

Lightweight Semi-supervised Domain Adaptation for Bearing Fault Diagnosis on Edge Devices

Chaoxuan Qiu

*College of Engineering, Zhejiang Normal University, Jinhua, China
qcx200003@163.com*

Abstract. Rolling bearings serve as critical components in rotating machinery, but practical fault diagnosis is affected by operating-condition changes, limited fault samples, and the restricted computing resources of edge devices. This paper proposes LiteDANN, a lightweight semi-supervised domain-adaptation framework for cross-condition bearing fault diagnosis. Drive-end vibration signals from the Case Western Reserve University bearing dataset are segmented and converted into 64×64 short-time Fourier transform spectrograms. A MobileNetV2-style backbone extracts compact features, while a gradient reversal layer and a domain classifier reduce the discrepancy between different operating conditions. Focal loss and weighted random sampling are used to improve learning from scarce and imbalanced target-domain fault samples. The 0 hp condition is used as the source domain, the noisy 3 hp condition is used as the target domain, and the unseen 2 hp condition is reserved for testing. The recorded experiment achieves 99.69% accuracy and 99.70% macro-F1 on the unseen test condition. The model contains 37,478 trainable parameters. After ONNX conversion and INT8 quantization, the model size is reduced to 0.10 MB, enabling fast real-time inference on resource-limited edge hardware. These results indicate that LiteDANN provides a practical basis for real-time bearing diagnosis on resource-constrained edge platforms.

Keywords: Bearing fault diagnosis, domain adaptation, lightweight neural network, edge computing, imbalanced learning

1. Introduction

Rolling bearings support motors, gearboxes, pumps, and other rotating machines. Failures cause vibration, shutdowns, and maintenance costs, making accurate diagnosis important. The CWRU dataset is a common benchmark [1].

Deep learning learns fault representations from vibration data [2], and mobile architectures reduce convolutional cost [3]. Loads, speed, sensors, and noise vary between training and service. Transfer learning reuses source knowledge [4], while domain-adversarial training promotes cross-domain features [5].

Imbalance is common: normal samples are abundant, whereas labeled faults are scarce. Models may favor majority cases; focal loss emphasizes difficult minority examples [6].

Edge devices impose memory and compute limits. MobileNetV2 reduces cost with inverted residuals and linear bottlenecks [7]; integer quantization uses lower-bit values [8].

This study integrates a 37,478-parameter extractor, gradient-reversal adaptation, imbalance-aware classification, and ONNX/INT8 conversion for CPU edge simulation. Retained target labels make the setting semi-supervised rather than unsupervised.

2. Theoretical background

2.1. Bearing fault mechanisms

Bearings contain inner and outer races, rolling elements, and a cage. A local defect produces periodic impacts and structural resonance; timing and amplitude depend on location, speed, load, and transmission path [9].

Four classes are considered: normal, inner-race, outer-race, and rolling-element faults. Their impacts yield distinct characteristic frequencies and amplitudes for four-class diagnosis [1, 9].

2.2. Short-time fourier transform

The raw signal is a one-dimensional sequence whose spectrum lacks event timing. STFT applies Fourier analysis to short overlapping windows [10]. For a discrete signal, the STFT is written as

$$X(m, f) = \sum_{n=-\infty}^{+\infty} x[n] w[n-m] \exp(-j2\pi fn) \quad (1)$$

Here, $x[n]$ is the vibration signal, $w[n-m]$ the sliding window, m its position, and f frequency. STFT magnitude forms a logarithmic spectrogram: horizontal axis time, vertical axis frequency, and color spectral magnitude. Signals are sampled at 12 kHz, segmented into 2,048 points with 1,024-point stride, and transformed using a 256-point window with 192-point overlap. Spectrograms are resized to 64×64 and standardized.

3. Proposed method

3.1. Overall framework

Figure 1 shows the pipeline. Source and target vibrations are segmented and converted to STFT spectrograms, then encoded by LiteDANN. The fault head predicts bearing class; the domain head predicts source or target through gradient reversal. The trained checkpoint is exported to ONNX and quantized to INT8.

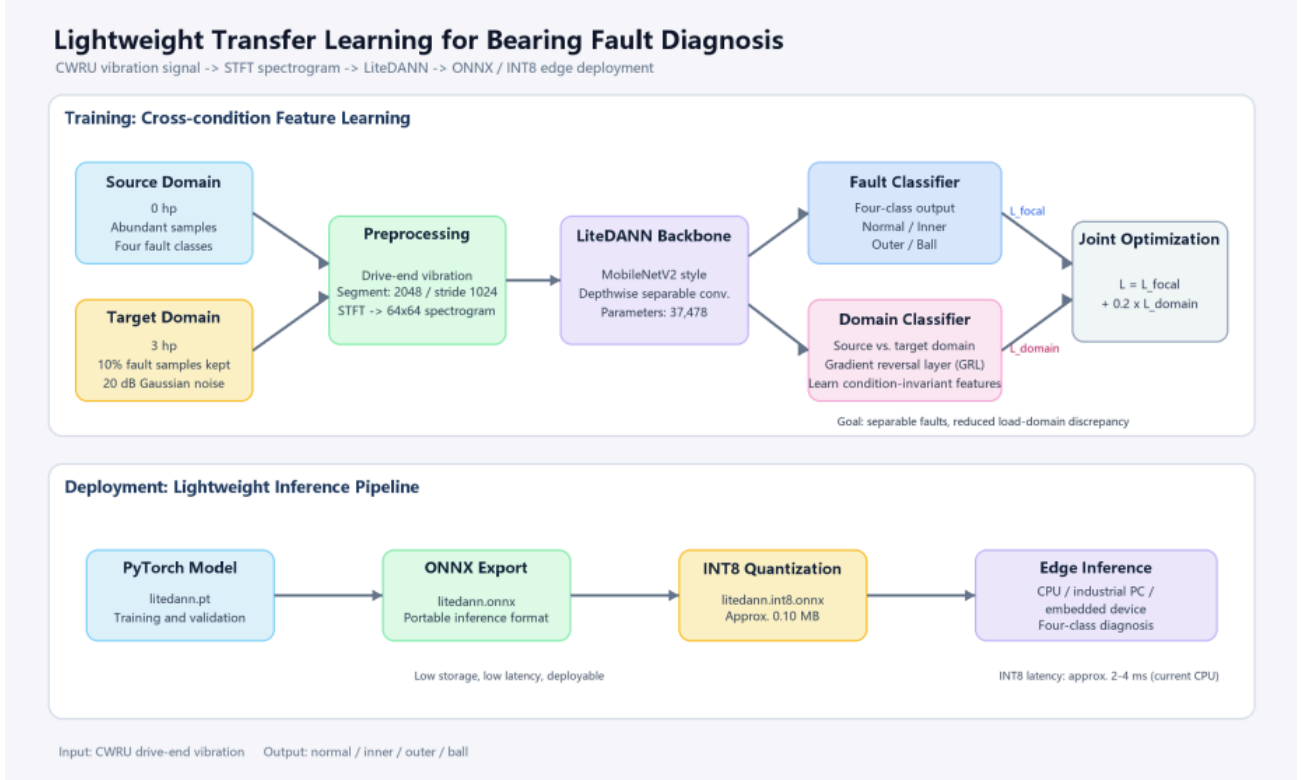


Figure 1. Overall architecture of the proposed LiteDANN framework

3.2. Domain construction

Source data use 0 hp with four classes; target data use 3 HP. To model noisy low-data operation, about 10% of target fault segments are retained and 20 dB Gaussian noise is added. The 2 hp condition is held out as an unseen test domain to assess generalization beyond adaptation loads.

3.3. Litedann architecture

Input is a single-channel 64×64 spectrogram. The backbone follows MobileNetV2 with depthwise separable convolution, inverted residuals, and linear bottlenecks. Depthwise convolution processes each channel spatially; pointwise convolution fuses channels. This decomposition reduces parameters and computation for edge diagnosis [7].

Global average pooling produces a compact vector for two heads. The fault head outputs four logits (normal, inner-race, outer-race, rolling-element); the domain head outputs source/target logits. The latter provides an adversarial signal rather than the final diagnosis.

3.4. Domain-adversarial learning

GRL is identity in forward propagation and multiplies the domain gradient by $-\alpha$ during backpropagation ($\partial R / \partial x = -\alpha I$). The domain classifier separates source and target, while the feature extractor makes them harder to distinguish. The fault head preserves four-class discrimination, reducing load information while retaining fault cues [5].

3.5. Joint loss function

Because target faults are scarce and imbalanced, classification uses focal loss and class weighting; weighted sampling increases minority selection. The objective is

$$L = L_focal \wedge source + L_focal \wedge target + 0.2 L_domain \quad (2)$$

Source and target focal terms preserve fault discrimination, while the domain term supplies adaptation with coefficient 0.2. Focal loss is $L_focal = -(1-p_t)^\gamma \log(p_t)$, where p_t is the correct-class probability and γ emphasizes difficult samples [6].

4. Experimental setup

4.1. Dataset and preprocessing

CWRU provides labeled drive-end signals for normal and seeded-fault bearings [1]. Labels are normal, inner, outer, and ball. MATLAB files are loaded from the drive-end time-series field and segmented before STFT; adjacent examples overlap but retain fixed length. Figure 2 shows target spectrograms after 3 HP and 20 dB noise processing.

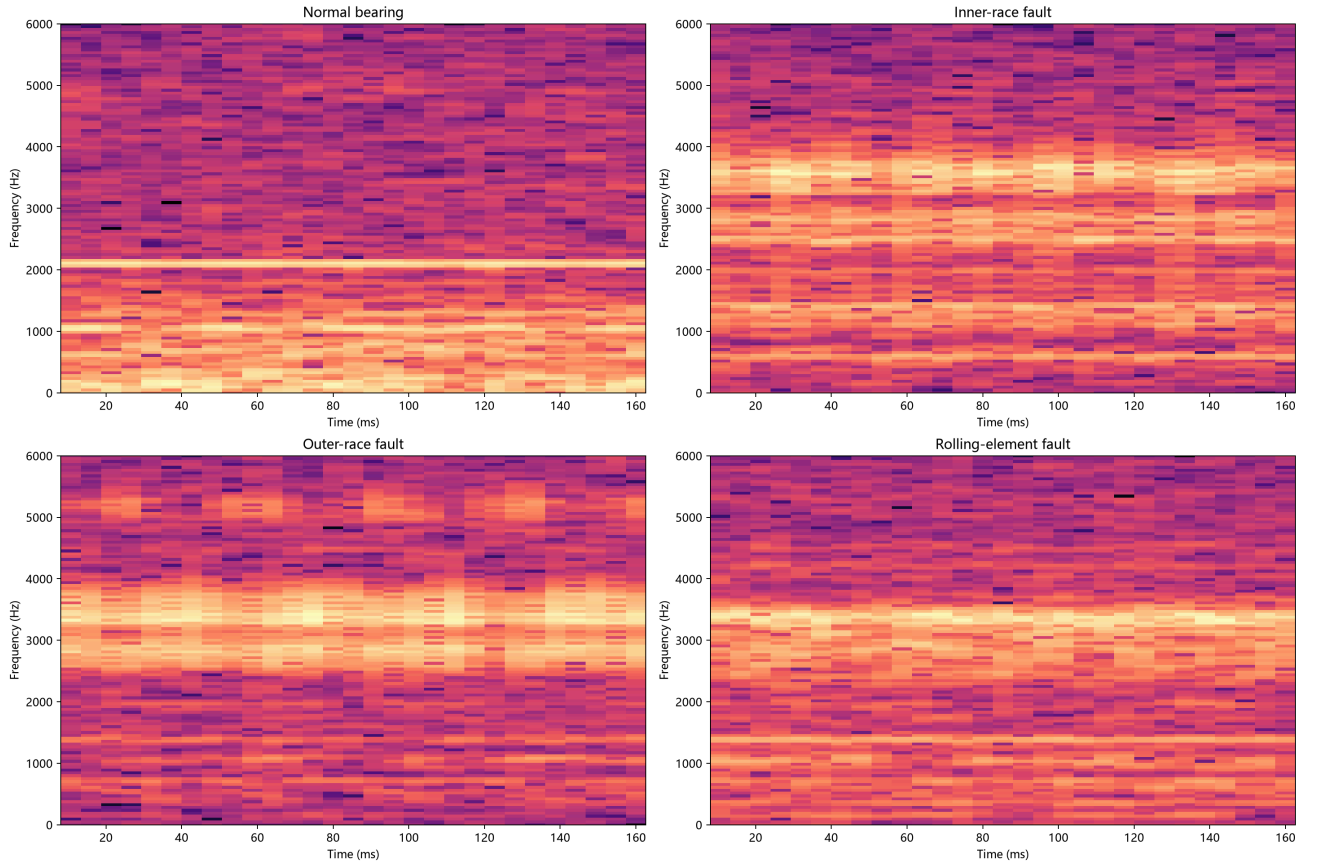


Figure 2. Representative 3 hp target-domain STFT spectrograms with 20 dB Gaussian noise

4.2. Training and evaluation protocol

Training uses AdamW for 20 epochs (learning rate 1×10^{-3} , weight decay 1×10^{-4} , batch size 32); CUDA is selected when available. Evaluation reports accuracy, macro-F1, and class-wise F1. Deployment reports parameters, model size, CPU latency, and throughput. ONNX Runtime measures latency over 200 warmed-up runs. The configuration uses 12 kHz sampling, 2,048-point segments, 1,024-point stride, a 256-point window, 192-point overlap, 64×64 input, and 20 dB target noise.

5. Results and discussion

5.1. Training behavior

The 20-epoch run reduces loss from 9.5843 to 0.1687, with a rapid early decline followed by stabilization. Three runs (seeds 2026–2028) produce the same rounded accuracy and macro-F1. A 10-epoch diagnostic is less stable (accuracies 0.9678, 0.7531, 0.5414), indicating that focal-weighted training needs sufficient optimization.

5.2. Unseen-condition diagnosis

Testing uses the unseen 2 hp condition. The reference run obtains 0.9969 accuracy and 0.9970 macro-F1; class-wise F1 is 0.9967 (normal), 1.0000 (inner-race), 1.0000 (outer-race), and 0.9915 (rolling-element). Table 1 summarizes the ablation. All five configurations reach 1.0000 after 20 epochs, so the split does not reveal numerical gains from DANN or focal loss. It instead confirms reproducibility and the need for a harder cross-domain benchmark; here, gains appear mainly in model size and deployment cost.

Table 1. Three-seed ablation results on the unseen 2 hp condition

Experiment	Accuracy (mean +/- SD)	Macro-F1 (mean +/- SD)
Conventional CNN (source-only)	1.0000 +/- 0.0000	1.0000 +/- 0.0000
Source-only MobileNet (CE)	1.0000 +/- 0.0000	1.0000 +/- 0.0000
MobileNet + target labels (CE)	1.0000 +/- 0.0000	1.0000 +/- 0.0000
MobileNet + DANN (CE)	1.0000 +/- 0.0000	1.0000 +/- 0.0000
LiteDANN (Focal + sampling)	1.0000 +/- 0.0000	1.0000 +/- 0.0000

5.3. Lightweight deployment

LiteDANN contains 37,478 parameters. FP32 and INT8 ONNX files are approximately 0.15 and 0.10 MB. On the desktop CPU, INT8 inference averages 2.13 ms and 468.5 samples/s over 200 runs. This is a CPU-based edge simulation; Raspberry Pi or gateway results may differ with hardware and operator support. Quantization reduces memory and integer cost, but speedup is platform-dependent [8, 11].

5.4. Limitations and reproducibility

Because target labels are limited, this is semi-supervised domain adaptation rather than fully unsupervised transfer. Artificial 20 dB noise may not capture factory interference, coupling, resonance, or sensor degradation. Latency was measured on a desktop CPU, not embedded hardware. Perfect rounded ablation scores also show that the CWRU split is too easy to isolate component effects. Future work should use file-level separation, varied severities, realistic noise, and an independent machine or dataset.

6. Conclusion

This paper presents LiteDANN, a lightweight semi-supervised domain-adaptation framework for bearing fault diagnosis on edge devices. The method combines a MobileNetV2-style feature extractor, a gradient reversal domain branch, focal loss, and weighted random sampling. The design addresses three practical constraints at the same time: vibration-distribution changes between operating loads, scarcity and imbalance of fault samples, and limited memory and computation on edge platforms.

The experimental pipeline starts from drive-end vibration signals in the CWRU dataset. Signals are divided into 2,048-point segments, transformed by STFT, resized to 64×64 , and standardized. The 0 hp condition supplies the source-domain data, the noisy and imbalanced 3 hp condition supplies target-domain adaptation data, and the unseen 2 hp condition is used for the final evaluation. In the recorded run, the combined loss decreases from 9.5843 to 0.1687 over 20 epochs. The resulting model achieves 99.69% accuracy and 99.70% macro-F1 on the unseen test condition. The inner-race and outer-race classes reach an F1-score of 1.0000, while the normal and ball classes reach 0.9967 and 0.9915, respectively.

The deployment results further show the value of the lightweight design. LiteDANN contains only 37,478 trainable parameters. ONNX conversion produces a portable inference representation, and INT8 quantization reduces the serialized model size to approximately 0.10 MB. On the current desktop CPU, the INT8 model reaches an average latency of 2.13 ms and a throughput of 468.5 samples per second. These figures support the feasibility of real-time inference under constrained storage and compute budgets. They do not, however, replace measurements on a real embedded target.

The added three-seed ablation confirms reproducibility, but it also shows that the present CWRU split is too easy to separate the contributions of DANN, Focal Loss, and sampling: every configuration reaches a rounded score of 1.0000 after 20 epochs. The standard CNN contains 406,406 parameters, compared with 37,478 for LiteDANN, so the lightweight design still provides a 90.8% parameter reduction even though accuracy is tied on this benchmark. Future work should use file-level and machine-level separation, multiple fault severities, and realistic industrial noise. The ONNX INT8 model should also be benchmarked on Raspberry Pi or an industrial edge gateway, with memory usage and energy consumption reported together with latency. These additions would turn the current proof of concept into a more discriminating and statistically grounded edge-diagnosis study.

References

- [1] Smith, W. A., & Randall, R. B. (2015). Rolling element bearing diagnostics using the Case Western Reserve University data: A benchmark study. *Mechanical Systems and Signal Processing*, 64-65, 100-131.
- [2] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521, 436-444.

- [3] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., et al. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.
- [4] Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345-1359.
- [5] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., et al. (2016). Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17, 2096-2030.
- [6] Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2020). Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2), 318-327.
- [7] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4510–4520.
- [8] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A. G., et al. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2704–2713.
- [9] Randall, R. B., & Antoni, J. (2011). Rolling element bearing diagnostics: A tutorial. *Mechanical Systems and Signal Processing*, 25(2), 485–520.
- [10] Allen, J. B. (1977). Short-term spectral analysis, synthesis, and modification by discrete Fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 25(3), 235–238.
- [11] Han, S., Mao, H., & Dally, W. J. (2016). Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. In *International Conference on Learning Representations*.