engineering proceedings

Proceeding Paper

Embedded Intelligence for Smart Home Using TinyML Approache to Keyword Spotting [†]

Jyoti Mishra [1], Timothy Malche [2,*] and Amit Hirawat [2]

[1] Manipal University Jaipur, Jaipur 303007, Rajasthan, India; jyoti.191018608@muj.manipal.edu
[2] Department of Computer Applications, Manipal University Jaipur, Jaipur 303007, Rajasthan, India; amit.hirawat@jaipur.manipal.edu
[*] Correspondence: timothy.malche@jaipur.manipal.edu;

[†] Presented at The 11th International Electronic Conference on Sensors and Applications (ECSA-11), 26–28 November 2024; Available online: https://sciforum.net/event/ecsa-11

Abstract: Current research in home automation focuses on integrating emerging technologies like Internet of Things (IoT), and machine learning to create smart home solutions that offer enhanced convenience, efficiency, and security. Benefits include remote control of household devices, optimized energy usage through automated systems, and improved user experience with real-time monitoring and alerts. In this study a TinyML (Tiny Machine Learning) based keyword spotting machine learning model and system is proposed which enables voice-based home automation. The proposed system allows users to control household devices through voice commands with minimal computational resources and real-time performance. The main objective of this research is to develop TinyML model for resource constrained devices. The system enables home systems to efficiently recognize specific keywords or phrases by integrating voice control for enhanced user convenience and accessibility. In this research the different voice keywords of users of different age groups have been collected in home environment and trained using machine learning algorithms. An IoT based system is then developed utilizing the TinyML model to recognize specific voice command and perform home automation tasks. The model has achieved 98% accuracy with F1 score of 1.00 and 92% recall. The quantized model uses Latency of 5 ms, 7.9K of RAM and 43.7K of flash for keyword classification which is the best fit for any resource constraint devices. The proposed system demonstrates the viability of deploying a keyword spotting model for home automation on resource-constrained IoT devices. The research helps in building efficient and user-friendly smart home solutions, enhancing the accessibility and functionality of home automation systems.

Keywords: embedded AI; TinyML; home automation; keyword spotting; Internet of Things

Citation: Mishra, J.; Malche, T.; Hirawat, A. Embedded Intelligence for Smart Home Using TinyML Approache to Keyword Spotting. *Eng. Proc.* **2024**, *6*, x. https://doi.org/10.3390/xxxxx

Academic Editor(s): Name

Published: 26 November 2024

1. Introduction

TinyML is a technology that brings intelligence to small, embedded devices. In smart homes, TinyML-powered appliances can operate more efficiently and autonomously. These devices can make smart decisions on their own, without needing constant connection to the internet. This improves privacy, speeds up responses, and provides more real-time experience.

Keyword Spotting (KWS) is a crucial technology in smart home automation. It allows devices to identify specific voice commands, enabling users to interact with their homes using natural language. TinyML is a powerful approach that makes KWS possible on small, embedded devices. By incorporating TinyML, smart home devices can recognize and respond to voice commands locally, without needing to send data to the cloud. This not only improves privacy but also reduces reliance on internet connectivity.

In essence, KWS with TinyML empowers smart home devices to understand and react to spoken instructions, providing a more intuitive and convenient user experience.

Traditional smart home automation systems rely on centralized cloud-based processing systems (e.g., Amozon Echo, Alexa and Google home) [1] which can introduce latency, privacy concerns and dependency on continuous internet connections. To address these limitations, researchers are exploring the concept of embedded intelligence using TinyML [2] which involves placing machine learning capabilities directly on the home appliances. This enables making decisions locally on the device, leading to faster response times and allowing devices to function without an internet connection. TinyML models are lightweight and designed for resource-constrained devices like microcontrollers. This allows to deploy intelligent functionalities without compromising on performance calibration. Overall, embedded intelligence powered by TinyML offers a promising future for smart homes. It provides faster, more private, and reliable user experience compared to traditional cloud-based systems.

Despite the potential benefits of using TinyML for KWS in smart home automation, there are challenges to overcome such as:

- Optimized Model Size: TinyML models need to be compact to fit on resource-constrained devices. This can sometimes lead to trade-offs in accuracy.
- Robustness: Ensuring that KWS models are accurate in noisy environments and don't trigger false positives is crucial for a seamless user experience.

This research focused on developing a TinyML based KWS system for smart home automation. To achieve this, the following steps were taken:

- Data Collection: Real-time voice data was gathered from various Indian families, representing different genders, ages, and home environments. This data was used to train the keyword spotting model.
- Model Development: A lightweight TinyML model was created. This model is designed for efficient processing on resource-constrained devices.
- Model Quantization: The model was optimized to reduce its size and computational requirements, making it suitable for deployment on IoT devices.

By following these steps, the research aimed to create a user-friendly smart home solution capable of accurately recognizing voice commands.

2. Related Work

Steven Guamán et al. [1] developed voice-controlled home automation using Amazon Echo, Alexa and Google Home. This system was based on cloud computing using more data and more memory. S Somesh et al. [3] developed real time smart home automation system by using Alexa, Amazon Echo dot and ESP2866 Node MCU.

Tomi Kinnunen et al. [4] presented voice activity detection using MFCC feature extraction and SVM. A deep neural network-based keyword spotting system is developed by Guoguo Chen et al. [5]. Pete Warden et al. [6] described necessary requirements of data collection, its properties and previous version of the data.

Urvi Singh et al. [7] presented IoT based smart home automation system. ESP8266 Wi-Fi technology and Blynk app had been used in this system to switch on\off all the home appliances while Danyar N Karim et al. [8] designed a multilingual keyword spotting system which was recognizing emergency keyword "help" in different four languages such as English, Arabic, Kurdish and Malay Languages. Deep convolutional spiking neural network-based keyword spotting system was presented by Emre Yilmaz et al. [9].

William Hartmann et al. [10] presented a systematic comparative study to spot keywords which was used for techniques of multiple system combination. Maria t. Nyamukuru et al. [11] proposed GRU architecture for tiny eats. It was implemented on Arm Cortex M0+ for detecting number of tiny meals taken.

Hilmat Yar et al. [12] presented a cost effective integrated smart home automation system on IoT and concept of edge computing. Akshata Kamble et al. [13] developed a smart home using Raspberry Pi which was based on Google Assistant. Andreas Kamilaris et al. [14] developed a web application for controlling smart home. A similar approach was presented by Chinmay Bepery et al. [15]. Vaishnavi S. Gunge et al. [16] presented comparison of various home automation systems which were based on web systems, e-mail, blue-tooth, sms, android, dual-tone multi-frequency, Zig-bee, cloud and Internet.

Dr. E. Chandra et al. [17] presented the concept of keyword spotting, its types, its processes, various applications and different approaches used for implementation of keyword spotting. Sumedha Rai et al. [18] developed a keyword spotting system by using MFCC, Hidden Markov Model (HMM) with Gaussian Mixture, CNN and variants of RNN including LSTM. They achieved 93.9% accuracy by using RNN with BiLSTM. For keyword spotting system Takuya Higuchi et al. [19] proposed a stacked 1D convolutional network (S1DCNN).

3. Materials and Methods

3.1. Dataset

We use the real time voice datasets of 10 Indian family members of different ages and genders in English language for examining experiments on neural network architecture. Total duration of voice dataset is 4 h and 58 s. After cleaning and processing the data, we obtained a dataset of 14,521 audio samples, each lasting 1 s. These samples included 14 different keywords spoken by 10 members of an Indian family, representing a variety of ages and genders. Each voice sample in the dataset contains only one keyword. The neural network model was trained to identify specific keywords from a list of 14 options, including “hello”, “hi”, “welcome”, “yes”, “no”, “on”, “off”, “start”, “stop”, “wake”, “sleep”, “open”, “close” and “silence” (when no words were spoken).

The entire dataset was split into training and testing sets in an 80:20 ratio and the audio clips from the same family member were kept together within the same set. The training set contained 11,632 audio clips, while the testing set had 2889. Figure 1 illustrates the frequency distribution of 1-s samples for the keyword “close” spoken by a single family member.

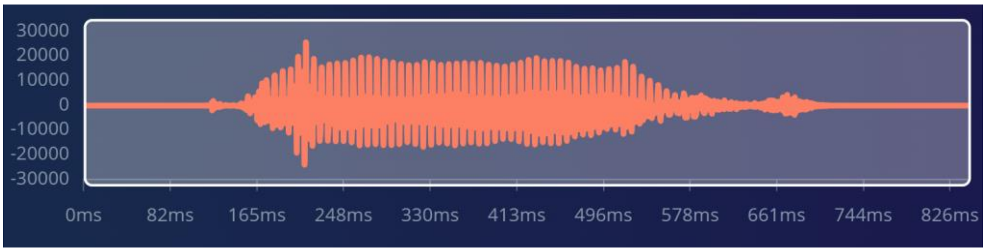
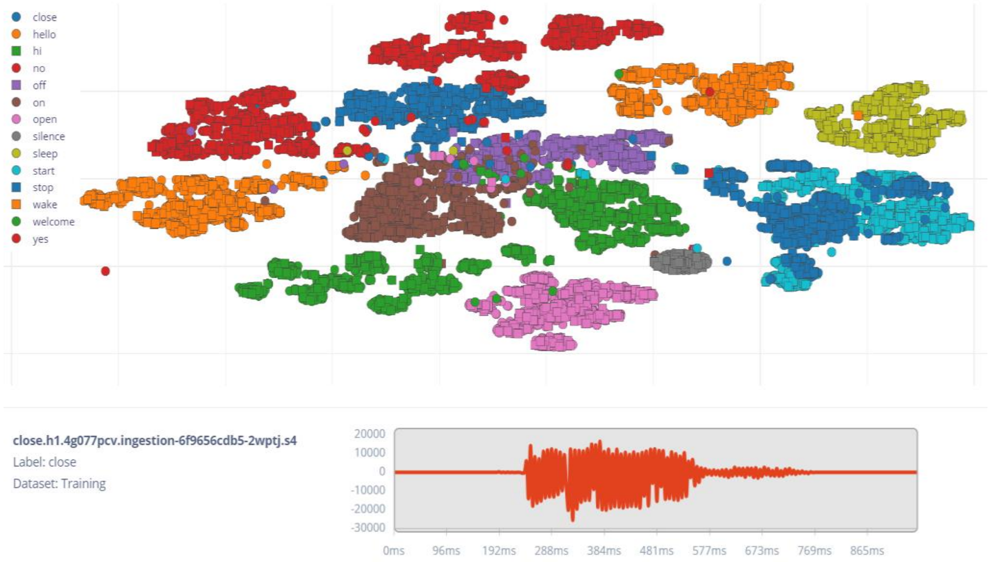


Figure 1. Raw Data representing ‘close’ keyword.

The following Figure 2 shows data for all 14 class labels (i.e., “close”, “hello”, “hii”, “welcome”, “yes”, “on”, “off”, “start”, “stop”, “no”, “wake”, “sleep”, “open” and “silence”).



close.h1.4g077pcv.ingestion-6f9656cdb5-2wptj.s4

Label: close

Dataset: Training

Figure 2. Audio data for 14 classes of keywords.

The voice data was collected using a 1-s window size and a 16,000 Hz sampling rate. To extract features from the audio data, the MFCC method was applied. The parameters (i.e., 13 coefficients, a 0.05-s frame length, a 0.025-s frame stride, 32 filters, a 256-point FFT, a 101-point normalization window, a low frequency of 0, a high frequency of the cepstral coefficient (0.98), and a single shift) were used. Using these settings, 11,632 training windows were generated, totaling 3 h, 13 min, and 2 s of data across 14 classes. During this processing, the device utilized an average of 80 milliseconds of processing time and 13 kilobytes of peak RAM.

3.2. Methodology

The KWS system is designed to detect various keywords, enabling control of IoT devices through voice commands. To build the proposed system, the data using a microphone from individuals of different genders and age groups speaking English was collected. This data is then preprocessed by resampling at 1 ms intervals and accurately labeled into 14 classes before training. Mel Frequency Cepstral Coefficient (MFCC) used for feature extraction from the voice dataset. This research utilizes a Convolutional Neural Network (CNN) model and a C++ library for keyword spotting, which is deployed on a Cortex-M7 216 MHz processor.

Figure 3 shows the overall workflow of the keyword spotting system for smart home automation.

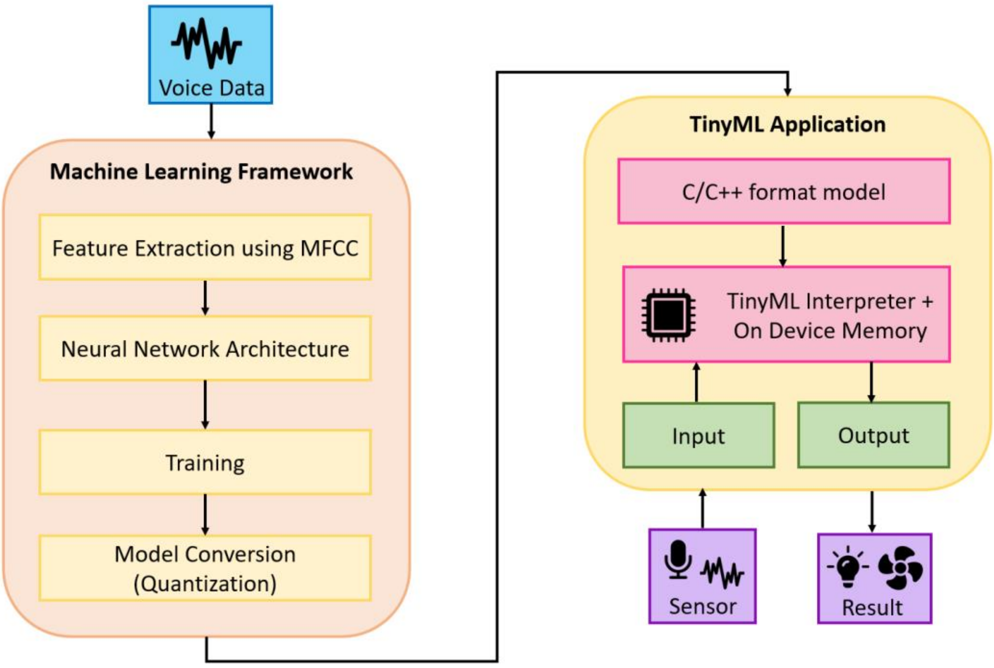

Voice Data
Machine Learning Framework
Feature Extraction using MFCC
Neural Network Architecture
Training
Model Conversion (Quantization)
TinyML Application
C/C++ format model
TinyML Interpreter + On Device Memory
Input
Output
Sensor
Result

Figure 3. Workflow of KWS.

3.3. Neural Network Architecture

The neural network model was trained using 100 training epochs with a learning rate of 0.005. To evaluate the model's performance during training, 20% of the dataset was set aside for validation, and a batch size of 32 was used.

The network architecture (as shown in Figure 4) designed for this study takes a 1 × 39 × 13 input. The first convolutional layer applies 16 filters, each with a size of 1 × 3 × 13. This layer extracts local features from the input image. A ReLU activation function is applied to introduce non-linearity. A max pooling layer downsamples the output of the convolutional layer to reduce dimensionality and computational cost. The shape is reduced to 1 × 20 × 1 × 16. The second convolutional layer applies 32 filters, each with a size of 1 × 3 × 16. This layer extracts more complex features from the previous layer's output. Another ReLU activation function is applied. The output of the final convolutional layer is flattened into a one-dimensional vector of size 1 × 320. A fully connected layer with 14 neurons is used to map the flattened features to the output classes. The softmax layer applies the softmax activation function to normalize the output probabilities, ensuring that they sum to 1. The final output is a 1 × 14 vector, representing the predicted probabilities for each of the 14 classes.

After training the keyword spotting model using the described neural network architecture, it achieved 99.1% accuracy with a minimal loss of 0.03%. When tested on the sampled voice data, the model demonstrated strong performance across all 14 target keywords.

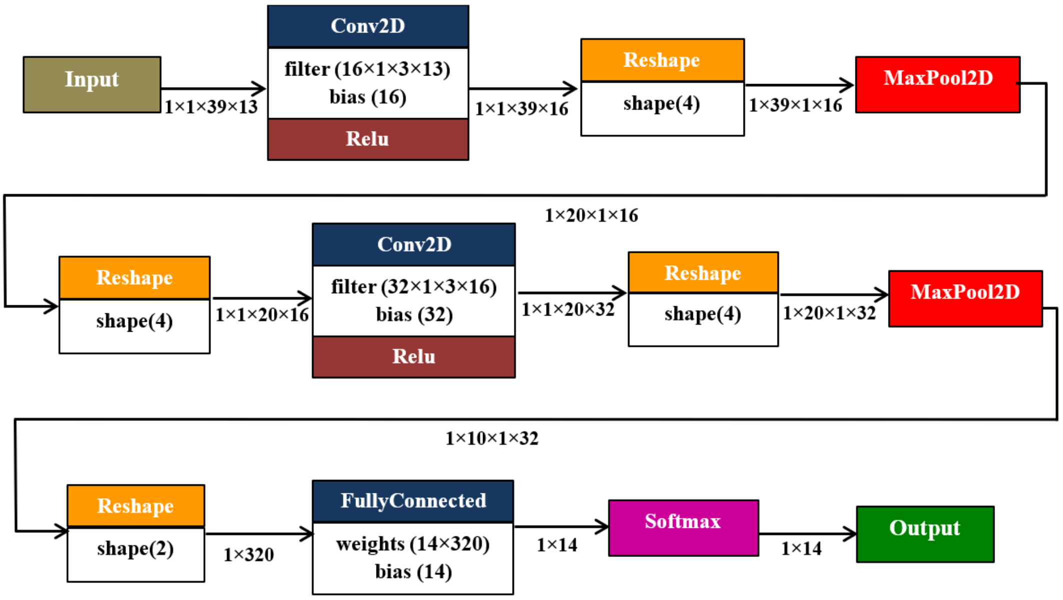


Figure 4. Neural Network Architecture for KWS.

4. Results & Discussion

To test the proposed model on-device, experiments were conducted by deploying the KWS model on the target device. The following hardware components were used for testing the on-device performance of the proposed KWS model. The following image shows the microcontroller board used for the experiment.

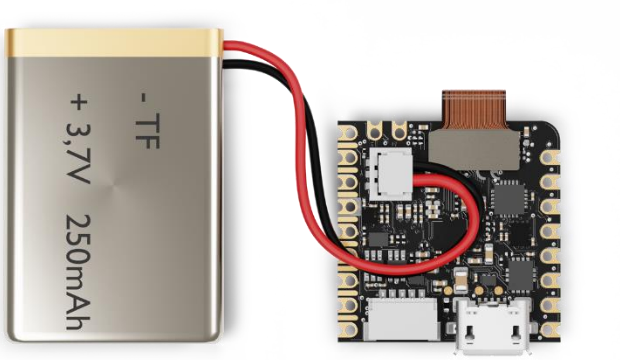


Figure 5. Arduino Nicla Vision.

Arduino Nicla Vision [20] consists of dual-core STM32H747 (Cortex-M7 at 480 MHz and Cortex-M4 at 240 MHz) processor, 2 MP Camera, 6-Axis IMU (LSM6DSOX), and Microphone (MP34DT05). The hardware is compact and can be easily integrated into various home appliances. Its built-in microphone makes it well-suited for this keyword spotting experiment.

Table 1 represents the performance of a keyword spotting model on a training dataset of 14 keywords. The model achieved an overall F1 score of 0.99, indicating strong performance. The highest accuracy was observed for the keywords “Silence” and “Wake” at 100%. The lowest accuracy was observed for the keyword “Hi” at 0.6%. Overall, the model demonstrated high accuracy and precision in recognizing the target keywords.

Table 1. Confusion Matrix (Training dataset).

	Close	**Hello**	**Hi**	**No**	**Off**	**On**	**Open**	**Silence**	**Sleep**	**Start**	**Stop**	**Wake**	**Welcome**	**Yes**
Close	**99.4%**	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0.6%

Hello	0%	**98.7%**	0%	0.9%	0%	0%	0.4%	0%	0%	0%	0%	0%	0%	0%
Hi	0%	0.6%	**97.8%**	0%	0.6%	0%	0.6%	0%	0%	0.6%	0%	0%	0%	0%
No	0.6%	0%	0%	**98.9%**	0%	0%	0%	0%	0%	0%	0%	0%	0.6%	0%
Off	0%	0%	0%	0%	**99.4%**	0.6%	0%	0%	0%	0%	0%	0%	0%	0%
On	0%	0%	0.6%	0%	1.7%	**97.8%**	0%	0%	0%	0%	0%	0%	0%	0%
Open	0%	0%	0%	0%	0%	0%	**100%**	0%	0%	0%	0%	0%	0%	0%
Silence	0%	0%	0%	0%	0%	0%	0%	**100%**	0%	0%	0%	0%	0%	0%
Sleep	0%	0%	0%	0%	0%	0%	0%	0%	**99.4%**	0%	0%	0.6%	0%	0%
Start	0%	0%	1.2%	0%	0%	0%	0%	0%	0%	**98.8%**	0%	0%	0%	0%
Stop	0.5%	0%	0%	0%	0%	0%	0%	0%	0%	0%	**99.5%**	0%	0%	0%
Wake	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	**100%**	0%	0%
Welcome	0%	0.6%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	**99.4%**	0%
Yes	0.6%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	**99.4%**
F1 Score	0.99	0.99	0.98	0.99	0.99	0.99	0.99	1.00	1.00	0.99	1.00	1.00	0.99	0.99

Table 2 presents the real-time classification results of a KWS model on a test dataset. Each row represents a time stamp, and each column corresponds to a specific keyword. The values in the table represent the probability assigned by the model to each keyword at a given time.

Table 2. Live Classification.

Timestamp	Close	Hello	Hi	No	Off	On	Open	Silence	Sleep	Start	Stop	Wake	Welcome	Yes
0	0	1.00	0	0	0	0	0	0	0	0	0	0	0	0
1000	0	0	0	0	0	0	0	0	0	0	0	0	1.00	0
2000	0	0	1.00	0	0	0	0	0	0	0	0	0	0	0
3000	0	0	0	1.00	0	0	0	0	0	0	0	0	0	0
4000	0	0	0	0	0	0	0	0	0	0	1.00	0	0	0
5000	0	0	0	0	0	0	0	0	0	0	0	1.00	0	0
6000	0	0	0	1.00	0	0	0	0	0	0	0	0	0	0
7000	0	0	0	0.95	0.03	0	0	0	0	0	0	0.02	0	0
8000	0	0	0	0	0	0	0	0	0	0	0	1.00	0	0
9000	0	0	0	0	0	0	0	0	1.00	0	0	0	0	0

Table 3 visualization of the performance of a KWS classification model on test dataset. Each row represents a true class (the actual keyword spoken) and each column represents a predicted class (the keyword the model thought was spoken). The diagonal elements show the correct classifications (e.g., the model correctly identified "Close" 98.2% of the time) and the off-diagonal elements show the misclassifications (e.g., the model incorrectly identified "No" as "Open" 0.5% of the time).

Table 3. Confusion Matrix for test dataset.

	Close	Hello	Hii	No	Off	On	Open	Silence	Sleep	Start	Stop	Wake	Welcome	Yes	Uncertain
Close	**98.2%**	0%	0%	1.4%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0.5%	0%
Hello	0%	**98.9%**	0.4%	0%	0%	0%	0.4%	0%	0%	0%	0%	0%	0.4%	0%	0%
Hi	0%	0%	**99.6%**	0%	0%	0%	0%	0%	0%	0%	0%	0.4%	0%	0%	0%
No	0%	0%	0%	**99.0%**	0%	0%	0.5%	0%	0%	0%	0%	0.5%	0%	0%	0%
Off	0%	0.5%	0%	0.5%	**95.0%**	1.4%	0.9%	0%	0%	0%	0%	0%	0%	0%	1.8%
On	0%	0%	0%	0%	0.9%	**99.1%**	0%	0%	0%	0%	0%	0%	0%	0%	0%
Open	0%	0%	0%	0%	0%	0%	**99.5%**	0%	0%	0%	0%	0%	0.5%	0%	0%
Silence	0%	0%	0%	0%	0%	0%	0%	**100%**	0%	0%	0%	0%	0%	0%	0%
Sleep	0%	0%	0%	0%	0%	0%	0%	0%	**100%**	0%	0%	0%	0%	0%	0%

Start	0%	0%	0%	0%	0%	0%	0%	0%	0%	**98.3%**	1.3%	0%	0%	0.4%	0%
Stop	0.8%	0%	0%	0.4%	0%	0%	0%	0%	0%	0.8%	**97.5%**	0%	0%	0%	0.4%
Wake	0%	0%	0.5%	0%	0%	0%	0%	0%	0%	0%	0%	**99.0%**	0%	0.5%	0%
Welcome	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0.5%	**99.5%**	0%	0%
Yes	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	1.0%	0%	**99.0%**	0%
F1 Score	0.99	0.99	0.99	0.98	0.97	0.99	0.99	1.00	1.00	0.99	0.98	0.98	0.99	0.99	

Table 4 summarizes the on-device performance of the quantized keyword spotting model for smart home automation on the target device. It includes the inference time, the maximum RAM usage, and the peak flash utilization observed. Table 4 also presents a side-by-side analysis of the quantized and un-optimized versions of the model. Quantization is a technique that reduces the model's size, memory footprint, and power consumption, making it more suitable for deployment on resource-constrained devices. The table compares various parameters, including:

- Latency: The total processing time required for the model to make a prediction.
- RAM Usage: The amount of RAM consumed by the model during operation.
- Flash Memory Usage: The amount of flash memory required to store the model.
- Accuracy: The highest achieved accuracy for the model.

By examining these metrics, we can assess the trade-offs between model size, performance, and accuracy when using quantization.

Table 4. Comparison of Quantized and Un-optimized Model.

Parameters	Quantized (int8)	Unoptimized (float32)
Latency	5 ms.	9 ms.
RAM	7.9 K	22.3 K
Flash	43.7 K	79.9 K
Accuracy	**98.65%**	**98.55%**

5. Conclusion

This research successfully developed a novel keyword spotting model developed for resource-constrained devices. The model achieved high accuracy while maintaining minimal computational requirements. The model's ability to accurately recognize keywords in real-time demonstrates its potential for practical applications in smart home automation and other voice-controlled devices. The model achieved an accuracy of 99.1% on the training dataset and 98.65% on the test dataset for the quantized version. The quantized model is optimized to address latency, memory footprint and accuracy associated with the deployment on resource-limited devices. The model demonstrated real-time capabilities, enabling prompt responses to voice commands. Future research could explore model's ability to handle more complex scenarios, such as background noise and multiple simultaneous speakers.

Author Contributions: J.M. designed methodology. Gathered dataset. Designed and developed the system and TinyML model. Prepared the original draft. T.M. conceptualized the idea for this manuscript. supervised and administered the work. Validated the data and results. A.H. carried out the investigation and data curation of the acquired data. validated the data and results. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the first author.

Acknowledgments: We are grateful for the Edge Impulse [19] platform, which was used to generate the TinyML model and graphics used in this study.

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

References

1. Guamán, S.; Calvopiña, A.; Orta, P.; Tapia, F.; Yoo, S.G. Device control system for a smart home using voice commands: A practical case. In Proceedings of the 2018 10th International Conference on Information Management and Engineering, Salford, UK, 22–24 September 2018; pp. 86–89.
2. Guo, B.; Zhang, D.; Yu, Z.; Liang, Y.; Wang, Z.; Zhou, X. From the internet of things to embedded intelligence. World Wide Web **2013**, *16*, 399–420.
3. Somesh, S.; Senthilnathan, N.; Sabarimuthu, M.; Kumar, A.S.; Rishikeshanan, R.; Bala, V. Real time Implementation of Home appliance control using ALEXA. *IOP Conf. Ser.: Mater. Sci. Eng.* **2020**, *937*, 012008. Available online: https://iopscience.iop.org/article/10.1088/1757-899X/937/1/012008/pdf (accessed on).
4. Kinnunen, T.; Chernenko, E.; Tuononen, M.; Fränti, P.; Li, H. Voice activity detection using MFCC features and support vector machine. In Int. Conf. on Speech and Computer (SPECOM07), Moscow, Russia, October 2007; Volume 2, pp. 556–561.
5. Chen, G.; Parada, C.; Heigold, G. Small-footprint keyword spotting using deep neural networks. In 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP), Florence, Italy, 4–9 May 2014; IEEE; pp. 4087–4091.
6. Warden, P. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv* **2018**, arXiv:1804.03209.
7. Singh, U.; Ansari, M.A. Smart home automation system using Internet of Things. In 2019 2nd International Conference on Power Energy, Environment and Intelligent Control (PEEIC), Greater Noida, India, 18-19 October 2019; IEEE; pp. 144–149.
8. Nabaz, D.; Shafie, N.; Azizan, A. Design Of Emergency Keyword Recognition Using Arduino Nano BLE Sense 33 And Edge Impulse. *Open Int. J. Inform.* **2023**, *11*, 46–57.
9. Yılmaz, E.; Gevrek, O.B.; Wu, J.; Chen, Y.; Meng, X.; Li, H. Deep convolutional spiking neural networks for keyword spotting. In Proceedings of INTERSPEECH, October 2020; pp. 2557–2561.
10. Hartmann, W.; Le Zhang, 0.0.0.2.; Barnes, K.; Hsiao, R.; Tsakalidis, S.; Schwartz, R.M. (2016, September). Comparison of Multiple System Combination Techniques for Keyword Spotting. In Interspeech, San Francisco, CA, USA, 8–12 September 2016; pp. 1913–1917).
11. Nyamukuru, M.T.; Odame, K.M. Tiny eats: Eating detection on a microcontroller. In 2020 IEEE Second Workshop on Machine Learning on Edge in Sensor Systems (SenSys-ML), Sydney, NSW, Australia, 21–21 April 2020; IEEE; pp. 19–23.
12. Yar, H.; Imran, A.S.; Khan, Z.A.; Sajjad, M.; Kastrati, Z. Towards smart home automation using IoT-enabled edge-computing paradigm. Sensors **2021**, *21*, 4932. Available online: https://www.mdpi.com/1424-8220/21/14/4932 (accessed on).
13. Kamble, A.; Mulani, A.O. Google assistant based device control. *Int. J. Aquat. Sci.* **2022**, *13*, 550–555.
14. Kamilaris, A.; Trifa, V.; Pitsillides, A. (2011, May). HomeWeb: An application framework for Web-based smart homes. In 2011 18th International Conference on Telecommunications, Ayia Napa, Cyprus, 8–11 May 2011; IEEE; pp. 134–139.
15. Bepery, C.; Baral, S.; Khashkel, A.; Hossain, F. Advanced home automation system using Raspberry-Pi and Arduino. *Int. J. Comput. Sci. Eng.* **2019**, *8*, 1–10.
16. Gunge, V.S.; Yalagi, P.S. Smart home automation: A literature review. *Int. J. Comput. Appl.* **2016**, *975*, 8887–8891.
17. Chandra, E.; Senthildevi, K.A. Keyword spotting: An audio mining technique in speech processing–a survey. *IOSR J. VLSI Signal Process.* **2015**, *5*, 22–27.
18. Rai, S.; Li, T.; Lyu, B. Keyword spotting--Detecting commands in speech using deep learning. *arXiv* **2023**, arXiv:2312.05640. Available online: https://arxiv.org/html/2312.05640v1 (accessed on).
19. Edge Impulse. TinyML Tool. 2024. Available online: https://www.edgeimpulse.com (accessed on).
20. Nicla Vision. Available online: https://store.arduino.cc/products/nicla-vision (accessed on).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.