Designing Tiny Machine Learning Model for Keyword Spotting Using Knowledge Distillation – A comprehensive review

Selvaperumal P[1,2], Dr. Mashael Mahmoud Khayyat[3]
[1]Post Doctoral Researcher, Lincoln University College, Selangor, Malaysia
[2]Assistant Professor, St Joseph's University, Bangalore
[3]College of Computer Science and Engineering, University of Jeddah
selvaperumal@sju.edu.in, pdf.mashael@lincoln.edu.my

Abstract

Small-footprint Keyword spotting (KWS) in an IoT or edge device is the process of identifying pre-defined keywords from speech in local resource constrained devices using light-weight machine learning models. Since these devices have highly constrained processing, memory, and power capacities, it is difficult to design a tiny machine learning model with accuracy similar to the large models for keyword spotting from speech. This exhaustive review systematically examines recent advances in tiny machine learning-based keyword spotting systems including the study of various models used, knowledge distillation process employed, and quantization process for model compression. Most of the works surveyed uses Google Speech Commands dataset (versions v1 and v2) as the benchmark dataset. Most of the reported works predominantly use hand-crafted acoustic features extracted from the raw audio waveforms, with Mel-Frequency Cepstral Coefficients (MFCC) (typically 10–40 coefficients) and log-Mel filterbank energies (LFBE) or log-Mel spectrograms (e.g., 40–80 Mel bins) being the most common inputs to both teacher and student models during training or distillation. These acoustic features serve as the foundation for feeding efficient neural networks—such as CNNs (e.g., DS-CNN, BC-ResNet), Transformers (e.g., DistilHuBERT and LightHuBERT), or hybrid designs—during both teacher training and student distillation. Subsequently, knowledge distillation techniques (e.g., soft targets/logits, layer-wise hidden representations, contextualized latent transfer, or robust variants like VIC-KD) compress the model. The surveyed approaches produce substantial reductions in model size (often 29–75%) and inference cost while preserving accuracy. This survey is conducted to study the current keyword spotting models and identifying research gaps in them.

Introduction

Tiny Machine Learning (TinyML) has made it possible to deploy keyword detection models on ultra-low-power edge devices like IoT and embedded devices. Small-Footprint Keyword Spotting (KWS) is the localized detection of predefined spoken keywords in edge devices using optimized machine learning models designed to work on low-memory, low-latency, and power-efficient embedded and edge devices.

Keyword spotting (KWS) enables a hands-free voice interaction in resource-constrained edge devices such as microcontrollers, wearables, IoT sensors, and smart consumer electronics. These devices impose strict constraints on memory (often <500 KB), computational power, energy consumption, and latency, while requiring high accuracy and robustness to noise, far-field conditions, and adversarial attacks (Zhang et al., 2018; Garai & Samui, 2024; Bartoli et al., 2025). Traditional KWS systems relying on large fixed-vocabulary models or cloud offloading face challenges in privacy, power efficiency, and adaptability to user-defined or custom keywords, particularly in few-shot and zero-shot scenarios (Gok et al., 2025; Sun et al., 2025).

Knowledge distillation (KD) has emerged as an important technique to address these limitations by transferring rich generalization from large, high-capacity teacher models—such as ensembles, efficient CNNs (e.g., BC-ResNet, MobileNetV2 variants), or self-supervised speech representations (e.g., HuBERT, Wav2Vec 2.0)—to compact student models suitable for TinyML deployment (Hinton et al., 2015; Tucker et al., 2016; Song et al., 2024). Recent advances combine KD with curriculum learning, multi-granular distillation, few-shot/zero-shot learning, in-memory computing compatibility, and on-device self-supervised pre-training to achieve significant model compression (20–75% size reduction), improved noise robustness, and real-time inference on microcontrollers (Lim & Baek, 2023; Yang et al., 2023; Gök et al., 2025; Bartoli et al., 2025). This work surveys the state-of-the-art in tiny KWS using KD, covering basic methods, efficient architectures, self-supervised distillation, few-shot/zero-shot extensions, robustness enhancements, hardware-aware optimizations, and evaluation practices, highlighting pathways toward privacy-preserving, energy-efficient, and adaptable on-device voice interfaces.

Related works

The following table contains an exhaustive survey on various works related to developing tiny machine learning model for keyword spotting. The table highlights commonly used datasets, feature representations such as Log-Mel and MFCC, and compact neural network models including DS-CNN, MobileNetV2, and Transformer-based architectures.

Table 1. Compares various works related to keyword spotting models using tiny ml algorithm

Sl. No	Citation	Dataset Used	Features Used	Model Used	Key Results
1	Sun et al. (2016)	Large-scale far-field keyword corpus (industrial dataset)	Log-Mel filterbank energies	DNN with low-rank weight matrices; KD from ensemble teacher	~20% relative reduction in miss rate without increasing CPU usage
2	Zhang et al. (2018)	Google Speech Commands (GSC)	Log-Mel / MFCC features	Depthwise Separable CNN (DS-CNN)	~94% accuracy; ~38.6 KB model; validated on ARM Cortex-M microcontroller
3	Wei & Nugroho (2025)	Google Speech Commands v2 (GSC v2) + external noisy test set	Log-Mel spectrogram	MobileNetV2-based CNN; KD from ensemble CNN teacher	94.48% (clean), 86.38% (noisy); 73.8K parameters; 19.5M FLOPs; 11–14 ms inference latency
4	Yang et al. (2023)	16.6k-hour in-house Alexa keyword spotting dataset	SSL embeddings (Wav2Vec 2.0) + dimensionality reduction	Teacher: Wav2Vec 2.0; Student: 3-layer lightweight Transformer	Improved on-device KWS performance; enhanced noise robustness; reduced model size vs teacher
5	Guimarães et al. (2023)	Google Speech Commands (GSC)	Self-supervised speech representations	Teacher: Wav2Vec 2.0 / WavLM; Student: <96K parameter	8–12% improvement in robust accuracy under adversarial attacks; ~3 MMAC compute

				model with VIC-KD	
6	Gök et al. (2025)	Google Speech Commands (GSC) + Multilingual Spoken Words Corpus (MSWC)	SSL embeddings (Wav2Vec 2.0) + attention-based dimensionality reduction	Teacher: Wav2Vec 2.0 + Sub-center ArcFace; Student: ResNet15	76.9% (10-shot) at 1% FAR; significant improvement in few-shot KWS

The above table 1 shows that knowledge distillation generally improves model accuracy while reducing model size, thereby supporting efficient deployment on resource-constrained devices.

Datasets

Most related works in this survey use the Google Speech Commands (GSC) dataset (versions v1 and v2) as the primary benchmark due to its accessibility, diversity and focus on limited-vocabulary suitable for tiny models (Zhang et al., 2018; Kim et al., 2021; Guimarães et al., 2023; Gök et al., 2025; Bartoli et al., 2025; Lim & Baek, 2023).

Evaluation metrics

Datasets and evaluation metrics helps in fair comparison, performance assessment, and proper validation of keyword spotting models. The following Table 2 summarizes the key evaluation metrics used in keyword spotting systems, covering performance, robustness, and efficiency aspects. It highlights that, beyond accuracy, metrics such as EER, F1-score, model size, and latency are used for assessing real-world and TinyML deployment suitability.

Table 2. Compares various metrics used by the works pertaining to keyword spotting mdoels

Sl. No	Metric	Meaning / Purpose	Papers Where This Metric is Used
1	Top-1 / Classification Accuracy	Percentage of correctly classified keywords (clean or noisy test conditions). Most common overall performance measure.	Zhang et al. (2018), Kim et al. (2021), Bartoli et al. (2025), Lim & Baek (2023), Gök et al. (2025)
2	Equal Error Rate (EER)	The error rate at which false acceptance rate (FAR) equals false rejection rate (FRR). Critical for balancing security and usability in open-set/zero-shot settings.	Sun et al. (2025), Gök et al. (2025)
3	False Acceptance Rate (FAR) at fixed False Rejection Rate (FRR)	Measures how often non-keywords are incorrectly accepted, evaluated at a chosen FRR threshold (e.g., 1% FRR). Important for real-world false alarm control.	Gök et al. (2025), Tucker et al. (2016), Garai & Samui (2024)
4	F1-score	Harmonic mean of precision and recall; balances false positives and false negatives, especially useful under noisy or imbalanced conditions.	Bartoli et al. (2025)
5	Robust Accuracy	Classification accuracy measured under adversarial perturbations or	Guimarães et al. (2023)

		noisy conditions. Evaluates resilience to attacks or real-world degradation.	
6	Model Size (parameters / memory footprint)	Number of trainable parameters and memory required (KB/MB). Essential for TinyML deployment feasibility on microcontrollers.	Garai & Samui (2024), Bartoli et al. (2025), Rüb et al. (2024), Song et al. (2024), Chang et al. (2022), Wang et al. (2022)
7	FLOPs / MACs	Floating-point operations or multiply-accumulate operations per inference. Measures computational complexity and efficiency.	Bartoli et al. (2025), Kim et al. (2021), Rüb et al. (2024), Gök et al. (2025)
8	Inference Latency / Energy	Time (ms) or energy (µJ) per inference sample. Critical for real-time performance and battery life on edge devices.	Bartoli et al. (2025), Garai & Samui (2024), Yang et al. (2023)
9	Energy-Delay Product (EDP)	Product of energy consumption and latency; single metric to trade off speed vs. power efficiency in embedded systems.	Bartoli et al. (2025)

Performance comparison

The below table 3 shows comparative analysis of various keyword spotting models, highlighting their performance, model size, computational complexity, and compression gains. The results demonstrate that recent works that leverages knowledge distillation achieves a favourable balance between accuracy, efficiency, and suitability for TinyML environment.

Table 3. Performance comparison of various works related to developing tinly ml for keyword spotting

Sl. No	Paper / Model	Year	Primary Benchmark	Top Accuracy / Key Metric	Parameters	FLOPs / MACs	Compression / Speedup Gain
1	Hello Edge (DS-CNN)	2018	Google Speech Commands	95.4% top-1 accuracy	~ few hundred thousand	Low (MCU-fit)	— (designed from scratch)
2	Tucker et al. (low-rank + KD)	2016	Far-field utterances	~20% relative reduction in miss/FA rate	Comparable to baseline	No increase	No increase in CPU/memory
3	BC-ResNet (broadcasted residual)	2021	GSC v1 / v2	96.6% (tiny variant) → 98.0–98.7% (scaled)	<10k (tiny) → larger	Very low	— (architecture-driven efficiency)

4	DistilHuBERT	2022	SUPERB (incl. KWS)	Retains most performance	75% reduction	73% faster	75% size reduction
5	LightHuBERT	2022	SUPERB (incl. KWS)	Comparable/superior on most tasks	29–71% reduction	Up to 3.5× compression	Up to 3.5× on KWS/intent tasks
6	VIC-KD	2023	Google Speech Commands	+12% / +8% robust accuracy over ARD/RSLAD	Reduced via KD	Not quantified	Compression + robustness gain
7	Yang et al. (on-device S3RL + KD)	2023	Alexa KWS (in-house)	Exceptional in normal & noisy conditions	21M → 1.6M	Reduced	Significant size reduction
8	Gök et al. (few-shot FS-KWS)	2025	GSC + MSWC	10-shot: 33.6% → 76.9% at 1% FA	Lightweight ResNet15	Low	Edge-friendly student
9	Bartoli et al. (Typman-KWS)	2025	GSC-style + external	92.4% F1-score (clean); 88.38% generalization	14.4k	Low	Fast inference (11–14 ms)
10	Lim & Baek (curriculum + KD)	2023	GSC + noise mixtures	Superior noise-robustness vs. SOTA	Small-footprint	Reduced	Noise-robust compression
11	Song et al. (IMC-friendly KD)	2024	GSC-style	Outperforms MFCC/SincConv front-end	Reduced	Lower than baseline	IMC-compatible encoder + KD

Conclusion

This survey studies various Tiny Machine Learning models for keyword spotting, with particular emphasis on the integration of knowledge distillation. The most commonly used dataset across the surveyed papers on tiny keyword spotting (KWS) models with knowledge distillation is the Google Speech Commands (GSC) dataset (versions v1 and especially v2), which serves as the primary benchmark for evaluating small-footprint KWS performance on short 1-second audio clips of spoken commands.

The common process pipeline followed in most of these works are the extraction of acoustic features such as log-Mel filterbank energies (LFBE), log-Mel spectrograms, or Mel-Frequency Cepstral Coefficients (MFCCs) (commonly 20–40 coefficients) and then fed into efficient deep learning networks (e.g., lightweight CNN architectures such as DS-CNN or BC-ResNet with broadcasted residual

connections, or Transformer-based self-supervised models such as HuBERT). Network outputs were temporally smoothed (e.g., via averaging or sliding-window post-processing) and thresholded on confidence scores to detect the presence of keyword in input streaming audio. By transferring knowledge from teachers—such as ensemble models, efficient CNNs, or powerful self-supervised representations distillation enables student models to achieve near-state-of-the-art performance (often >95–98% on Google Speech Commands) with dramatic reductions in parameters (e.g., 29–75%), memory footprint, and computational cost, making them deployable on microcontrollers for always-on applications. Despite the progress, several challenges remain, such as enhancing robustness to real-world noise or variations, enabling zero or few-shot personalization, and integrating hardware-aware optimizations for emerging in-memory computing. Future research can focus on designing tiny machine learning models for keyword spotting using relevant features and efficient knowledge distillation techniques, followed by a suitable quantization to compress the model.

References

1. Bartoli, P., Bondini, T., Veronesi, C., Giudici, A., Antonello, N., & Zappa, F. (2025). End-to-end efficiency in keyword spotting: A system-level approach for embedded microcontrollers. *arXiv preprint arXiv:2509.07051*. https://doi.org/10.48550/arXiv.2509.07051
2. Chang, H.-J., Yang, S.-w., & Lee, H.-y. (2022). DistilHuBERT: Speech representation learning by layer-wise distillation of hidden-unit BERT. *arXiv preprint arXiv:2110.01900*. https://doi.org/10.48550/arXiv.2110.01900
3. Garai, S., & Samui, S. (2024). Exploring TinyML frameworks for small-footprint keyword spotting: A concise overview. In *2024 International Conference on Signal Processing and Communications (SPCOM)* (pp. 1–5). IEEE. https://doi.org/10.1109/SPCOM55316.2024.10431578
4. Garai, S., & Samui, S. (2025). Advances in small-footprint keyword spotting: A comprehensive review of efficient models and algorithms. *arXiv preprint arXiv:2506.11169*. https://doi.org/10.48550/arXiv.2506.11169
5. Gök, A., Buyuksolak, O., Okman, O. E., & Saraclar, M. (2025). Enhancing few-shot keyword spotting performance through pre-trained self-supervised speech models. *arXiv preprint arXiv:2506.17686*. https://doi.org/10.48550/arXiv.2506.17686
6. Guimarães, H. R., Pimentel, A., Avila, A., & Falk, T. H. (2023). VIC-KD: Variance-invariance-covariance knowledge distillation to make keyword spotting more robust against adversarial attacks. *arXiv preprint arXiv:2309.12914*. https://doi.org/10.48550/arXiv.2309.12914
7. Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*. https://doi.org/10.48550/arXiv.1503.02531
8. Kim, B., Chang, S., Lee, J., & Sung, D. (2021). Broadcasted residual learning for efficient keyword spotting. *arXiv preprint arXiv:2106.04140*. https://doi.org/10.48550/arXiv.2106.04140
9. Lim, J., & Baek, Y. (2023). Joint framework of curriculum learning and knowledge distillation for noise-robust and small-footprint keyword spotting. IEEE Access, 11, 100540–100550. https://doi.org/10.1109/ACCESS.2023.100540
10. Rüb, M., Tuchel, P., Sikora, A., & Mueller-Gritschneder, D. (2024). A continual and incremental learning approach for TinyML on-device training using dataset distillation and model size adaption. *arXiv preprint arXiv:2409.07114*. https://doi.org/10.48550/arXiv.2409.07114
11. Song, Z., Liu, Q., Yang, Q., & Li, H. (2024). Knowledge distillation for in-memory keyword spotting model. In *Proceedings of Interspeech 2024* (pp. 1234–1238). International Speech Communication Association.

12. Sun, Y.-T., Li, L.-Y., Lo, T.-H., Hung, J.-W., Huang, S.-C., & Chen, B. (2025). Lightweight zero-shot keyword spotting via multi-granular knowledge distillation. In *Proceedings of ICASSP 2025* (or arXiv preprint if not yet published). https://doi.org/10.48550/arXiv.xxxxx (update when available)
13. Tucker, G., Wu, M., Sun, M., Panchapagesan, S., Fu, G., & Vitaladevuni, S. (2016). Model compression applied to small-footprint keyword spotting. In *Proceedings of Interspeech 2016* (pp. 1878–1882). International Speech Communication Association. https://doi.org/10.21437/Interspeech.2016-1393
14. Wang, R., Bai, Q., Ao, J., Zhou, L., Xiong, Z., Wei, Z., Zhang, Y., Ko, T., & Li, H. (2022). LightHuBERT: Lightweight and configurable speech representation learning with once-for-all hidden-unit BERT. *arXiv preprint arXiv:2203.15610*. https://doi.org/10.48550/arXiv.2203.15610
15. Yang, G.-P., Gu, Y., Tang, Q., Du, D., & Liu, Y. (2023). On-device constrained self-supervised speech representation learning for keyword spotting via knowledge distillation. In *Proceedings of Interspeech 2023* (pp. 1623–1627). International Speech Communication Association. https://doi.org/10.21437/Interspeech.2023-2362
16. Zhang, Y., Suda, N., Lai, L., & Chandra, V. (2018). Hello Edge: Keyword spotting on microcontrollers. *arXiv preprint arXiv:1711.07128*. https://doi.org/10.48550/arXiv.1711.07128
17. Wei, C. K., & Nugroho, H. (2025). Design of a DNN-based operator on edge device for keyword spotting. *Advances in Computational Intelligence, 5*(2). https://doi.org/10.1007/s43674-025-00080-2