Quantizing Whisper: How Design Choices Affect ASR Performance

Arthur Söhler[1], Julian Irigoyen[2], Andreas Søeborg Kirkedal[3]
[1] Copenhagen Business School, Copenhagen, Denmark
[2] Danske Bank, Copenhagen, Denmark
[3] Jabra (GN Group), Copenhagen, Denmark
arthursoehler@gmail.com, jiri@danskebank.dk, askirkedal@jabra.com

Abstract

Large speech recognition models like OpenAI's Whisper achieve high accuracy but are difficult to deploy in resource-constrained environments due to their high memory and computational demands. This matters for low-resource and on-device settings, where compute and memory constraints often limit the practical use and evaluation of ASR systems. To address this, we present a unified, cross-library evaluation of post-training quantization (PTQ) on Whisper-small, comparing supported configurations across quantization scheme, method, granularity, and bit-width. Our study is based on four libraries—*PyTorch*, *Optimum-Quanto*, *HQQ*, and *bitsandbytes*. Experiments on *LibriSpeech test-clean* and *test-other* show that dynamic *int8* quantization with *Optimum-Quanto* offers the best trade-off, reducing model size by 57% while lowering Word Error Rate below the baseline. Additional experiments on Whisper-base and Whisper-tiny confirm these trends, though with more pronounced degradation at lower bit-widths. Static quantization performed worse, likely due to the absence of efficient low-bit implementations for operations such as LayerNorm and Softmax. More aggressive formats (e.g., *nf4*, *int3*) achieved up to 71% compression at the cost of accuracy in acoustically challenging conditions. Our results demonstrate that carefully chosen PTQ methods can substantially reduce model size and inference cost without retraining, enabling efficient deployment of Whisper on constrained hardware.

Keywords: automatic speech recognition, neural network quantization, model compression

1. Introduction

Automatic speech recognition (ASR) has advanced rapidly with large-scale Transformer models (Vaswani et al., 2017) such as the Whisper family (Radford et al., 2023), which deliver state-of-the-art transcription accuracy across diverse languages and domains. However, this accuracy comes at a cost: models with hundreds of millions of parameters are difficult to deploy on edge devices, embedded systems, or latency-sensitive applications (Gholami et al., 2021; Wei et al., 2024). This challenge is especially pronounced in low-resource and on-device settings, where limited compute budgets and strict memory constraints can prevent both deployment and systematic evaluation. Post-training compression methods such as post-training quantization (PTQ) (Gholami et al., 2021; Nagel et al., 2021) therefore offer a practical path to making strong ASR models usable in settings where retraining and specialized hardware are not feasible.

Model compression techniques address this challenge by reducing computational cost in deep neural networks (Vanhoucke et al., 2011). Among them, *quantization* has emerged as one of the most practical and hardware-friendly approaches. By mapping 32-bit floating point weights and activations to lower-precision formats, quantization can shrink memory footprint and accelerate inference (Gray and Neuhoff, 1998). The choice of method—such as PTQ, which requires no retraining, or quantization-aware training (QAT), which incorporates quantization effects during training—determines the balance between efficiency and accuracy. While QAT often yields superior results under aggressive compression, PTQ is attractive for its speed and ease of deployment (Wu et al., 2020).

Recent years have seen rapid progress in low-bit quantization, including sub-8-bit formats (e.g., *int4*, *nf4*) and mixed-precision strategies designed to handle outlier activations (Dettmers et al., 2022; Shen et al., 2024). These methods have been evaluated extensively in computer vision and large language models, but their impact on ASR—and particularly on large-scale architectures like Whisper—remains underexplored.

In this paper, we take Whisper-small as a case study to evaluate post-training quantization across multiple libraries, methods, schemes, and bit-widths. We present a comparative study of Whisper PTQ in which all configurations are evaluated under a unified measurement protocol on both CPU and GPU, highlighting practical trade-offs between latency, accuracy, and compression across devices and tools. Our goal is to uncover how specific PTQ design choices translate into deployment behavior (efficiency and robustness) in realistic scenarios, where library support and overheads can materially affect outcomes. We additionally confirm our findings across model sizes by performing experiments on Whisper-base and Whisper-tiny, the two smallest variants of the Whisper family.

Proceedings of SPEAKABLE @ LREC 2026, pages 39–46

Our contributions are:

(1) Unified evaluation framework: Systematic cross-library comparison of PTQ for Whisper under a controlled, unified evaluation protocol, enabling direct comparison of library-specific trade-offs.

(2) Comparative analysis of PTQ design choices: We compare the effects of quantization scheme (dynamic vs. static), method (symmetric vs. asymmetric), granularity (per-tensor vs. per-channel), and bit-width (*int8* to *int3*/*nf4*) across representative libraries.

(3) Device-specific deployment insights: Parallel evaluation on CPU and GPU revealing different optimal configurations per device and acoustic condition (clean vs. acoustically challenging).

(4) Cross-model validation: Verification that the main findings hold across Whisper-small, Whisper-base, and Whisper-tiny, demonstrating robustness of deployment conclusions across model sizes.

2. Related Work

While its theoretical foundations date back decades (Gray and Neuhoff, 1998), modern neural network quantization has evolved rapidly, with several surveys providing systematic overviews (Gholami et al., 2021; Nagel et al., 2021; Wei et al., 2024). These works classify methods into post-training quantization and quantization-aware training, and outline factors such as bit-width, quantization granularity, and calibration strategies that strongly influence performance.

In Transformer-based architectures, Kim et al. (2021) proposed I-BERT, an integer-only Transformer that approximates nonlinear operations for complete deployment on integer hardware. Kim et al. (2022) further introduced a zero-shot static quantization approach for ASR models such as Conformer and QuartzNet, achieving $4\times$ compression without real training data by synthesizing calibration inputs. Wu et al. (2020) demonstrated that with per-channel weight quantization and entropy-based activation calibration, PTQ can maintain accuracy within 1% of full precision.

Recent efforts have focused on stabilizing quantization for large Transformer models by handling outlier activations. Dettmers et al. (2022) introduced LLM.int8(), a mixed-precision method that preserves high precision for outlier dimensions, enabling robust 8-bit inference for models up to 175B parameters. Shen et al. (2024) explored low-precision floating-point formats such as *fp8*, reporting improved accuracy and flexibility compared to *int8* in outlier-heavy models.

Prior work has also examined Whisper quantization in specific toolchains and limited method sets. For example, Andreyev (2025) evaluates different bit-widths using *whispercpp* on *LibriSpeech* corpus (Panayotov and Povey, 2015) and reports latency and accuracy trade-offs for edge deployment scenarios. In contrast, our work focuses on a controlled, cross-library comparison spanning multiple PTQ libraries and design choices (scheme, granularity, and bit-width) under one measurement protocol across CPU and GPU.

More recent work has explored low-bit quantization specifically for speech and audio models. Feng et al. (2025) propose Edge-ASR, a comprehensive benchmark of post-training quantization methods applied to Whisper and other edge-ASR models. Kang and Kim (2025) propose GenPTQ, a mixed-precision post-training quantization method that performs efficient layer-wise bit allocation based on gradient-driven sensitivity analysis, achieving strong compression with minimal WER degradation. Beyond speech recognition, Khandelwal and Fuentes (2025) study post-training quantization for audio diffusion transformers, highlighting challenges related to activation variability and outliers in transformer-based audio models. These works primarily focus on model-specific optimization strategies, whereas our study provides a cross-library evaluation of quantization approaches under a unified experimental framework.

While most prior work has focused on vision and text models, the techniques—especially sub-8-bit quantization, mixed-precision schemes, and outlier handling—are directly relevant to ASR. However, to our knowledge, there is no comparative evaluation of quantization libraries that jointly contrasts (i) activation method (dynamic vs. static), (ii) granularity (per-channel vs. per-tensor), (iii) weight-only vs. weight+activation quantization, and (iv) multiple bit-widths, while evaluating accuracy, latency, and compression on both CPU and GPU under a single protocol. This work addresses that gap by assessing PTQ strategies and their trade-offs for Whisper-small, Whisper-base, and Whisper-tiny.

3. Background

Quantization reduces the memory and computational cost of neural networks by mapping high-precision values (typically 32-bit floating point) to lower-precision representations. In practice, this means storing weights and activations in compact formats such as *int8* or *fp8*, with memory usage scaling approximately as $b/32$ relative to *fp32*, yielding a compression ratio of $32/b$ where b is the bit-width (Gholami et al., 2021; Nagel et al., 2021).

Formally, a quantizer can be written as

$$Q(x) = \text{clip}\Big(\text{round}\big(\tfrac{x}{s}\big), q_{\min}, q_{\max}\Big), \qquad (1)$$

where s is a scaling factor, and $[q_{\min}, q_{\max}]$ defines the representable integer range for a given bit-width.

Subset	Hours	Minutes/ speaker	Female speakers	Male speakers	Total speakers
test-clean	5.4	8	20	20	40
test-other	5.1	10	17	16	33

Table 1: Dataset statistics for *test-clean* and *test-other*.

The effectiveness of this mapping depends not only on the chosen bit-width, but also on how scales are applied across the network.

This is captured by the notion of *granularity*. In *per-tensor* quantization, a single scale is shared across an entire tensor, which is efficient but may distort values when distributions vary widely. *Per-channel* quantization instead assigns a scale to each output channel, better capturing local statistics and improving accuracy in deep Transformers, though at higher storage and compute cost (Gholami et al., 2021). A middle ground is *per-group* quantization, which clusters channels into groups with shared scales to balance efficiency and accuracy (Yao et al., 2022).

Beyond granularity, quantization can be applied either post-training (PTQ), which avoids retraining but may reduce accuracy at low bit-widths, or during training (QAT), which improves robustness at the cost of additional training (Han et al., 2015; Gholami et al., 2021).

Additionally, quantization methods differ on how they distribute values within the quantized range. Symmetric quantization centers values around zero, whereas asymmetric quantization shifts the zero-point to better match non-zero-centered distributions of weights (Nagel et al., 2021).

Finally, schemes differ in how scaling factors and zero-points are determined. *Static quantization* fixes them in advance using calibration data, while *dynamic quantization* computes them at runtime.

These design choices become especially important for Transformer-based ASR models such as Whisper-small. Operations like *LayerNorm*, *Softmax*, and *GELU* are highly sensitive to reduced precision, and activations often exhibit heavy-tailed distributions (Dettmers et al., 2022; Shen et al., 2024). As a result, quantization can compromise robustness, underscoring the need for careful selection of quantization strategies in deployment.

4. Data

We conduct our experiments on the English-language subsets *test-clean* and *test-other* of the *LibriSpeech* corpus (Panayotov and Povey, 2015). Dataset statistics are shown in Table 1.

These test sets have been used to evaluate ASR systems for many years. *test-clean* represents an easy evaluation task, whereas *test-other* is a more challenging dataset. The original *LibriSpeech* data consists of read-aloud books and was divided into a *clean* and an *other* partition based on Word Error Rate (WER) scores produced by a hybrid ASR system with an acoustic model trained on a subset of the Wall Street Journal corpus and a bigram LM estimated from the text of the respective books. The speakers were ranked according to WER and the data partitioned into two sets of roughly equal sizes. *test-clean* consists of randomly selected speakers from the *clean* partition. *test-other* specifically consists of speakers from the 3rd quartile according to WER. This sampling is intended to create a more challenging test dataset. Gender balance is ensured at the speaker level (Panayotov et al., 2015).

5. Methods

We evaluate PTQ on Whisper-small, a 244M-parameter Transformer-based ASR model pretrained for multilingual and multitask speech recognition (Radford et al., 2023).

Depending on library support, we apply quantization across a range of bit-widths (*int8*, *int4*, *int3*, *nf4*, and *fp8*) and compare four widely used PTQ libraries: *PyTorch*, offering native dynamic *int8* quantization on CPU; *Optimum-Quanto* (hereafter *Quanto*), supporting both integer and low-precision floating formats across CPU, GPU, and MPS backends; *HQQ*, a quantization library based on half-quadratic optimization with configurable group sizes; and *bitsandbytes (BNB)*, which provides GPU-only implementations of normalized formats such as *nf4* with optional double quantization. Together, these libraries cover a diverse spectrum of formats, methods, and quantization schemes, making them representative of the practical choices available to users when designing model compression pipelines.

Performance is evaluated along three dimensions. Accuracy is measured using WER and Character Error Rate (CER). Efficiency is captured through the Real-Time Factor (RTF), quantifying inference speed relative to audio duration. Finally, we report model size reduction relative to the full-precision *fp32* baseline. To ensure comparability, all configurations were evaluated under the same preprocessing pipeline, dataset splits, batch size, decoding procedure, hardware selection, and timing protocol. RTF was computed as total timed generation time divided by total audio duration over the evaluation set. Because all measurements were collected on the HPC infrastructure described in Section 10, the reported RTF values should be interpreted primarily as relative comparisons across configurations under a fixed setup, rather than as direct estimates of edge-device latency. Inference time was measured only around the model genera-

Device/Method	WER_c	WER_o	CER_c	CER_o	RTF	Size Red.
CPU						
Baseline (fp32)	3.48	11.88	1.02	3.62	0.121	–
PyTorch int8 (dyn.)	3.72	13.67	1.11	4.21	0.077	57%
HQQ int4 (dyn.)	3.52	14.09	1.09	4.38	0.155	69%
Quanto int8/fp8 (stat.)	5.95	15.92	1.83	5.03	0.169	57%
GPU						
Baseline (fp32)	3.48	11.88	1.02	3.62	0.006	–
Quanto int8 (dyn.)	3.41	10.65	0.97	3.29	0.008	57%
BNB nf4 (dyn.)	3.54	13.49	1.05	4.05	0.008	70%
HQQ int3 (dyn.)	4.11	12.93	1.22	3.77	0.019	71%

Table 2: Selected best-performing dynamic (dyn.) and static (stat.) quantization configurations for Whisper-small on *LibriSpeech test-clean* (c) and *test-other* (o).

Device/Method	WER_c	WER_o	CER_c	CER_o	RTF	Size Red.
Whisper-base						
Baseline (fp32)	5.08	12.87	1.93	6.76	0.0022	–
Quanto int8 (dyn.)	5.10	14.72	1.93	7.60	0.0037	36.2%
BNB nf4 (dyn.)	6.05	18.09	2.31	10.16	0.0036	52.1%
Whisper-tiny						
Baseline (fp32)	7.60	23.69	2.98	12.84	0.0015	–
Quanto int8 (dyn.)	7.64	24.64	2.99	13.24	0.0025	19.3%
BNB nf4 (dyn.)	11.16	32.19	4.78	20.20	0.0026	37.5%

Table 3: Quantization results for Whisper-base and Whisper-tiny using the strongest dynamic (dyn.) configurations on *LibriSpeech test-clean* (c) and *test-other* (o).

tion step, with CUDA synchronization on GPU and warm-up runs before timed evaluation. Audio inputs were converted to model input features using the standard WhisperProcessor from the Transformers library. In the adopted Transformers implementation, Whisper used its default fixed 30 s front end; accordingly, shorter utterances were zero-padded to the 30 s window. The reported RTF values therefore reflect this fixed-window inference setup rather than variable-length audio processing. Relative RTF differences nevertheless remain directly comparable across quantization settings because all configurations were evaluated under the same protocol.

For static quantization, calibration used a randomly selected 10% subset of the full evaluation data, processed with the same feature-extraction pipeline as the test data.

While the main study focuses on Whisper-small, we also run a limited follow-up study on Whisper-tiny and Whisper-base under the same dataset, settings, and measurement procedure. The cross-model follow-up is intended as a validation of the main deployment trends, not as a second exhaustive benchmark. We therefore restrict Whisper-tiny and Whisper-base to the strongest dynamic configurations from the Whisper-small study. Static quantization is omitted in this follow-up because it underperformed on Whisper-small and would substantially increase experimental volume without changing the main practical conclusion.

6. Results

6.1. Quantizing Whisper-small

Table 2 summarizes the best-performing quantized models relative to the full-precision baseline.

On **CPU**, *PyTorch* dynamic *int8* delivered the fastest inference (RTF 0.077; 36.4% faster than the 0.121 baseline) with only a small accuracy drop. *HQQ* dynamic *int4* preserved accuracy on clean speech while achieving the largest compression (69%). In contrast, static *int8*/*fp8* quantization with *Quanto* degraded performance substantially, confirming that Whisper's architecture is ill-suited to static quantization.

On **GPU**, *Quanto* dynamic *int8* not only matched but slightly outperformed the baseline on the more challenging *test-other* split (WER_{other} = 10.65, CER_{other} = 3.29), while reducing model size by 57%. *BNB nf4* offered a 70% size reduction with minimal accuracy loss on *test-clean*, but suffered on *test-other*. *HQQ int3* achieved the smallest size (71% reduction) but at the cost of higher error rates.

Overall, dynamic quantization consistently outperformed static quantization in both accuracy and speed. *int8* proved the most reliable setting across devices, while more aggressive formats (*nf4*, *int3*) enabled extreme compression but compromised robustness, particularly under acoustically challenging conditions.

6.2. Scaling Across Whisper Model Sizes

To assess whether the main conclusions are specific to Whisper-small or stable within the Whisper model family, we evaluate Whisper-tiny and Whisper-base on *LibriSpeech* splits using the strongest quantization configurations from the Whisper-small study. Table 3 shows that the overall trends remain consistent across model sizes: dynamic *int8* remains the most robust operating point, while more aggressive formats trade robustness (especially on *test-other*) for additional compression.

7. Discussion

7.1. Trade-offs Between Different Quantization Methods

On CPUs, *PyTorch* dynamic *int8* consistently achieved the fastest inference. Its advantage likely stems from using a per-tensor asymmetric scheme,

which applies a single scale across an entire tensor. This approach simplifies computation and reduces the overhead of quantization and dequantization, explaining the strong runtime performance. The trade-off, however, is lower representational flexibility with quantization parameters only calculated per-tensor, which contributed to weaker robustness on the acoustically diverse *test-other* dataset. However, with a 57% size reduction, it still offers clear advantages over the baseline.

On the GPU, *Quanto* dynamic *int8* took a different approach. It prioritizes accuracy over speed. It was slower but outperformed the *fp32* baseline on WER and CER on both *test-clean* and *test-other*, with the most striking improvement seen under acoustically challenging conditions. One plausible explanation is its symmetric, per-channel scheme, which aligns well with Whisper's near-zero-centered weight distributions and assigns independent scales to each channel. This granularity can better preserve fine-grained variation across channels, yielding higher accuracy but at the cost of additional computational overhead. Similar to the *PyTorch* model, this model comes with a 57% size reduction, a strong compression relative to the baseline.

Libraries enabling more aggressive formats, such as *HQQ* (*int4*, *int3*) and *BNB* (*nf4*), pushed compression further—up to 70%—but consistently degraded robustness, especially in acoustically challenging environments. These results highlight that precision below 8-bit remains less reliable in real-world ASR applications.

Taken together, these comparisons show that library differences are largely driven by the schemes and bit-widths they implement. Per-tensor quantization favors efficiency, making *PyTorch int8* attractive when low latency is the main deployment priority. Per-channel quantization favors accuracy, as shown by *Quanto int8*, which is more suitable when robustness is critical. More aggressive low-bit formats are best reserved for deployments where extreme compression outweighs the need for reliability.

The additional results on Whisper-base and Whisper-tiny (Table 3) follow the same qualitative pattern: dynamic *int8* remains the most robust operating point, while *nf4* increases compression but incurs a larger error increase—especially on *test-other*. This suggests that the robustness–compression trade-off observed for Whisper-small is stable across smaller Whisper variants, even though absolute WER/CER differs by model size.

7.2. Dynamic vs. Static Quantization

In theory, static quantization should reduce runtime overhead by fixing scales in advance, trading a small loss in accuracy for faster inference. Surprisingly, in Whisper-small we observed the opposite: static quantization was both slower and less accurate.

One possible explanation is that operations such as *LayerNorm* and *Softmax* lack efficient low-bit implementations, forcing repeated dequantization that cancels the expected speed gains of static quantization. On top of this, as expected, fixed calibration scales limited robustness under shifting distributions, especially in acoustically challenging conditions.

By contrast, dynamic quantization adapted better at runtime, preserving accuracy and delivering faster inference. This makes dynamic quantization the most reliable choice in our evaluation for Whisper-small, despite theoretical expectations to the contrary.

7.3. Clean vs. Acoustically Challenging Speech

Across nearly all configurations, quantized models suffered larger accuracy drops on *test-other* than on *test-clean*, indicating reduced robustness under the more challenging conditions represented by the *test-other* split. As shown in Figure 1, lower-bit formats such as *BNB nf4* and *HQQ int3* showed the steepest increases in WER, underlining how aggressive compression disproportionately reduces robustness.

Interestingly, *Quanto* dynamic *int8* was an exception: it not only matched the baseline on *test-clean* but also outperformed it on *test-other*. This suggests that under certain conditions, quantization can act as a form of regularization, stabilizing predictions in acoustically challenging environments. However, this effect appears limited to moderate

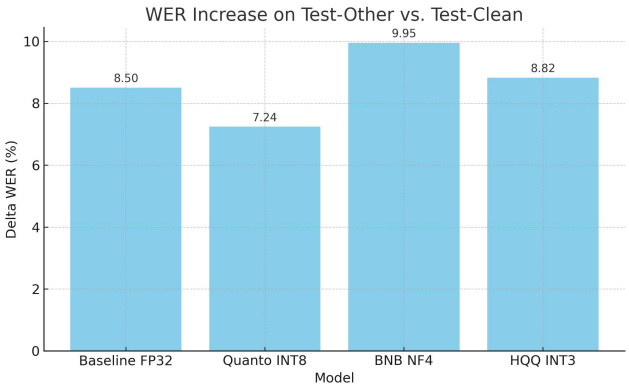

Figure 1: WER increase on *test-other* relative to *test-clean* for selected quantized models of Whisper-small. Lower-bit-width configurations (*nf4*, *int3*) show larger deltas, highlighting the trade-off between compression and robustness.

Priority	Best Model	Trade-off
Fastest Inference	PyTorch Dynamic int8	Higher WER compared to baseline
Best Accuracy	Quanto Dynamic int8	Higher RTF compared to baseline
Best Size Reduction	HQQ Dynamic int3	Higher WER and RTF compared to baseline

Table 4: Comparative Summary of Quantization Methods - Optimal Approaches Based on Deployment Priorities

settings (e.g., *int8*); once precision drops further, errors compound rapidly under adverse acoustic conditions. A similar robustness gap between *test-clean* and *test-other* is also evident for Whisper-base and Whisper-tiny, where *nf4* degrades more strongly than *int8* on *test-other*.

For deployment, this highlights that 8-bit precision is a safe minimum in real-world scenarios, while lower-bit formats should be applied selectively and only when acoustic variability is less of a concern.

7.4. Deployment Implications

Our results point to clear deployment strategies for Whisper-small and its variants under different constraints. Table 4 summarizes these strategies as rule-of-thumb guidance derived from our evaluation protocol. On CPUs, *PyTorch* dynamic *int8* offers the best speed–accuracy trade-off, while *HQQ* dynamic *int4* offers stronger compression when memory is the tighter constraint. On GPUs, *Quanto* dynamic *int8* is the most robust choice, delivering a 57% size reduction while preserving or even improving accuracy, including in acoustically challenging conditions. More aggressive formats such as *HQQ int3* or *BNB nf4* are suitable only in memory-constrained scenarios where some accuracy loss is acceptable, and ideally applied selectively to less critical layers. Finally, in acoustically diverse environments, at least 8-bit precision should be maintained for sensitive components such as attention and *LayerNorm* to avoid robustness degradation. These guidelines emphasize that quantization choices must be tailored not only to model architecture, but also to deployment context, device, and acoustic variability.

8. Conclusion

This study evaluated post-training quantization of Whisper-small across four libraries and multiple bit-widths, providing a controlled cross-library comparison of supported scheme, method, and granularity choices. On *LibriSpeech test-clean* and *test-other*, dynamic quantization consistently outperforms static quantization in both accuracy and inference speed for this architecture. *Quanto* dynamic *int8* emerged as the best overall configuration for GPU deployment, achieving 57% model-size reduction while matching baseline accuracy on *test-clean* and even exceeding it on *test-other*. On CPU, *PyTorch* dynamic *int8* delivered the fastest inference in our experiments, while *HQQ* dynamic *int4* offers a strong compromise when memory is limited.

From a high-level perspective, our results indicate that within this model family and library-supported configurations, dynamic quantization was more reliable than static quantization. Furthermore, the most accurate results were obtained with configurations using symmetric, per-channel quantization rather than asymmetric, per-tensor approaches, especially under acoustically variable conditions. In practice, *int8* is the most reliable precision across devices; more aggressive formats (e.g., *nf4*, *int3*) provide larger compression but degrade robustness on *test-other*. We also observe a mild regularization effect: dynamic *int8* can match or even exceed *fp32* on acoustically challenging speech, but this benefit does not persist below 8-bit precision. Taken together, the *scheme*, *method*, and *granularity* materially shape the accuracy–speed–size trade-off and should be chosen to match device constraints and expected acoustic conditions. Ultimately, our findings show that thoughtful quantization design enables deployment of more advanced speech recognition models in resource-constrained environments.

9. Limitations and Future Work

First, this work is designed as a controlled study to compare cross-library PTQ effects under a shared evaluation protocol. Because the study is constrained by library support, not all combinations of quantization scheme, method, granularity, and bit-width could be evaluated; the comparison is therefore structured rather than exhaustive. We restrict evaluation to the *LibriSpeech test-clean* and *test-other* splits, which provide a reproducible benchmark but do not capture the full diversity of domains, accents, and spontaneous speech encountered in practice. Extending the evaluation to broader domains, including multilingual data, is an important next step.

Second, we focus on PTQ to reflect scenarios where retraining is infeasible; quantization-aware training may achieve higher accuracy at more aggressive bit-widths. In addition, ONNX Runtime is excluded because it relies on a different execution stack involving model export and graph-level optimizations, introducing additional variables that make its quantization and runtime behavior not directly comparable to the native-library workflows

evaluated in this study. Both represent complementary directions for future work.

Third, our analysis targets the smaller Whisper variants as representative Transformer-based ASR models. Larger variants such as Whisper-medium and Whisper-large are excluded from this study, as our focus is on smaller models that are most relevant for deployment in resource-constrained environments. Results may differ for the larger Whisper models or for other ASR architectures.

Future work will expand evaluation to additional datasets and model families, and investigate mixed-precision and layer-wise strategies that apply aggressive quantization selectively while preserving accuracy in sensitive components.

10. Acknowledgements

We thank Jabra and GN Group for supporting this research. Computational experiments were performed on the Danish e-Infrastructure Consortium (DeiC) National HPC facilities, utilizing Lenovo ThinkSystem SR675 V3 nodes equipped with dual AMD EPYC 9454 processors (2.75 GHz, 192 vCPUs total), 768 GB DDR5-4800 memory, and four NVIDIA Hopper H100-SXM5 GPUs (80 GB HBM3) per node.

11. Bibliographical References

Allison Andreyev. 2025. Quantization for OpenAI's Whisper Models: A Comparative Analysis. *arXiv preprint arXiv:2503.09905*.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. GPT3.int8(): 8-bit Matrix Multiplication for Transformers at Scale. In *Advances in Neural Information Processing Systems*, volume 35, pages 30318–30332. Curran Associates, Inc.

Chen Feng, Yicheng Lin, Shaojie Zhuo, Chenzheng Su, Ramchalam Kinattinkara Ramakrishnan, Zhaocong Yuan, and Xiaopeng Zhang. 2025. Edge-ASR: Towards Low-Bit Quantization of Automatic Speech Recognition Models. *arXiv preprint arXiv:2507.07877*.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. 2021. A Survey of Quantization Methods for Efficient Neural Network Inference. *arXiv preprint arXiv:2103.13630*.

R.M. Gray and D.L. Neuhoff. 1998. Quantization. *IEEE Transactions on Information Theory*, 44(6):2325–2383.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both Weights and Connections for Efficient Neural Networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Beom Jin Kang and Hyun Kim. 2025. GenPTQ: Green Post-Training Quantization for Large-Scale ASR Models with Mixed-Precision Bit Allocation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 10704–10718, Suzhou, China. Association for Computational Linguistics.

Tanmay Khandelwal and Magdalena Fuentes. 2025. Post-training quantization for audio diffusion transformers. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, WASPAA 2025, Tahoe City, CA, USA, October 12-15, 2025*, pages 1–5. IEEE.

Sehoon Kim, Amir Gholami, Zhewei Yao, Nicholas Lee, Patrick Wang, Aniruddha Nrusimha, Bohan Zhai, Tianren Gao, Michael W. Mahoney, and Kurt Keutzer. 2022. Integer-Only Zero-Shot Quantization for Efficient Speech Recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4288–4292.

Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. 2021. I-BERT: Integer-only BERT Quantization. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5506–5518. PMLR.

Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart van Baalen, and Tijmen Blankevoort. 2021. A White Paper on Neural Network Quantization. *arXiv preprint arXiv:2106.08295*.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. LibriSpeech: an ASR corpus based on public domain audio books. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5206–5210. IEEE.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine Mcleavey, and Ilya Sutskever. 2023. Robust Speech Recognition via Large-Scale Weak Supervision. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 28492–28518. PMLR.

Haihao Shen, Naveen Mellempudi, Xin He, Qun Gao, Chang Wang, and Mengni Wang. 2024.

Efficient Post-training Quantization with FP8 Formats. In *Proceedings of Machine Learning and Systems*, volume 6, pages 483–498.

Vincent Vanhoucke, Andrew Senior, and Mark Z. Mao. 2011. Improving the speed of neural networks on CPUs. In *Deep Learning and Unsupervised Feature Learning Workshop, NIPS 2011*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Lu Wei, Zhong Ma, Chaojie Yang, and Qin Yao. 2024. Advances in the Neural Network Quantization: A Comprehensive Review. *Applied Sciences*, 14(17).

Hao Wu, Patrick Judd, Xiaojie Zhang, Mikhail Isaev, and Paulius Micikevicius. 2020. Integer Quantization for Deep Learning Inference: Principles and Empirical Evaluation. *arXiv preprint arXiv:2004.09602*.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. 2022. ZeroQuant: Efficient and Affordable Post-Training Quantization for Large-Scale Transformers. In *Advances in Neural Information Processing Systems*, volume 35, pages 27168–27183. Curran Associates, Inc.

12. Language Resource References

Panayotov, Vassil and Povey, Daniel. 2015. *LibriSpeech ASR corpus*. OpenSLR. PID https://www.openslr.org/12.