# Designing Tiny Machine Learning Models for Keyword Spotting Using Knowledge Distillation – A Comparative Review

**Selvaperumal p,**

Selvaperumal[1], Mashael M. Khayyat[2]

[1] Lincoln University College, Malaysia and St Joseph's University, Bengaluru; [2] Lincoln University College, Malaysia.

Email ID selvaperumal.p@gmail.com

**Abstract**:

Small-footprint Keyword Spotting (KWS) refers to the process of spotting key-words locally within highly optimized, resource-constrained IoT systems. Since these devices generally have low capability in computation, memory and power, the design of lightweight and accurate keyword spotting models is a big challenge of research. Tiny Machine Learning (TinyML) is a solution to this challenge that allows for the deployment of deep learning models on edge devices with limited resources. But size reduction invariably results in a drop in accuracy of recognition. Recently, Knowledge Distillation (KD) has been a promising method to compress knowledge from a large and accurate teacher model to a small student model while retaining competitive performance, and it has attracted significant research interest. In this review, the recent progress of TinyML-based keyword spotting models with knowledge distillation is compared. Different learning strategies, small neural network architectures, model compression methods, and deployment are explored in terms of model size, computational cost, latency for inference, and recognition accuracy. Moreover, existing methods are reviewed with their pros and cons and research directions such as self supervised learning, adapt knowledge distillation, few-shot learning, and hardware-aware model optimization are highlighted. The purpose of this review is not only to give researchers a thorough overview of the latest advances and future directions for efficient TinyML keyword spotting, but also to highlight the challenges encountered by current implementation methods.

**Keywords**: Keyword Spotting, Knowledge Distillation, TinyML, Model Compression, Efficient Neural Networks, Small-Footprint Models, Edge AI, Noise Robustness

## Introduction

A proliferation of the Internet of Things (IoT), wearable electronics, smart home devices and voice-enabled embedded systems has greatly enhanced the need for efficient on-device speech processing. Keyword Spotting (KWS) is a technique that enables the recognition of specific words or phrases that have been stored in the system or to recognize short voice commands, and is used as a front-end module for many intelligent voice assistants, including Google Assistant, Amazon Alexa, and Apple Siri. Tiny Machine Learning (TinyML), as opposed to traditional cloud-based speech recognition systems, allows KWS models to be run on low-power, low-memory, low-computation microcontrollers, while consuming minimal energy. On-device inference not only decreases communication latency, but also enhances user privacy since speech signals need not be continuously sent to cloud servers. Thus, the

problem of always-on voice interaction through intelligent edge devices is gaining popularity and TinyML-based keyword spotting has emerged as an active research field for the development of such intelligent devices.

While deep learning methods have greatly enhanced the performance of keyword spotting systems, putting modern neural networks into practice on devices with limited resources can be challenging. While conventional convolutional and recurrent neural networks can be large with millions of parameters and need a lot of computing power, they may not be suitable for embedded devices with only a few hundred kilobytes of memory. To overcome these drawbacks, many light weight architectures have been proposed such as Depthwise Separable Convolutional Neural Network (DS-CNN) [5], BC-ResNet [6], MobileNet based models, LiCo-Net, TENet and Typman-KWS that greatly reduce model size and computational complexity while preserving competitive recognition accuracy [1]–[4]. Moreover, there are other methods to optimize the model, like quantization, pruning, neural architecture search, and hardware-aware optimization, that further enhanced the deployment feasibility of keyword spotting models on TinyML platforms.

Knowledge Distillation (KD) is one of the most promising methods for compacting accurate TinyML models from various model compression methods. The key idea of knowledge distillation is to distillate the knowledge learned by a large and well-trained teacher network to a lightweight student network, such that the student model can obtain knowledge comparable with the teacher with much fewer parameters and lower computation complexity. In recent years, TinyML-based keyword spotting has been successfully extended to include curriculum learning, ensemble learning, multi-granular distillation, self-supervised learning, and even dataset distillation, which improves the robustness, generalization, few-shot and zero-shot learning, as well as continual learning capabilities [5]–[10]. The advances are not only indicative of the maturity of knowledge distillation as a fundamental technology for achieving both recognition accuracy and deployment efficiency for resource-constrained edge devices, but also show the extent of progress that can be made in the field.

Although these are important advances, the literature on the subject is mainly concerned with proposing individual architectures or specific knowledge distillation technique, which makes it difficult to know which are their strengths and weaknesses and their deployment trade-offs. In addition, there are very few systematic comparisons of lightweight TinyML architectures, varying teacher-student learning strategies, deployment platforms, evaluation datasets, and performance metrics in a single framework. Thus, this paper provides a complete comparative study of TinyML-based keyword spotting models based on knowledge distillation (KD). The review briefly discusses recent lightweight neural network architectures and explores different knowledge distillation strategies for TinyML deployment, as well as comparing previous approaches with respect to their computational efficiency and recognition performance. It highlights current research challenges and outlines a set of possible future research directions for developing next generation TinyML keyword spotting systems.

**Related work**

One of the most popular applications of Tiny Machine Learning (TinyML) is Keyword Spotting (KWS) which allows for ubiquitous speech recognition for smart speakers, wearables, and Internet of

Things (IoT) devices. Previous research was mainly concerned with creating lightweight neural network architectures that are still able to meet the memory, computational and energy requirements of microcontrollers. Zhang et al. proposed the Hello Edge framework, which systematically analyzed the deployment of several neural network architectures to microcontroller and showed that Depthwise Separable Convolutional Neural Network (DS-CNN) is an excellent choice for embedded KWS applications with a balance of recognition accuracy, memory footprint and computation efficiency [1]. After that, Kim et al. introduced the novel broadcasted residual learning mechanism, which combines one dimensional temporal convolutions with two dimensional frequency convolutions, to obtain less than 10,000 parameters and achieve state-of-the-art accuracy with a small computational complexity [2]. Bartoli et al. also explored end-to-end optimization of TinyML-based KWS systems, taking into account the features being extracted, inference latency, model size, and deployment efficiency, and presented lightweight architectures based on MobileNet architectures which are suitable for real-time implementation on a microcontroller [3]. Recently, Garai and Samui (2020) provided a thorough review of the small-footprint keyword spotting techniques, which included lightweight architectures, model compression, quantization, neural architecture search, attention mechanisms, and TinyML deployment strategies, and highlighted model compression as an important research avenue for edge-AI systems [4].

Knowledge distillation (KD) is an emerging model compression technique that has gained a significant amount of attention for creating efficient TinyML-based KWS models. Lim and Baek presented the first joint framework that places curriculum learning together with knowledge distillation - large teacher network is gradually trained in increasingly noisy environment and then knowledge is transferred to compact student network. They have a very powerful solution to make the model robust to noise while maintaining a low model size for microcontroller use [5]. To apply knowledge distillation in in-memory keyword spotting, Song et al. proposed a light-weight neural encoder which helps reduce the memory requirement for the front-end processing while maintaining the recognition performance, compared with the conventional MFCC-based front-end processing [6]. Sun et al. introduced a multi-granular knowledge distillation (MG-KD) framework that distills knowledge at the utterance level, phoneme level, and feature level from a large teacher model to a lightweight LiCo-Net student network with a memory usage of less than 0.5 MB, significantly reducing equal error rate [7]. Likewise, Gok et al. used large self-supervised speech models as teacher networks and transferred their learnt representations to a small model ResNet-15 which was more efficient on the edge, and obtained a significant boost in few-shot keyword spotting performance [8].

In recent years, knowledge distillation has been combined with other optimization methods to further improve TinyML deployment. Ensemble-based knowledge distillation (E-KD) was used to optimize the MobileNetV2 architecture, which yielded lightweight student models that achieved better recognition accuracy, noise immunity, computational complexity, and inference speed on edge devices when compared with traditional DS-CNN architectures [9]. In addition to inference optimization, Rub et al. also explored continual TinyML learning, which involves reducing the memory footprint of the system and distilling a dataset to enable efficient incremental on-device learning without catastrophic forgetting under extreme memory constraints. With just a minority of the original data and computation, their approach can achieve competitive performance [10]. While these studies have shown that different knowledge distillation methods can achieve good performance for TinyML-based keyword spotting, they have focused on single architectures or particular deployment scenarios. There is, however, a lack of

detailed comparative analysis of various knowledge distillation techniques, lightweight architectures, deployment requirements and trade-offs for TinyML based keyword spotting. This observation is the impetus behind the present review. The below table 1 compares the some of the various existing keyword spotting architectures.

*Table 1. Comparison of existing KWS architectures and deployment characteristics.*

| Reference | Main Contribution | TinyML Architecture | Knowledge Distillation | Deployment on Edge Device | Evaluation Dataset | Remarks |
|---|---|---|---|---|---|---|
| [1] | Hello Edge framework | DS-CNN | No | Yes | Google Speech Commands | Baseline TinyML architecture comparison |
| [2] | BC-ResNet | BC-ResNet | No | Yes | Google Speech Commands | Broadcasted residual learning improves accuracy |
| [3] | End-to-End KWS | MobileNet-based CNN | No | Yes | GSC V2 | Optimizes complete deployment pipeline |
| [4] | Survey | Multiple architectures | Discussed | Yes | Literature | Comprehensive review of SF-KWS techniques |
| [5] | Noise-robust KWS | CNN | Yes | Yes | Public noisy datasets | Curriculum learning + KD |
| [6] | In-memory KWS | Lightweight Neural Encoder | Yes | Yes | Google Speech Commands | KD for in-memory computing |
| [7] | Zero-shot KWS | LiCo-Net | Yes | Yes | Qualcomm Dataset | Multi-granular KD |
| [8] | Few-shot KWS | ResNet-15 | Yes | Yes | MSWC, GSC | Self-supervised teacher model |
| [9] | Edge KWS | MobileNetV2 | Yes | Yes | GSC V2 | Ensemble KD improves robustness |
| [10] | Continual TinyML | Adaptive CNN | Yes | Yes | Speech Commands | Dataset distillation + continual learning |

**Key Contribution**

This paper reviews recent small-footprint Tiny Machine Learning (TinyML) models for keyword spotting, especially focusing on knowledge distillation techniques for resource-constrained edge devices. The review presents and discusses the characteristics and limitations of the available lightweight architectures, teacher–student learning strategies, deployment platforms, evaluation datasets, and performance metrics, while explaining their pros, cons and practical considerations. Moreover, the paper presents some current research challenges and suggests future research directions to build an accurate, computationally efficient keyword spotting system based on TinyML.

*Table 2. Comparison of Knowledge Distillation Strategies in TinyML-based Keyword Spotting*

| Ref. | Teacher Model | Student Model | Distillation Type | Objective Function | Performance Gain | Limitation |
|---|---|---|---|---|---|---|
| [5] | CNN | Lightweight CNN | Curriculum KD | Soft labels | Improved noise robustness | Only noisy environments |
| [6] | Large CNN | Neural Encoder | Response-based KD | Output logits | Efficient in-memory KWS | Hardware specific |
| [7] | GACL | LiCo-Net | Multi-granular KD | MSE + KL + InfoNCE | 36% EER reduction | Zero-shot only |
| [8] | SSL Speech Model | ResNet-15 | Feature-based KD | Embedding matching | Better few-shot accuracy | Few-shot only |
| [9] | CNN Ensemble | MobileNetV2 | Ensemble KD | Soft labels | Better noisy accuracy | One architecture |
| [10] | Large CNN | Adaptive CNN | Dataset Distillation | Distilled dataset | Continual learning | Not standard KWS |

The above Table 2 presents a comparative analysis of the knowledge distillation strategies employed in TinyML-based keyword spotting models, highlighting the teacher–student architectures, distillation techniques, performance improvements, and their limitations.

**Conclusions**

Knowledge distillation has been a key technique to make keyword spotting on resource-constrained embedded devices possible. This comparison shows that the use of curriculum learning with weighted ensemble distillation yields state-of-the-art results, with an accuracy of 78.6% at −12.5 dB SNR on MUSAN, improving by 3–5% compared to baselines of CNN and DNN. The standard knowledge distillation method preserves the accuracy of 90–95% on clean speech, while the ensemble-based approaches show a more significant improvement of 2–3% in adverse acoustic conditions. The proposed adaptive noise-aware weighting mechanism that selects teacher snapshots from curriculum stages that correspond to the SNR distribution of training samples directly tackles the critical trade-off in maintaining noise robustness while complying with the strict microcontroller memory and power constraints. The results are verified on four publicly-available datasets (GSCD, MUSAN, QUT, UrbanSound8K) with a wider SNR range of 0 dB to −15 dB that will help the field move toward standardized benchmarking for small footprint KWS systems.

This comparative work shows that the knowledge distillation technique is one of the foundations of engineering the high fidelity, small footprint keyword spotting model which can maximize classification

accuracy under the limited power and memory constraints of edge devices. In the future, the studies can also consider the system-level execution patterns in depth, and explicitly match the structural properties of the distilled student networks with the hardware-integrated Neural Processing Units (NPUs). In addition, a fundamental paradigm shift is taking place from static offline learning to dynamic on-device continuous learning. Combining data condensation with sparse update techniques that save memory will enable ultra-low power edge devices to continuously learn new keywords and accent variations for the user, while avoiding catastrophic forgetting. Addressing these architectural and deployment challenges will pave the way for building secure, privacy-protecting, and highly-adaptive voice interfaces in the growing global IoT and wearable market.

**References**


1. Y. Zhang, N. Suda, L. Lai, and V. Chandra, “Hello Edge: Keyword Spotting on Microcontrollers,” in Proc. Machine Learning for Mobile and Embedded Devices Workshop (NeurIPS Workshop), Long Beach, CA, USA, 2017.
2. B. Kim, S. Chang, J. Lee, and D. Sung, “Broadcasted Residual Learning for Efficient Keyword Spotting,” in Proc. Interspeech 2021, Brno, Czech Republic, Aug. 2021, pp. 4548–4552.
3. P. Bartoli, T. Bondini, C. Veronesi, A. Giudici, N. Antonello, and F. Zappa, “End-to-End Efficiency in Keyword Spotting: A System-Level Approach for Embedded Microcontrollers,” arXiv preprint arXiv:2509.07051, 2025.
4. S. Garai and S. Samui, “Advances in Small-Footprint Keyword Spotting: A Comprehensive Review of Efficient Models and Algorithms,” Engineering Applications in Artificial Intelligence, preprint, 2025.
5. J. Lim and Y. Baek, “Joint Framework of Curriculum Learning and Knowledge Distillation for Noise-Robust and Small-Footprint Keyword Spotting,” IEEE Access, vol. 11, pp. 100540–100553, 2023.
6. Z. Song, Q. Liu, Q. Yang, and H. Li, “Knowledge Distillation for In-Memory Keyword Spotting Model,” in Proc. Interspeech 2022, Incheon, South Korea, Sep. 2022, pp. 4128–4132.
7. Y.-T. Sun, L.-Y. Li, T.-H. Lo, J.-W. Hung, S.-C. Huang, and B. Chen, “Lightweight Zero-Shot Keyword Spotting via Multi-Granular Knowledge Distillation,” in Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2025, pp. 1182–1187.
8. A. Gok, O. Buyuksolak, O. E. Okman, and M. Saraclar, “Enhancing Few-Shot Keyword Spotting Performance through Pre-Trained Self-Supervised Speech Models,” IEEE Signal Processing Letters, submitted, 2025.
9. C. K. Wei and H. Nugroho, “Design of a DNN-Based Operator on Edge Device for Keyword Spotting,” Advances in Computational Intelligence, vol. 5, no. 2, 2025.
10. M. Rub, P. Tuchel, A. Sikora, and D. Mueller-Gritschneder, “A Continual and Incremental Learning Approach for TinyML On-Device Training Using Dataset Distillation and Model Size Adaption,” arXiv preprint arXiv:2409.07114, 2024.