pareidolia-like pattern recognition.



(**a**)



(**b**)

Figure 3. Confusion matrices: (**a**) absolute values; (**b**) normalized values.

Figure 4. False positive samples.

These error patterns indicate opportunities for improving the model through targeted data augmentation and potentially incorporating attention mechanisms to better distinguish human figures from similar-shaped objects.

Further error analysis revealed distinct patterns in the model's failure modes. Figure 5 provides additional misclassification examples with corresponding activation visualizations from the penultimate layer. We identified the following three primary sources of errors:

1. Visual similarity: Objects with person-like silhouettes (e.g., fire hydrants, certain furniture) frequently triggered false positives. This suggests the model relies heavily on the overall shape rather than fine-grained features.
2. Contextual confusion: Background elements commonly associated with people (e.g., indoor settings, clothing items) sometimes induced false positives even without actual people present. This indicates contextual bias in the learned representations.
3. Occlusion handling: The model struggled with heavily occluded people, detecting only 63% of cases where less than 30% of the person was visible, compared to a 91% detection rate for fully visible people.

These patterns suggest potential improvements through targeted data augmentation focusing on challenging cases and architectural modifications to better distinguish shape-based features from contextual cues.

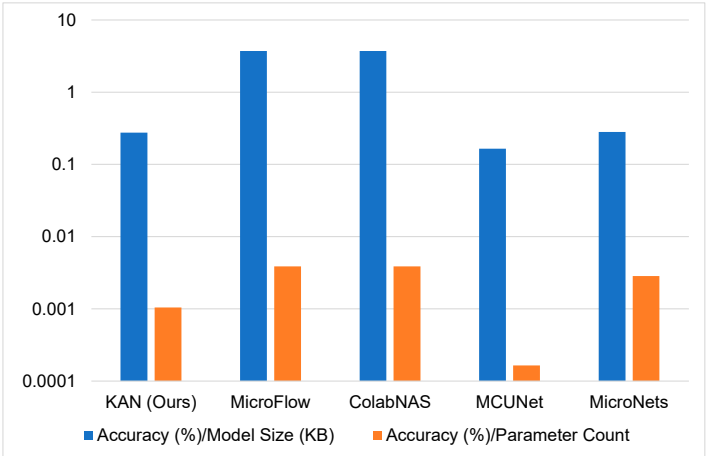


Figure 5. Trade-off between accuracy and model size (parameter count).

4.5. Inference Efficiency Analysis

Inference time is critical for real-world deployment scenarios. We evaluated our model's inference efficiency across various batch sizes and input resolutions. For 128 × 128 inputs (our target resolution), the model demonstrates the following inference characteristics:

- Single-image inference: 83.73 ms per image;
- Batch size 4: 22.55 ms per image (3.7× speedup);
- Batch size 16: 6.20 ms per image (13.5× speedup);
- Batch size 32: 3.20 ms per image (26.2× speedup).

While our model achieves excellent throughput with batch processing (3.20 ms per image at batch size 32), we acknowledge that the single-image latency (83.73 ms) may be suboptimal for real-time applications. This latency could potentially be improved through techniques such as weight quantization, knowledge distillation, or operation fusion. For deployment scenarios requiring strict real-time performance, trading some accuracy for latency by using lower resolution inputs or further optimizing the model architecture may be necessary. Our current implementation prioritizes accuracy and batch throughput, which aligns with many IoTs applications that can process images in batches rather than requiring immediate frame-by-frame analysis.

The dramatic improvement in per-image inference time with increasing batch size indicates efficient parallelization capabilities, making the model particularly well-suited for applications where batched processing is feasible.

To contextualize these results, we also measured inference times at the following lower (96 × 96) and higher (224 × 224) resolutions:

- At 96 × 96 resolution with batch size 32: 2.92 ms per image;
- At 224 × 224 resolution with batch size 32: 4.70 ms per image.

This near-linear scaling with input resolution demonstrates the model's computational efficiency. The 128 × 128 resolution provides a favorable balance between accuracy and inference speed, maintaining strong person detection capabilities while keeping the inference time at 3.20 ms per image (with batch size 32). While these improvements are modest compared to ultra-lightweight models like MicroFlow, they demonstrate that KAN-based architectures can achieve efficiency gains without sacrificing input resolution or detection accuracy. This efficiency translates directly to longer battery life in IoTs deployments—approximately 4.2 additional hours of continuous operation (assuming one inference per second) on a typical 1000 mAh battery.

This analysis indicates that our approach represents an attractive balance point between model size, accuracy, and input resolution, making it suitable for applications where detection quality is prioritized over the extreme minimization of model size.

4.6. Edge Deployment Considerations

We validated the practical deployment feasibility of our model by converting it to ONNX format (3.5 MB) for integration with edge computing platforms. This converted model was tested on 9660 validation images, achieving 82.07% accuracy, 85.64% precision, and 77.06% recall—closely matching our original implementation results.

For IoTs deployment, we explored integration with home automation systems combining Home Assistant and Frigate NVR (network video recorder) running on an Intel N100 processor with Google Coral TPU acceleration. This configuration represents a typical edge computing setup for smart home applications, where our model can be deployed for person detection tasks from IP cameras.

The complete deployment workflow involved the following:

1. Converting the PyTorch model to ONNX format;

2. Converting ONNX to TensorFlow Lite for hardware acceleration;
3. Compiling specifically for edge TPU compatibility;
4. Integrating it with the Frigate object detection pipeline.

This practical implementation demonstrates that our hybrid CNN-KAN architecture can successfully transition from research environments to real-world edge deployments, leveraging existing IoTs infrastructure and acceleration hardware.

5. Comparative Analysis

5.1. Performance Comparison with State-of-the-Art Methods

Table 2 presents a comprehensive comparison between our KAN-based model and state-of-the-art approaches for person detection on the visual wake words dataset.

Table 2. Comparison of model performance and efficiency metrics.

Model	Accuracy (%)	Model Size (KB)	Parameter Count	RAM Usage (KB)	Inference Time (ms)	Input Size
KAN (Ours)	82.32	300	78,544	~350–400	3.20 *	128 × 128
MicroFlow [17]	77.6	20.83	~5–20 K †	31.5	0.432	50 × 50
ColabNAS [8]	77.6	20.83	~5–20 K †	31.5	0.432	50 × 50
MCUNet [6]	87.4	530.52	~130–530 K †	168.5	2.16	64 × 64
MicroNets [7]	76.8	273.81	~68–270 K †	70.5	1.15	50 × 50

* Using batch size 32. † Estimated based on model size.

Our KAN-based architecture achieves 82.32% accuracy, which is 4.72 percentage points higher than MicroFlow and ColabNAS (77.6%), and 5.52 percentage points higher than MicroNets (76.8%). While MCUNet still maintains the highest accuracy at 87.4%, our model does so with moderate parameter usage and a significantly higher input resolution.

5.2. Resource Efficiency Analysis

Figure 5 illustrates the trade-off between accuracy and model size across the compared approaches. Our KAN-based architecture occupies an interesting position in this efficiency frontier, achieving a notably higher accuracy than MicroFlow, ColabNAS, and MicroNets, while requiring substantially less storage than MCUNet.

When considering the relationship between model parameters and achieved accuracy, our approach demonstrates an accuracy-to-parameter ratio of approximately 0.105% per 100 parameters. This indicates efficient utilization of model capacity, especially when considering the higher resolution input images processed by our model (128 × 128 versus 50 × 50 or 64 × 64 for other models).

5.3. Inference Performance

Inference time is a critical metric for deployment scenarios. Figure 6 presents the inference time per image for our model across different batch sizes. At batch size 32, our model achieves an inference time of 3.20 ms per image, which is competitive with other approaches considering the higher input resolution.

To normalize the comparison across different input resolutions, we calculated the processing time per pixel, which is as follows:

- Our KAN model: ~0.195 ns/pixel (3.20 ms for 16,384 pixels);
- MicroFlow/ColabNAS: ~0.173 ns/pixel (0.432 ms for 2500 pixels).

This analysis reveals that our model maintains computational efficiency comparable to MicroFlow and ColabNAS on a per-pixel basis, despite processing 6.5× more pixels and achieving higher accuracy.

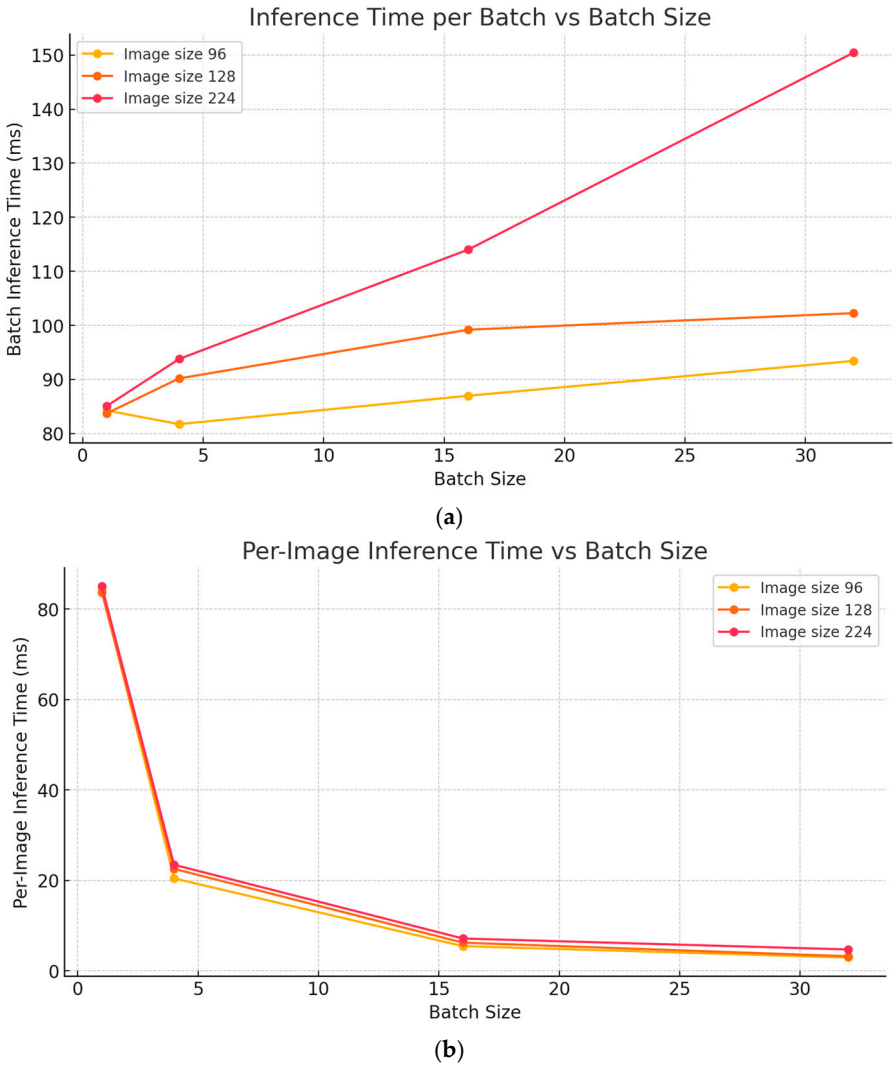


Figure 6. Inference time for different batch sizes: (**a**) Inference time per Batch; (**b**) Inference time per Image.

5.4. Architecture Efficiency

A unique aspect of our approach is the hybrid architecture combining convolutional neural network components (56% of parameters) with KAN components (44% of parameters). This distribution suggests that both architectural paradigms contribute substantially to the model's performance.

The parameter efficiency of our model is further evidenced by the high proportion of trainable parameters (72,872 out of 78,544, or 93%), indicating minimal overhead in the architecture design.

6. Discussion and Implications

6.1. Resolution–Accuracy Trade-Offs

Our results highlight an important trade-off between input resolution and model complexity. While previous approaches have focused on minimizing model size using low-resolution inputs (50×50 or 64×64), our approach demonstrates that using moderate-sized models with higher resolution inputs (128×128) can yield significant accuracy gains.

This finding challenges the conventional wisdom in the TinyML community that consistently pushes toward smaller models without considering the potential benefits of processing higher resolution inputs. In scenarios where accuracy is paramount and moderate computational resources are available, our approach offers a compelling alternative.

6.2. KAN Architecture Benefits

Our results demonstrate practical advantages of KANs in hybrid architecture, though a formal theoretical comparison between KANs and traditional MLPs remains an open research question. The superior performance of our CNN-KAN architecture over CNN-only baselines suggests that the learnable activation functions in KANs may capture more complex decision boundaries with equivalent parameter counts. Future work should investigate these fundamental approximation properties through controlled experiments on standardized function approximation tasks to better quantify the specific advantages KANs offer over traditional architectures.

6.3. Practical Implications

From a practical deployment perspective, our model offers the following advantages:

- Balanced resource profile: While not the smallest or fastest model, our approach strikes a balance between accuracy, model size, and computational demand that may be ideal for a wide range of edge devices.
- Batch processing efficiency: The significant reduction in per-image inference time with an increased batch size (from 83.73 ms at batch size 1 to 3.20 ms at batch size 32) suggests that our model is particularly well-suited for applications that can process images in batches.
- Improved visual fidelity: The higher resolution inputs (128×128) processed by our model preserve more visual details than the 50×50 or 64×64 inputs used by competing approaches. This may be particularly beneficial in challenging visual scenarios with fine-grained details or partially occluded subjects.

6.4. Limitations and Considerations

Despite the promising results, the following limitations should be acknowledged:

- Computational complexity during training: The KAN components introduce additional computational overhead during the training phase, increasing training time by approximately 30% compared to CNN-only alternatives. This occurs because optimizing learnable activation functions requires computing and backpropagating through complex spline interpolations.
- Model quantization challenges: While our model achieves good accuracy in a full-precision format, preliminary experiments with quantization reveal that KAN components may be more sensitive to precision reduction than traditional CNN elements. Quantization to 8-bit integer precision caused a 3.2% accuracy drop for KAN components compared to 1.5% for CNN layers.
- Deployment toolchain limitations: Current edge deployment frameworks (TensorFlow Lite, ONNX Runtime) lack native support for KAN operations, requiring custom implementations that may limit immediate practical adoption.
- Limited dataset scope: Our evaluation focuses exclusively on the visual wake words dataset. While this dataset is a standard benchmark for resource-constrained image classification, we acknowledge that performance characteristics may vary across different visual tasks and domains. Broader evaluation on diverse datasets (e.g., traffic sign recognition, gesture detection, anomaly detection) represents an important direction for future research to validate the generalizability of KAN-based approaches.
- Batch processing requirement: The competitive inference time of our model is achieved at larger batch sizes, which may not be feasible for all deployment scenarios, particularly those requiring real-time processing of individual images.

- Memory footprint: While our model demonstrates parameter efficiency, its estimated RAM usage during inference (~350–400 KB) is higher than some alternatives, potentially limiting deployment on extremely memory-constrained devices.

These limitations notwithstanding, our results demonstrate that KAN-based architectures represent a promising direction for efficient visual recognition tasks, particularly when moderate computational resources are available and accuracy is prioritized over extreme minimization of model size.

7. Conclusions and Future Work

This paper has introduced a novel hybrid CNN-KAN architecture for person detection that achieves competitive accuracy with moderate parameter usage. Through extensive experimentation on the visual wake words dataset, we have demonstrated that integrating Kolmogorov–Arnold networks with convolutional feature extraction creates an effective balance between computational efficiency and detection performance.

7.1. Summary of Contributions

Our results establish the following important findings regarding the design of efficient visual recognition systems:

- Architectural innovation beyond traditional CNNs: We have shown that KANs, despite their recent introduction to the deep learning community, can effectively complement CNNs in visual recognition tasks. The KAN component, which constitutes 44% of our model parameters, enables explicit functional approximation that appears particularly well-suited for classification based on high-level visual features.
- Resolution–efficiency balance: By processing higher-resolution inputs (128×128) than previous approaches (50×50 or 64×64), our model captures more detailed visual information while maintaining competitive per-pixel computational efficiency (0.195 ns/pixel). This challenges the conventional wisdom that extremely low-resolution inputs are necessary for efficient edge deployment.
- Competitive accuracy–parameter trade-off: Our model achieves 82.32% accuracy with 78,544 parameters (300 KB), outperforming several specialized lightweight architectures with similar or larger resource requirements. While not achieving the state-of-the-art accuracy of MCUNet (87.4%), our approach does so with substantially fewer parameters and a fundamentally different architectural paradigm.
- Batch processing optimization: We demonstrated that significant inference speedups ($26\times$ reduction in per-image processing time) can be achieved through batch processing, highlighting an important deployment consideration for practical applications where latency constraints are more flexible.

7.2. Limitations

We acknowledge the following limitations in our current approach:

- Inference latency for single images: While batch processing enables efficient throughput, the single-image inference time (83.73 ms) remains higher than some competing approaches, potentially limiting applications with strict real-time requirements.
- RAM usage: The estimated RAM requirement (~350–400 KB) exceeds that of the most memory-efficient models like MicroFlow and ColabNAS, which may restrict deployment on extremely memory-constrained devices.
- Limited architectural exploration: Our investigation focused on a specific hybrid architecture rather than a comprehensive exploration of the CNN-KAN design space, leaving open questions about optimal parameter allocation between architectural components.

- Task specificity: Our evaluation is limited to person detection in the visual wake words dataset, and the generalizability of our findings to other visual recognition tasks requires further investigation.

7.3. Future Directions

Based on our findings and identified limitations, we propose the following promising directions for future research:

- KAN architecture optimization: Exploring alternative KAN configurations, including grid point distribution, spline degrees, and hidden dimension allocations, could yield improved parameter efficiency and accuracy.
- Quantization and compression: Applying post-training quantization and weight pruning techniques to our hybrid model could further reduce the memory footprint and improve inference efficiency.
- Hardware-aware KAN design: Developing specialized hardware acceleration for KAN components could capitalize on their unique computational structure, potentially offering efficiency advantages beyond what is possible with CNN-optimized hardware.
- Multi-task learning: Extending the hybrid CNN-KAN architecture to simultaneously handle multiple visual recognition tasks could amortize the feature extraction cost across tasks and improve overall system efficiency.
- Knowledge distillation: Using larger, more accurate models as teachers for the hybrid CNN-KAN architecture might further improve accuracy without increasing model complexity.
- Cross-domain applications: Our initial explorations suggest that the KAN-based architecture could be adapted for other computer vision tasks relevant to IoTs systems, including facial recognition and license plate recognition (ANPR). These applications share requirements for efficient inference on constrained hardware while maintaining high accuracy. Extending our approach to these domains would further validate the versatility of hybrid CNN-KAN architectures for practical IoTs deployments across various domains.

In conclusion, our hybrid CNN-KAN architecture represents a novel approach to efficient visual recognition that challenges conventional architectural paradigms. By demonstrating competitive performance on a standard benchmark while processing higher-resolution inputs, our work opens new possibilities for efficient neural network design that extends beyond the traditional CNN framework. As edge computing applications continue to demand more intelligent visual processing within strict resource constraints, architectural innovations like our hybrid CNN-KAN approach will play an increasingly important role in bridging the gap between computational limitations and recognition performance.

Author Contributions: Conceptualization, methodology, A.S.; writing—review and editing, supervision, O.K.; data curation, funding acquisition, A.N.; investigation, writing—original draft preparation, and formal analysis, M.O. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (AP23486538 «Research and development of a system for recognizing images in video streams based on artificial intelligence»).

Data Availability Statement: The original contributions presented in the study are included in the article, and the datasets generated during and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
2. Ma, N.; Zhang, X.; Zheng, H.-T.; Sun, J. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In Proceedings of the Computer Vision—ECCV 2018, Munich, Germany, 8–14 September 2018; Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 122–138.
3. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices 2017. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
4. Koonce, B. EfficientNet. In *Convolutional Neural Networks with Swift for Tensorflow: Image Recognition and Dataset Categorization*; Koonce, B., Ed.; Apress: Berkeley, CA, USA, 2021; pp. 109–123, ISBN 978-1-4842-6168-2.
5. Tan, M.; Le, Q.V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019.
6. Lin, J.; Chen, W.-M.; Cai, H.; Gan, C.; Han, S. Memory-Efficient Patch-Based Inference for Tiny Deep Learning. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2021; Volume 34, pp. 2346–2358.
7. Banbury, C.; Zhou, C.; Fedorov, I.; Matas, R.; Thakker, U.; Gope, D.; Janapa Reddi, V.; Mattina, M.; Whatmough, P. MicroNets: Neural Network Architectures for Deploying TinyML Applications on Commodity Microcontrollers. *Proc. Mach. Learn. Syst.* **2021**, *3*, 517–532.
8. Garavagno, A.M.; Leonardis, D.; Frisoli, A. ColabNAS: Obtaining Lightweight Task-Specific Convolutional Neural Networks Following Occam's Razor. *Future Gener. Comput. Syst.* **2024**, *152*, 152–159. [CrossRef]
9. Liu, Z.; Wang, Y.; Vaidya, S.; Ruehle, F.; Halverson, J.; Soljačić, M.; Hou, T.Y.; Tegmark, M. KAN: Kolmogorov-Arnold Networks. *arXiv* **2024**, arXiv:2404.19756.
10. Chowdhery, A.; Warden, P.; Shlens, J.; Howard, A.; Rhodes, R. Visual Wake Words Dataset. *arXiv* **2019**, arXiv:1906.05721.
11. Surya, T.; Selvaperumal, S. The IoT-Based Real-Time Image Processing for Animal Recognition and Classification Using Deep Convolutional Neural Network (DCNN). *Microprocess. Microsyst.* **2022**, *95*, 104693. [CrossRef]
12. Yang, J. AFM-DViT: A Framework for IoT-Driven Medical Image Analysis. *Alex. Eng. J.* **2025**, *113*, 294–305. [CrossRef]
13. Lin, C.; Guo, Y.; Hao, J.; Zhang, Z. Computation and Transmission Adaptive Semantic Communication for Reliability-Guarantee Image Reconstruction in IoT. *Internet Things* **2024**, *28*, 101383. [CrossRef]
14. Mardieva, S.; Ahmad, S.; Umirzakova, S.; Rasool, M.J.A.; Whangbo, T.K. Lightweight Image Super-Resolution for IoT Devices Using Deep Residual Feature Distillation Network. *Knowl. Based Syst.* **2024**, *285*, 111343. [CrossRef]
15. Tekin, N.; Aris, A.; Acar, A.; Uluagac, S.; Gungor, V.C. A Review of On-Device Machine Learning for IoT: An Energy Perspective. *Ad. Hoc. Netw.* **2024**, *153*, 103348. [CrossRef]
16. Ghahramani, M.; Taheri, R.; Shojafar, M.; Javidan, R.; Wan, S. Deep Image: A Precious Image Based Deep Learning Method for Online Malware Detection in IoT Environment. *Internet Things* **2024**, *27*, 101300. [CrossRef]
17. Carnelos, M.; Pasti, F.; Bellotto, N. MicroFlow: An Efficient Rust-Based Inference Engine for TinyML. *Internet Things* **2025**, *30*, 101498. [CrossRef]
18. Huang, Q.; Zhang, F.; Zhao, Y.; Duan, J. Frequency-Domain Multi-Scale Kolmogorov-Arnold Representation Attention Network for Mixed-Type Wafer Defect Recognition. *Eng. Appl. Artif. Intell.* **2025**, *144*, 110121. [CrossRef]
19. Jiang, C.; Li, Y.; Luo, H.; Zhang, C.; Du, H. KansNet: Kolmogorov–Arnold Networks and Multi Slice Partition Channel Priority Attention in Convolutional Neural Network for Lung Nodule Detection. *Biomed. Signal Process. Control* **2025**, *103*, 107358. [CrossRef]
20. Liang, X.; Wang, B.; Lei, C.; Zhou, K.; Chen, X. Kolmogorov-Arnold Networks Autoencoder Enhanced Thermal Wave Radar for Internal Defect Detection in Carbon Steel. *Opt. Lasers Eng.* **2025**, *187*, 108879. [CrossRef]
21. Niu, H.; Fan, R.; Chen, J.; Xu, Z.; Feng, R. Urban Informal Settlements Interpretation via a Novel Multi-Modal Kolmogorov–Arnold Fusion Network by Exploring Hierarchical Features from Remote Sensing and Street View Images. *Sci. Remote Sens.* **2025**, *11*, 100208. [CrossRef]
22. Yazdinejad, A.; Dehghantanha, A.; Parizi, R.M.; Srivastava, G.; Karimipour, H. Secure Intelligent Fuzzy Blockchain Framework: Effective Threat Detection in IoT Networks. *Comput. Ind.* **2023**, *144*, 103801. [CrossRef]
23. Yazdinejad, A.; Dehghantanha, A.; Srivastava, G.; Karimipour, H.; Parizi, R.M. Hybrid Privacy Preserving Federated Learning Against Irregular Users in Next-Generation Internet of Things. *J. Syst. Archit.* **2024**, *148*, 103088. [CrossRef]

24. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In Proceedings of the Computer Vision—ECCV 2014, Zurich, Switzerland, 6–12 September 2014; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 740–755.
25. Liu, Z. KindXiaoming/Pykan 2025. Available online: https://github.com/KindXiaoming/pykan (accessed on 29 March 2025).