Efficient Post-training Quantization with FP8 Formats

Haihao Shen [1] **Naveen Mellempudi** [2 *] **Xin He** [1] **Qun Gao** [1] **Chang Wang** [1] **Mengni Wang** [1]

Abstract

Recent advances in deep learning methods such as LLMs and Diffusion models have created a need for improved quantization methods that can meet the computational demands of these modern architectures while maintaining accuracy. Towards this goal, we study the advantages of FP8 data formats for post-training quantization across 75 unique network architectures covering a wide range of tasks, including machine translation, language modeling, text generation, image classification, generation, and segmentation. We examine three different FP8 representations (E5M2, E4M3, and E3M4) to study the effects of varying degrees of trade-off between dynamic range and precision on model accuracy. Based on our extensive study, we developed a quantization workflow that generalizes across different network architectures. Our empirical results show that FP8 formats outperform INT8 in multiple aspects, including workload coverage (92.64% vs. 65.87%), model accuracy and suitability for a broader range of operations. Furthermore, our findings suggest that E4M3 is better suited for NLP models, whereas E3M4 performs marginally better than E4M3 on computer vision tasks.

1 Introduction

Quantization is the process of reducing the numeric precision of weights and activations of a neural network to lower the computation costs of inference. INT8 quantization (Vanhoucke et al., 2011; Han et al., 2015a) is the most widely-accepted choice today due to its ability to deliver high inference performance on modern deep learning hardware while maintaining reasonable model accuracy. It has been particularly effective for computer vision tasks such as object detection and image classification, and has been widely deployed in production both at the data center scale and on resource-constrained edge devices. However, INT8 presents several challenges that arise due to its limited dynamic range. Several quantization techniques have been developed to address these challenges. For example, asymmetric quantization (Jacob et al., 2018; Krishnamoorthi, 2018; Bhalgat et al., 2020) allocates different numbers of bits for the positive and negative ranges with a non-zero offset, to better represent the distribution of the original values. Non-uniform quantization methods (Miyashita et al., 2016; Zhou et al., 2017; Cai et al., 2017; Fang et al., 2020; Li et al., 2020) attempt to assign more precision to the parts of the data that are deemed more important to reduce quantization errors. Methods that use per-group (Zhou et al., 2016; Mellempudi et al., 2017) or per-channel (Jacob et al., 2018; Krishnamoorthi, 2018) scaling extend the effective dynamic range by using independent scaling factor for each selected group of elements. The limited dynamic range of INT8 also results in poor representation of outliers that are typically found in activations. This is especially prevalent in Large Language Models (LLMs), where outliers are significantly larger when compared to the rest of the activations. Most common approach for handling outliers is to clip them using threshold values that are either obtained through calibration (Sung et al., 2015; Zhao et al., 2019b) or learned during training (Bhalgat et al., 2020; Choi et al., 2018; Esser et al., 2020; Zhang et al., 2018a). More recently (Wei et al., 2022; Xiao et al., 2022) have proposed applying mathematical transformations to redistribute the magnitude of outliers between weights and activation tensors to minimize their impact. Despite these advancements, INT8 methods remain ineffective for a wide range of language modeling tasks, where the presence of LayerNorm was shown to amplify the occurrence of outliers (Wei et al., 2022). Therefore, a significant percentage of these workloads falls back to using higher precision to preserve model accuracy.

This paper argues that 8-bit floating-point (FP8) formats are an efficient and more productive alternative to INT8 for deep neural network quantization. We evaluated three different representations (E5M2, E4M3, and E3M4) that offer varying degrees of trade-off between dynamic range and precision. Table 1 shows the details of the binary format and special value encoding. The study focused on the benefits of FP8 formats for post-training quantization as the preferred approach used in production. We developed quantization workflows that generalized across different network

[*] The Work was done at Intel. [1]Intel Corporation, Shanghai, China. [2]AMD, Austin, Texas, United States. Correspondence to: Haihao Shen <haihao.shen@intel.com>.

Proceedings of the 5th MLSys Conference, Santa Clara, CA, USA, 2024. Copyright 2024 by the author(s).

Table 1. FP8 binary formats: The E$\underline{e}$M$\underline{m}$ notation represents bit allocation for *Exponent (e)* and *Mantissa (m)* respectively. The formats support a *sign-bit* and an implicit leading bit in the mantissa. E5M2 follows IEEE-like encoding rules, while E4M3 and E3M4 use extended encoding to reclaim ±Infinity for useful encoding, a unique bit-sequence of *all-ones* represents a NaN.

	E5M2	E4M3	E3M4
EXPONENT BIAS (b)	15	7	3
MAX VALUE	57344.0	448.0	30.0
MIN VALUE	1.5×10^{-5}	1.9×10^{-3}	1.5×10^{-2}
SUBNORMALS	YES	YES	YES
NANS	ALL	SINGLE	SINGLE
INFINITY	YES	NO	NO

architectures, and conducted experiments on 75 networks that cover a wide range of application domains. Our results show that FP8 formats overall provide higher accuracy, better workload coverage compared to INT8 (92.64% vs. 65.87%) and can handle more operations such as LayerNorm and BatchNorm. The data also suggests that E4M3 is better suited for a broad range of NLP models with a coverage of 96.32% compared to E3M4 (92.11%), while E3M4 performs slightly better on computer vision models with 78.95% coverage compared to E4M3 (73.68%). Our contributions are as follows:

- Propose a unified and scalable FP8 quantization flow that works across application domains and different model sizes. To the best of our knowledge, our work is the first to study this problem across 200+ tasks and 75+ models demonstrating the scalability of our approach.
- Demonstrate the advantages of FP8 formats over INT8, in terms of workload coverage, model accuracy and suitability for a broader range of operations. Our work is also the first study to showcase accuracy-driven automatic model tuning for quantization.
- Suggest that E4M3 is better suited for NLP models, whereas E3M4 performs marginally better than E4M3 on computer vision tasks.

1.1 Related Work

There is a growing body of research is studying the use of 8-bit floating-point formats to accelerate deep learning training and inference tasks. Initial studies by (Wang et al., 2018) and (Mellempudi et al., 2019) focused on the E5M2 format for training tasks due to its wider dynamic range which is necessary for representing gradient values. (Sun et al., 2019) subsequently proposed using a combination of two binary formats, E5M2 and E4M3, for training and extended their research to include inference tasks. They also suggested using an exponent bias to shift the numeric range of E4M3 format for handling outliers in activations. Later studies by (Noune et al., 2022) and (Kuzmin et al., 2022) have extended this scope to include variable exponent bias and formats with fewer exponent bits, such as E3M4 and E2M5. More recently, (Micikevicius et al., 2022) presented a generalized training method that employs per-tensor scaling using E5M2 and E4M3 formats. They also extended the inference studies to cover large language models such as GPT-3 (6.7B).

The rest of this paper is organized as follows. Section 2 discusses the advantages of 8-bit floating point representation in handling outliers. Section .3 introduces the quantization workflow and components of a standard, extended quantization scheme and a framework for tuning model performance. Section 4 outlines the experimental setup, presents accuracy results, and offers discussion on performance tuning. Section 5 presents the conclusions and future work.

2 BACKGROUND

FP8 Value Distribution and Quantization Error: Floating-point formats can express a large dynamic range of values using a combination of a mantissa and an exponent. A set of floating point numbers in $X \in \mathbb{R}$ are expressed as follows:

$$x = (-1)^s \times 2^{2^e - b} \times (1 + f_1 \times 2^{-1} + f_2 \times 2^{-2} + ... + f_m \times 2^{-m}) \quad (1)$$

where $s \in \{0, 1\}$ is the sign, e is exponent bit width and $f_i \in \{0, 1\}$ is the m-bit mantissa or fraction.

The dynamic range of a floating point format is determined by the width of its exponent. The exponent value is expressed in powers of 2 and serves as a scaling factor for the mantissa. This means that floating-point numbers are not uniformly spaced, but have a smaller step-size around zero that increases with the magnitude of the represented value. This allows floating-point formats to represent smaller values with better accuracy.

The width of the mantissa determines the number of grid

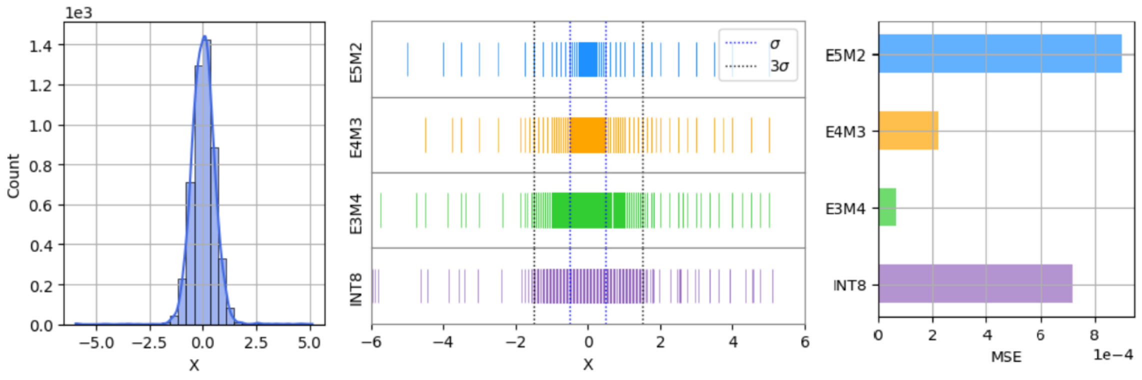


Figure 1. (**left**) Histogram of the tensor $X \sim \mathcal{N}(\mu = 0.0,\, \sigma^2 = 0.5)$, that contains a small number (1%) of outliers uniformly distributed between -6.0 to 6.0. (**center**) Distribution of quantized values represented by E5M2, E4M3, E3M4 and INT8 data formats. (**right**) Overall quantization error as measured by mean-square-error (MSE).

points represented for each incremental step of the exponent, which in turn affects the precision of the format. These properties allow floating-point formats to support higher dynamic range without compromising the accuracy of smaller values, making them well-suited for representing many frequently occurring data patterns in deep learning workloads that exhibit long-tailed normal distributions.

Figure 1 illustrates the differences in distribution of quantized values and impact of outliers on both FP8 and INT8 formats. In the center plot, FP8 formats show a greater concentration of grid points in the middle of the distribution, indicating a region of higher precision closer to zero. The high-precision band is wider for formats with more mantissa bits, allowing them to represent a greater percentage of the 3σ region of the original data with higher accuracy. In contrast, INT8 quantization operates with a *fixed step-size* that is determined by the largest value present in the input data. This means that the outliers can significantly influence the step-size by stretching the quantization grid, resulting in fewer grid points under the 3σ region. This is reflected in the overall quantization error (MSE) shown on the right, where E4M3 and E3M4 formats have significantly outperformed INT8, while E5M2 performed worse because it has fewer mantissa bits.

3 Quantization Workflow

There are several challenges in creating a generalized quantization scheme that can be applied to networks across multiple application domains and involves multiple data formats. The networks may have different requirements for dynamic range, precision and may contain operations that are sensitive to quantization. To facilitate generalization, the quantization scheme must be capable of supporting a broad set of common operations, while also having the ability to adapt to meet the unique requirements of various applications. Our framework accomplishes this by incorporating both a *standard quantization scheme* that can be broadly applied, as well as an *extended quantization scheme* that optimizes specific operations through an iterative tuning process. Figure 2 depicts the high-level workflow for post-training FP8 quantization. The standard quantization scheme is the default configuration applied to common set of operators across different architectures, while the extended scheme is specific to an architecture and is applied incrementally in a feedback loop.

The flow diagram in Figure 2 also includes an additional *BatchNorm Calibration* step applied only to computer vision models. (Sun et al., 2019) have shown that retuning BatchNorm parameters (*mean* and *variance*) to compensate for the variance shift caused by quantization, has significantly improved the inference accuracy. Additionally, please note that E5M2 uses *direct quantization* and does not require *Range Calibration* because it has sufficient dynamic range to handle outliers. For E4M3 and E3M4 formats, we found simple *max* scaling to be sufficient for handling outliers. We also examined more sophisticated range-calibration methods such as KL divergence (Darvish Rouhani et al., 2020; Migacz, 2017), MSE error (Choukroun et al., 2019; Zhao et al., 2019a) and percentile (Gholami et al., 2021) which did not provide any additional benefits.

3.1 Standard Quantization Scheme

This section outlines the components of the standard quantization scheme, which is derived from our extensive studies conducted on several deep learning tasks across multiple application domains. This scheme is applied to the common subset of operators including Convolution, Linear and Embedding. This scheme is also identical to INT8 quantization scheme, allowing a fair accuracy comparison.

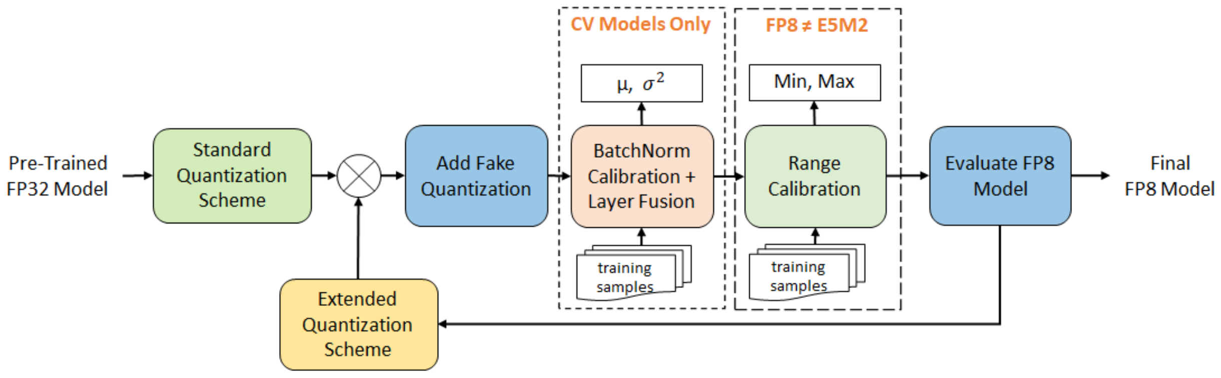


Figure 2. Standard Quantization Scheme: default configuration for broad set of operations across different workloads, *Extended Quantization Scheme*: configuration for additional operator coverage (Ex: LayerNorm, BatchNorm & element-wise), mixed FP8 formats, dynamic quantization, *BatchNorm Calibration*: recalibrate mean and variance parameters to recover accuracy lost due to quantization, *Range calibration*: max scaling, outlier clipping (more discussions in Appendix A.1).

Weight and Activation Scaling: We recommend using per-channel scaling for weights across all networks. Although FP8 formats have sufficient dynamic range to handle common weight distributions, empirical evidence suggests that applying per-channel scaling can reduce rounding errors by effectively utilizing the full encoding space for each channel. Similarly, we found per-tensor scaling to be adequate for handling outliers using FP8 formats. The scale factors are computed as below:

$$s = (float\_max/max\_T) \tag{2}$$

where *float_max* is the max representable value of the selected FP8 format, and *max_T* is the calibrated *absmax* value of the tensor. Some recent studies (Xiao et al., 2022; Wei et al., 2022; Dettmers et al., 2022) have indicated that per-channel activation scaling can benefit INT8 quantization. However, such methods may require special kernel implementations that are likely to incur higher compute overheads, hence they are not included in our study.

First and Last Operator: Previous studies (Han et al., 2015b; Choi et al., 2018; Micikevicius et al., 2022) on convolution networks have shown that the first convolution and the last fully-connected layers are more sensitive to quantization. These two operators typically constitute $< 1\%$ of the total computation. Therefore, we continue to maintain these layers in higher precision to preserve model accuracy. Please note that this exception is only applicable to convolutional neural networks.

3.2 Extended Quantization Scheme

This section outlines the quantization scheme that is selectively applied to address the specific needs of an application. These methods are applied incrementally to maximize model efficiency while preserving accuracy.

Expanded Operator Coverage: Neural networks spend significant fraction of their execution time in memory-bound operations such as LayerNorm, BatchNorm[1] and element-wise operators such as Add and Mul. Previous attempts Bhandare et al. (2019); Kim et al. (2021) to quantize these operators using integer approximation were unsuccessful in maintaining the model accuracy. Our experiments show that FP8 formats are capable of handling these operators without sacrificing model accuracy.

Mixed FP8 Formats: The data distributions of weights and activations can vary depending on the architecture of the model and the dataset it is trained on. Figure 3 shows typical distributions of weight and activation tensors in NLP and computer vision workloads. The weight distributions in both classes of models tend to follow normal distributions with lots values near zero. These tensors require more mantissa bits in the data format to represent the distribution accurately. In contrast, activations of NLP models show a lot of outliers which demand a larger dynamic range in the data format to ensure the outliers are accurately represented. We balance this trade-off by assigning E5M2 or E4M3 format for *range-bound* tensors and E3M4 for *precision-bound* tensors.

Static vs. Dynamic Quantization: We use static quantization as the default method throughout our study because it is computationally more efficient. However, we studied the accuracy impact of dynamic quantization on all FP8 formats and found that it offers no additional benefits to E5M2 but observed a noticeable improvement in accuracy for E4M3 and E3M4 formats on selected models.

[1]Ones that cannot be folded into Convolution layers, Ex: Densenet

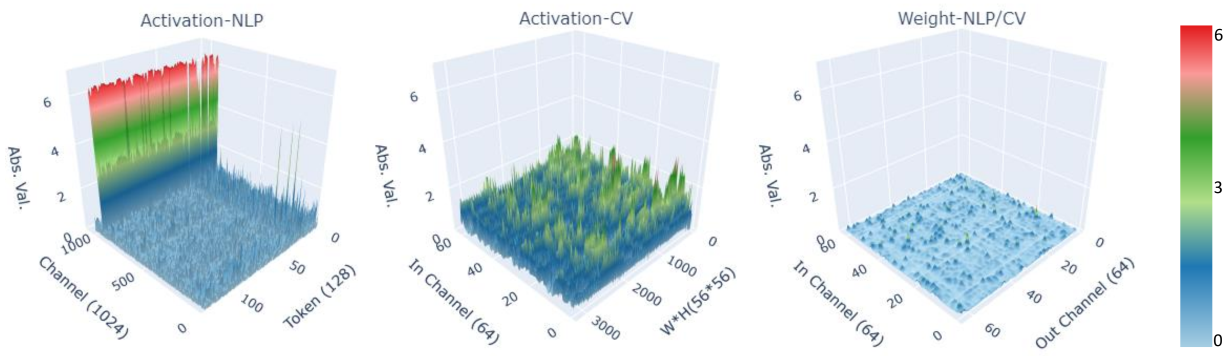


Figure 3. Tensor Distributions: (**left**) activations in NLP workloads contain outliers, hence they are *range-bounded*, (**center**) Activation in CV workloads tend to be *precision-bounded*, (**right**) Weight tensors from both CV & NLP networks tend to be *precision-bounded*.

4 Results

4.1 Experimental Setup

We demonstrate the FP8 quantization results using a software emulation framework which contains two major components, *data type emulation* and *model quantization*. For data type emulation, we utilized the FP8 Emulation Toolkit, which provides a reference implementation that runs FP32 hardware. We leverage Neural Compressor to perform model quantization by incorporating both standard and extended quantization schemes, along with FP8 specific quantization methods such as BatchNorm calibration and support for mixed FP8 formats. Our framework supports a wide range of quantized operators, including compute operators such as Convolution, Linear, MatMul, BatchMatMul and memory operators such as Embedding, BatchNorm, LayerNorm, Add and Mul.

We evaluated our quantization methods on more than 200 different tasks, using 75 unique model architectures and over 20 different datasets. The models were selected randomly from a pool of a combination of diversity and popularity from mainstream hubs such as Hugging Face Models and Torch Vision, as well as individual models from Github based on their popularity. The following is a partial list of workloads that are broadly categorized under Natural Language Processing (NLP) and Computer Vision (CV).

Text and Natural Language Processing: We have evaluated 38 different networks in this category on a wide range of NLP tasks, which can be further subdivided as follows:

- *Generative language modeling*. We evaluated *Bloom* (Scao et al., 2022) and *LLaMA* (Touvron et al., 2023), two representative open-source LLMs, and evaluate the accuracy using *lambada-openai*.
- *Text classification*. We evaluated over 30 different networks (e.g, *Bert-Large* (Devlin et al., 2018), *DistilBert* (Sanh et al., 2019), *Longformer* (Beltagy et al., 2020)) on a wide variety of tasks (e.g., *mrpc*, *cola*, *sts-b*, *sst2*).
- *Summarization*. We measured the accuracy of *pegasus* (Zhang et al., 2020) on *samsum* dataset.
- *Other NLP tasks*. Few other selected models such as MarianMT (Junczys-Dowmunt et al., 2018) for neural machine translation and DialogGPT (Zhang et al., 2019) for language modeling on WMT_EN_RO and wikitext datasets.

Image and Computer Vision: We evaluated 34 different networks on various computer vision tasks from the following categories.

- *Image generation*. We evaluated Stable Diffusion, an open-source state-of-the-art latent text-to-image diffusion model and evaluate using FID (Heusel et al., 2017).
- *Image classification*. We evaluate a wide range of convolutional neural networks (CNNs) such as VGG (Simonyan & Zisserman, 2014), GoogleNet (Szegedy et al., 2015), ResNet (He et al., 2016), ShuffleNet (Zhang et al., 2018b), EfficientNet (Tan & Le, 2019), and Transformer-based vision models such as ViT (Dosovitskiy et al., 2020) on ImageNet ILSVRC 2012 and CIFAR-10.
- *Image segmentation & object detection*. We select typical models such as U-Net (Ronneberger et al., 2015) for image segmentation using the dataset from Kaggle Carvana Image Masking Challenge (Shaler et al., 2017) and YoloV3 (Redmon & Farhadi, 2018) for object detection using COCO2014 (Lin et al., 2014).

Audio and Speech Processing. We evaluated two models HuBERT (Hsu et al., 2021) and wav2vec 2.0 (Baevski et al.,

Table 2. Workload Pass Rate. The **bold** shows the overall highest pass rate where E4M3 is 92.64% and INT8 is 65.87%. In particular, E4M3 shows the promising workload coverage 96.32% on NLP.

Data Type	Quantization Approach	Pass Rate (CV)	Pass Rate (NLP)	Pass Rate (All)
E5M2	Direct	55.26%	78.42%	74.89%
E4M3	Static	73.68%	**96.32%**	**92.64%**
E4M3	Dynamic	71.05%	92.11%	88.74%
E3M4	Static	**78.95%**	92.11%	90.04%
E3M4	Dynamic	**78.95%**	92.11%	90.04%
INT8	Static CV | Dynamic NLP	57.89%	67.65%	65.87%

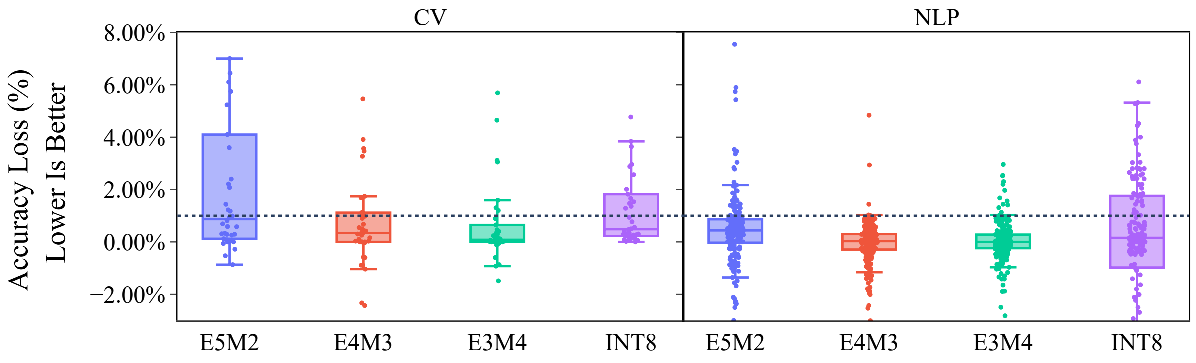


Figure 4. Variability in accuracy loss: INT8 shows higher variability for CV models than E4M3 and E3M4 due to its ineffectiveness on models such as EfficientNet, MobileNetV3, and ViT. Quantization-aware training may partially mitigate this issue, but it is out of scope of this paper. E4M3 and E3M4 show better accuracy & less variability with very few outliers compared to INT8.

Table 3. Model Accuracy. The **bold** shows the best accuracy is less than 1% loss against FP32 baseline.

Model	Dataset/Task	FP32	E5M2	E4M3	E3M4	INT8
ResNet-50	ImageNet 2012	0.7615	0.7544	0.7592	**0.7604**	0.7595
DenseNet-121	ImageNet 2012	0.7444	0.7435	0.7451	**0.7459**	0.7253
Wav2Vec2	LibriSpeech	0.9660	0.9632	**0.9661**	0.9658	0.9552
DLRM	Criteo Terabyte	0.8027	0.8016	**0.8025**	**0.8025**	0.8024
Bert-Base	STS-B	0.8975	0.8934	**0.8979**	0.8966	0.8809
Bert-Large	COLA	0.6257	0.6238	0.6257	0.6282	**0.6389**
DistilBert	MRPC	0.8916	0.8897	0.8943	0.895	**0.9042**
Bloom-7B1	Lambada-openai	0.5764	0.5424	0.5748	0.5824	**0.5977**
Bloom-176B	Lambada-openai	0.6777	0.6753	0.6757	**0.6938**	0.6899
LLaMA-65B	Lambada-openai	0.7908	0.7840	**0.7914**	0.7778	0.7155

2020) for speech recognition and evaluate the accuracy using LibriSpeech (Panayotov et al., 2015).

Recommendation System. We evaluated Deep Learning Recommendation Model (DLRM) (Naumov et al., 2019) and measured the accuracy on Criteo Terabyte.

4.2 Quantization Results

4.2.1 *Accuracy*

Note that the *pass rate* in Table 2 is the percentage of workloads that meet the accuracy criterion of 1% relative loss against FP32 baseline. SmoothQuant Xiao et al. (2022) is enabled on NLP models with the default smoothing alpha

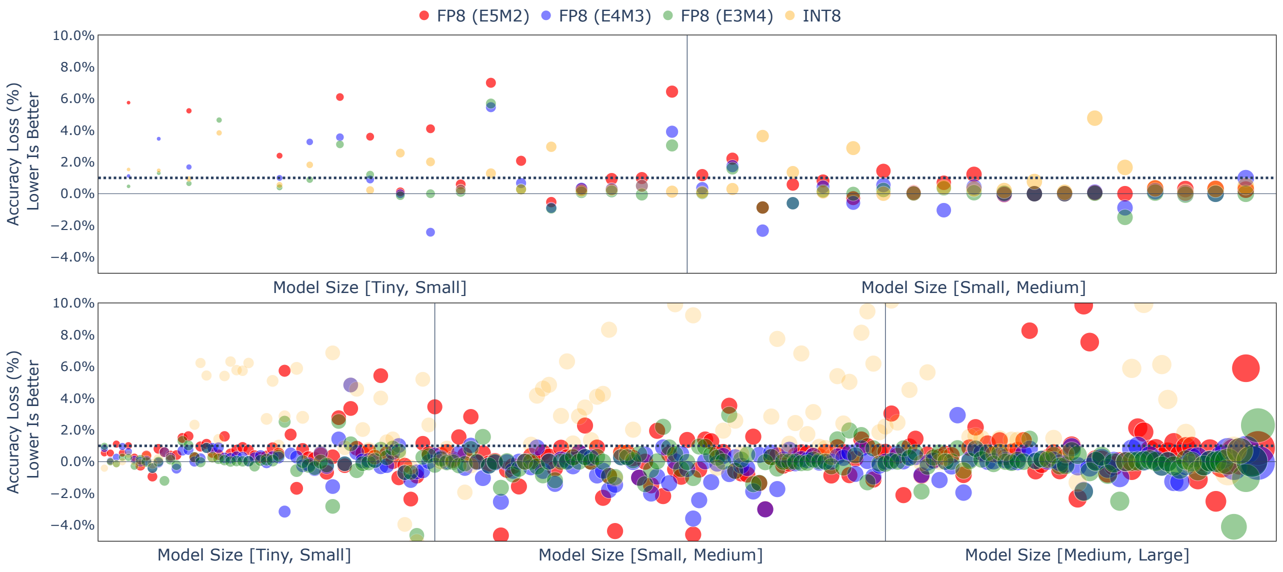


Figure 5. Accuracy Loss by Size on CV (top) and NLP (bottom). The model size is represented by the ball size in the scale of $log10(model\_size)$, where tiny/small/medium/large is defined by the size range in MB $<= 32$, $(32, 384]$, $(384, 512]$, and > 512 respectively. Note that some points are overlayed due to the similar accuracy (e.g., E4M3 in blue and E3M4 in green on NLP models).

value (alpha tuning is out of scope in this paper). Figure 4 illustrates the variability of accuracy loss for different data formats across CV and NLP workloads.

Table 3 shows the accuracy of a few representative samples from all CV and NLP workloads. Figure 5 shows the accuracy loss of all workloads sorted by the model size in ascending order.

4.2.2 *Generation Quality*

Figure 6 shows the image generated by Stable Diffusion with the prompt ”A photo of an astronaut riding a horse on Mars”. Our subjective analysis reveals that FP8 formats achieve superior image quality compared to INT8, as indicated by the green arrow. Additionally, E4M3 and E3M4 produce smoother images and generate more intricate details, particularly on the astronaut. We employ FID score to compare the quality of generated images (lower is better) and see that FID score aligns with our subjective evaluation. More samples on Stable Diffusion are shown in Appendix A.2.

Table 4 shows the sample text generated by Bloom on the prompt with 32 input tokens using beam search size 4. Given the prompt as the input, you can see E3M4 shows better response than INT8 with more comprehensive content and few repeated tokens (e.g., *saw many strange*). Appendix A.3 shows the full output on different data format and quantization approach.

4.3 Discussion

4.3.1 *Standard Quantization Scheme*

Quantizing First and Last Operators : For convolutional networks, quantizing the first and last operators reduced the *Pass Rate* for E5M2 and E4M3 formats by 25% and 15% respectively. However, E3M4 can maintain a *Pass Rate* of 70% even with the first and last operators quantized. Therefore, we recommend the enabling of first and last operators for FP8 quantization as a tuning option.

BatchNorm Calibration: We use data augmentation to enhance the feature diversity of the calibration data which impacts the quality of BatchNorm statistics and model accuracy. Figure 7 compares the effectiveness of training and inference data augmentation methods in preserving model accuracy at different calibration data sample sizes. We found training transform to be more effective even at smaller sample sizes (<3K). However, we recommend sample size of 3K with training transform for achieving best results across a wide range of networks.

4.3.2 *Extended Quantization Scheme*

Mixed FP8 Formats: Figure 8 illustrates how using mixed FP8 formats on the input can impact the quantization error of the output of a Linear operator from BERT-base (MPRC) model. Our experiments show that using E4M3 for activations and E3M4 for weights produced best accuracy results on a range of NLP workloads. The accuracy improvements achieved by this scheme for Bert, Funnel, and Longformer

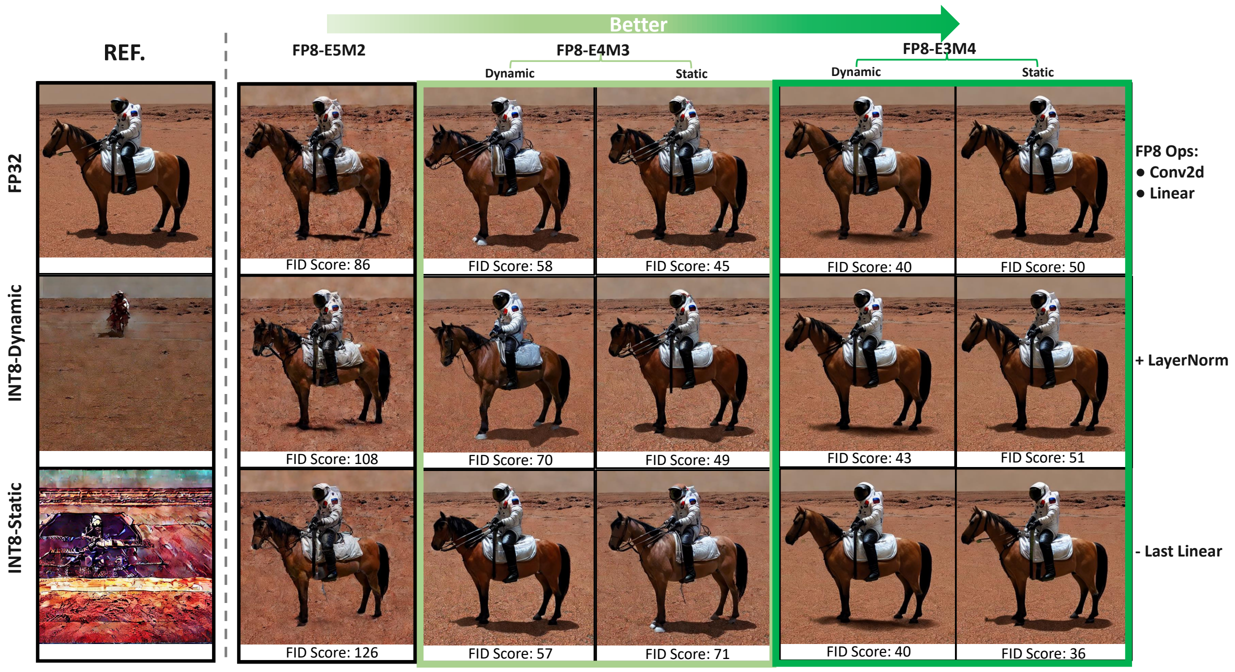


Figure 6. Stable Diffusion with Prompt "A photo of an astronaut riding a horse on Mars"

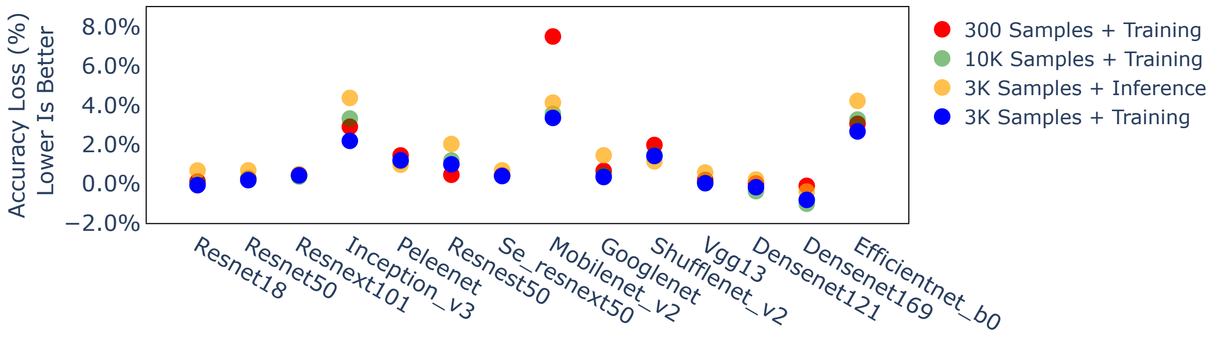


Figure 7. CV Models with BatchNorm Operation

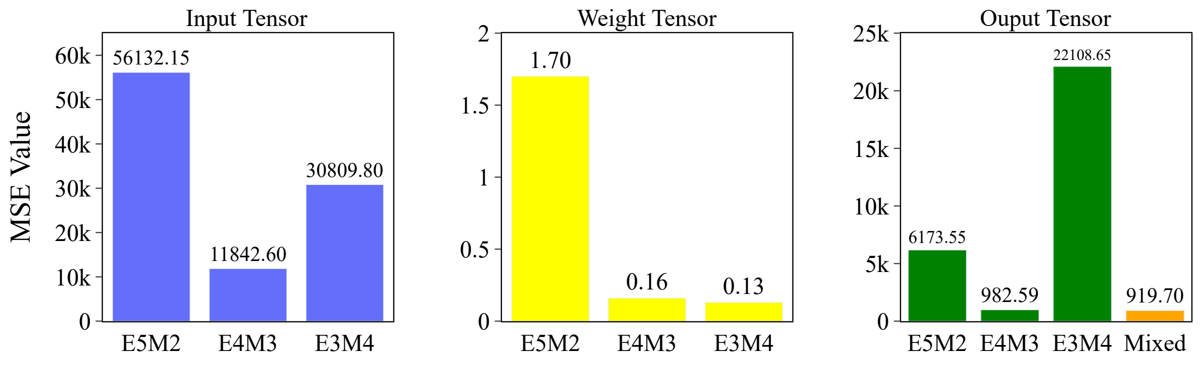


Figure 8. MSE of FP8 Quantization with Mixed Formats vs. Single Format on Bert-Base (MRPC)

Table 4. Generated Text of Bloom on 32 Input Tokens

Prompt: Once upon a time, there existed a little girl, who liked to have adventures. She wanted to go to places and meet new people, and have fun.
Output (FP32): One day, she decided to go on a trip. She packed her suitcase and went to the airport. When she got there, she found out that there was no flight to her destination, so she decided to take a bus. When she got there, she found out that there was no bus to her destination...
Output (INT8): This little girl was very adventurous. One day she decided to go on a trip to a faraway country. When she got there the little girl *saw many strange* things. She *saw many strange* people. She *saw many strange* animals. She saw many strange sights. She saw many ...
Output (E3M4): One day, she decided to go on an adventure. She packed her suitcase and went to the airport. She boarded a plane and flew to New York City. There, she met a man, and they had a great time together. They went to a restaurant and ate delicious food. Then, they went to...

Table 5. Model Accuracy of FP8 Format (Single vs. Mixed). Mixed FP8 formats (in bold) show higher accuracy than all the other single FP8 formats on the below NLP workloads.

Model	Task	FP32	E5M2	E4M3	E3M4	Mixed
Bert-Base	MRPC	0.9069	0.9040	0.9050	0.9050	**0.9069**
Bert-Large	RTE	0.7256	0.6968	0.7329	0.6931	**0.7365**
Funnel	MRPC	0.9225	0.9215	0.9207	0.3704	**0.9233**
Longformer	MRPC	0.9146	0.8374	0.9113	0.9084	**0.9143**

Table 6. Model Accuracy of Quantization Approach (Static vs. Dynamic)

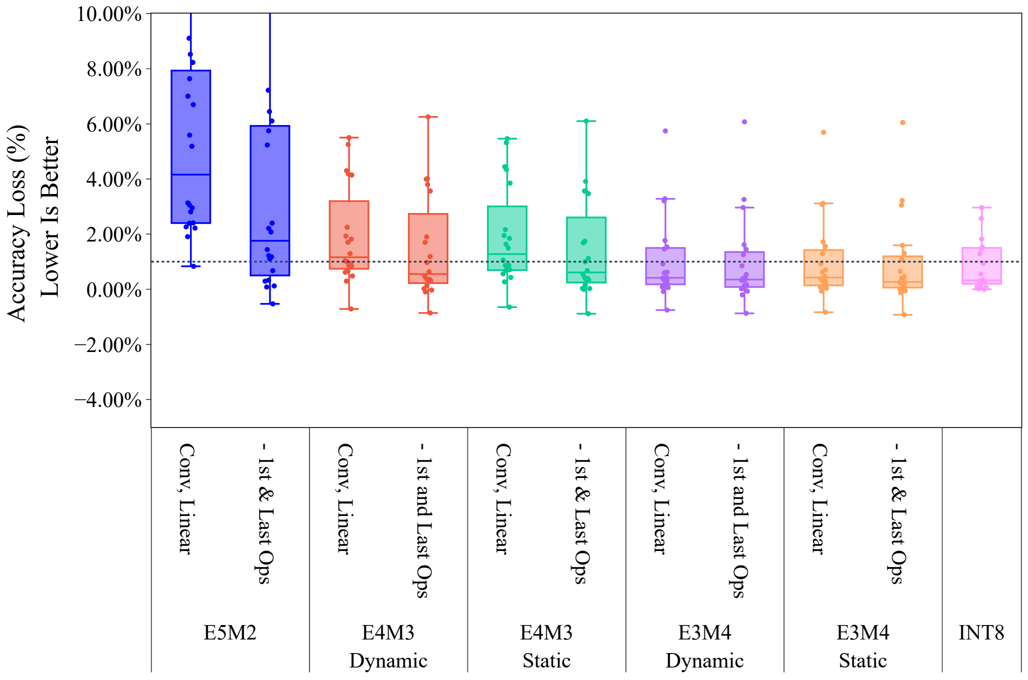
Model	Task	FP8 Format	Dynamic	Static	Improvement
Bert-Base	MRPC	E4M3	0.9151	0.9072	**+0.87%**
Bert-Base	COLA	E4M3	0.6058	0.6033	**+0.41%**
Bert-Large	RTE	E4M3	0.7401	0.7329	**+0.98%**
Xlm-Roberta-Base	MRPC	E3M4	0.8962	0.8919	**+0.48%**

models are presented in Table 5.

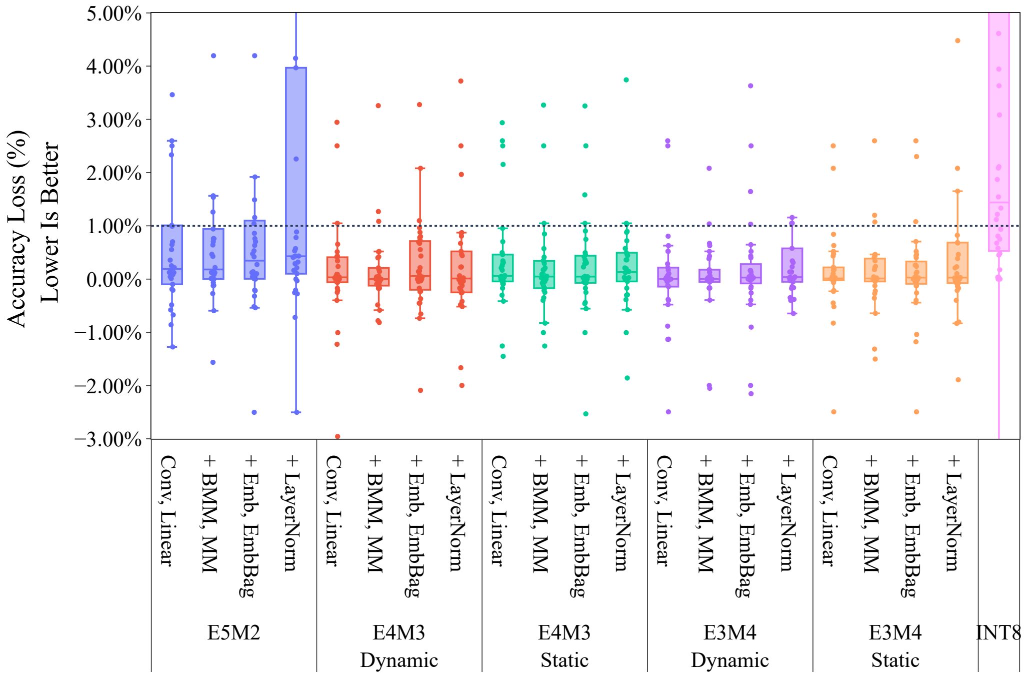
Expanded Operator Coverage: Figure 9 has the results from our quantization studies extended to a wider range of operators such as BatchMatMul, MatMul, Embedding and LayerNorm. Our results show that E4M3 achieves overall better accuracy and smaller variability in accuracy loss across a broad range of NLP tasks. **Static vs. Dynamic Quantization:** While static quantization is the default approach in our recipes, we also studied the impact of dynamic quantization on model accuracy. The results indicate that dynamic quantization can improve the accuracy of NLP models when quantizing with E4M3 and E3M4 formats as shown in Table 6.

5 Summary and Future Work

We present a set of post-training quantization recipes for FP8 inference and demonstrate the effectiveness across 75 unique network architectures covering a wide range of tasks such as language modeling, text generation, image classification and generation. We recommend E3M4 and E4M3 as the default FP8 format for CV and NLP models respectively, while additional recipes such as mixed FP8 formats and expanded FP8 operator coverage are worthwhile exploring to produce an optimal FP8 model. As our future work, we plan to apply FP8 quantization recipes to more diverse LLM models (e.g., BioGPT (Luo et al., 2022), Llama2 Chat (Touvron et al., 2023), Code Llama (Rozière et al., 2023)), and contribute our recipes and implementation to the open source community.



(a) CV Models



(b) NLP Models

Figure 9. Model Accuracy Impact by Extended Quantization Recipes

References

Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Bhalgat, Y., Lee, J., Nagel, M., Blankevoort, T., and Kwak, N. LSQ+: improving low-bit quantization through learnable offsets and better initialization. *CoRR*, abs/2004.09576, 2020. URL https://arxiv.org/abs/2004.09576.

Bhandare, A., Sripathi, V., Karkada, D., Menon, V., Choi, S., Datta, K., and Saletore, V. Efficient 8-bit quantization of transformer neural machine language translation model. *arXiv preprint arXiv:1906.00532*, 2019.

Cai, Z., He, X., Sun, J., and Vasconcelos, N. Deep learning with low precision by half-wave gaussian quantization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Choi, J., Wang, Z., Venkataramani, S., Chuang, P. I., Srinivasan, V., and Gopalakrishnan, K. PACT: parameterized clipping activation for quantized neural networks. *CoRR*, abs/1805.06085, 2018. URL http://arxiv.org/abs/1805.06085.

Choukroun, Y., Kravchik, E., Yang, F., and Kisilev, P. Low-bit quantization of neural networks for efficient inference. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 3009–3018. IEEE, 2019.

Darvish Rouhani, B., Lo, D., Zhao, R., Liu, M., Fowers, J., Ovtcharov, K., Vinogradsky, A., Massengill, S., Yang, L., Bittner, R., et al. Pushing the limits of narrow precision inferencing at cloud scale with microsoft floating point. *Advances in neural information processing systems*, 33: 10271–10281, 2020.

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Esser, S. K., McKinstry, J. L., Bablani, D., Appuswamy, R., and Modha, D. S. Learned step size quantization. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkgO66VKDS.

Fang, J., Shafiee, A., Abdel-Aziz, H., Thorsley, D., Georgiadis, G., and Hassoun, J. Near-lossless post-training quantization of deep neural networks via a piecewise linear approximation. *CoRR*, abs/2002.00104, 2020. URL https://arxiv.org/abs/2002.00104.

Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M. W., and Keutzer, K. A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630*, 2021.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding, 2015a. URL https://arxiv.org/abs/1510.00149.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015b.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Klambauer, G., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017. URL http://arxiv.org/abs/1706.08500.

Hsu, W.-N., Bolte, B., Tsai, Y.-H. H., Lakhotia, K., Salakhutdinov, R., and Mohamed, A. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Jiang, C. Efficient quantization techniques for deep neural networks. In *2021 International Conference on Signal Processing and Machine Learning (CONF-SPML)*, pp. 271–277. IEEE, 2021.

Junczys-Dowmunt, M., Grundkiewicz, R., Dwojak, T., Hoang, H., Heafield, K., Neckermann, T., Seide, F., Germann, U., Fikri Aji, A., Bogoychev, N., Martins, A. F. T., and Birch, A. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pp. 116–121, Melbourne, Australia, July 2018. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P18-4020.

Kim, S., Gholami, A., Yao, Z., Mahoney, M. W., and Keutzer, K. I-bert: Integer-only bert quantization. In *International conference on machine learning*, pp. 5506–5518. PMLR, 2021.

Krishnamoorthi, R. Quantizing deep convolutional networks for efficient inference: A whitepaper. *CoRR*, abs/1806.08342, 2018. URL http://arxiv.org/abs/1806.08342.

Kuzmin, A., Van Baalen, M., Ren, Y., Nagel, M., Peters, J., and Blankevoort, T. FP8 Quantization: The Power of the Exponent, August 2022. URL http://arxiv.org/abs/2208.09225. arXiv:2208.09225 [cs].

Li, Y., Dong, X., and Wang, W. Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BkgXT24tDS.

Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. URL http://arxiv.org/abs/1405.0312.

Luo, R., Sun, L., Xia, Y., Qin, T., Zhang, S., Poon, H., and Liu, T.-Y. Biogpt: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics*, 23(6), 2022.

Mellempudi, N., Kundu, A., Mudigere, D., Das, D., Kaul, B., and Dubey, P. Ternary neural networks with fine-grained quantization. *CoRR*, abs/1705.01462, 2017. URL http://arxiv.org/abs/1705.01462.

Mellempudi, N., Srinivasan, S., Das, D., and Kaul, B. Mixed Precision Training With 8-bit Floating Point, May 2019. URL http://arxiv.org/abs/1905.12334. arXiv:1905.12334 [cs, stat].

Micikevicius, P., Stosic, D., Burgess, N., Cornea, M., Dubey, P., Grisenthwaite, R., Ha, S., Heinecke, A., Judd, P., Kamalu, J., et al. Fp8 formats for deep learning. *arXiv preprint arXiv:2209.05433*, 2022.

Migacz, S. 8-bit inference with tensorrt, 2017. URL https://on-demand.gputechconf.com/gtc/2017/presentation/s7310-8-bit-inference-with-tensorrt.pdf.

Miyashita, D., Lee, E. H., and Murmann, B. Convolutional neural networks using logarithmic data representation. *CoRR*, abs/1603.01025, 2016. URL http://arxiv.org/abs/1603.01025.

Naumov, M., Mudigere, D., Shi, H.-J. M., Huang, J., Sundaraman, N., Park, J., Wang, X., Gupta, U., Wu, C.-J., Azzolini, A. G., et al. Deep learning recommendation model for personalization and recommendation systems. *arXiv preprint arXiv:1906.00091*, 2019.

Noune, B., Jones, P., Justus, D., Masters, D., and Luschi, C. 8-bit Numerical Formats for Deep Neural Networks, June 2022. URL http://arxiv.org/abs/2206.02915. arXiv:2206.02915 [cs].

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. Librispeech: an asr corpus based on public domain audio books. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 5206–5210. IEEE, 2015.

Redmon, J. and Farhadi, A. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pp. 234–241. Springer, 2015.

Rozière, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Remez, T., Rapin, J., Kozhevnikov, A., Evtimov, I., Bitton, J., Bhatt, M., Ferrer, C. C., Grattafiori, A., Xiong, W., Défossez, A., Copet, J., Azhar, F., Touvron, H., Martin, L., Usunier, N., Scialom, T., and Synnaeve, G. Code llama: Open foundation models for code, 2023.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Scao, T. L., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A. S., Yvon, F., Gallé, M., et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2022.

Shaler, B., DanGill, Maggie, McDonald, M., Patricia, and Cukierski, W. Carvana image masking challenge, 2017. URL https://kaggle.com/competitions/carvana-image-masking-challenge.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Sun, X., Choi, J., Chen, C.-Y., Wang, N., Venkataramani, S., Srinivasan, V. V., Cui, X., Zhang, W., and Gopalakrishnan, K. Hybrid 8-bit Floating Point (HFP8) Training and Inference for Deep Neural Networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/65fc9fb4897a89789352e211ca2d398f-Abstract.html.

Sung, W., Shin, S., and Hwang, K. Resiliency of deep neural networks under quantization. *CoRR*, abs/1511.06488, 2015. URL http://arxiv.org/abs/1511.06488.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Tan, M. and Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pp. 6105–6114. PMLR, 2019.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Vanhoucke, V., Senior, A., and Mao, M. Z. Improving the speed of neural networks on cpus. In *Deep Learning and Unsupervised Feature Learning Workshop, NIPS 2011*, 2011.

Wang, N., Choi, J., Brand, D., Chen, C.-Y., and Gopalakrishnan, K. Training Deep Neural Networks with 8-bit Floating Point Numbers. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/335d3d1cd7ef05ec77714a215134914c-Paper.pdf.

Wei, X., Zhang, Y., Zhang, X., Gong, R., Zhang, S., Zhang, Q., Yu, F., and Liu, X. Outlier suppression: Pushing the limit of low-bit transformer language models. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=yW5zeRSFdZ.

Xiao, G., Lin, J., Seznec, M., Demouth, J., and Han, S. Smoothquant: Accurate and efficient post-training quantization for large language models. *arXiv preprint arXiv:2211.10438*, 2022.

Zhang, D., Yang, J., Ye, D., and Hua, G. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. *CoRR*, abs/1807.10029, 2018a. URL http://arxiv.org/abs/1807.10029.

Zhang, J., Zhao, Y., Saleh, M., and Liu, P. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pp. 11328–11339. PMLR, 2020.

Zhang, X., Zhou, X., Lin, M., and Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6848–6856, 2018b.

Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., Gao, J., Liu, J., and Dolan, B. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*, 2019.

Zhao, R., Hu, Y., Dotzel, J., De Sa, C., and Zhang, Z. Improving neural network quantization without retraining using outlier channel splitting. In *International conference on machine learning*, pp. 7543–7552. PMLR, 2019a.

Zhao, R., Hu, Y., Dotzel, J., Sa, C. D., and Zhang, Z. Improving neural network quantization without retraining using outlier channel splitting. *CoRR*, abs/1901.09504, 2019b. URL http://arxiv.org/abs/1901.09504.

Zhou, A., Yao, A., Guo, Y., Xu, L., and Chen, Y. Incremental network quantization: Towards lossless CNNs with low-precision weights. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=HyQJ-mclg.

Zhou, S., Ni, Z., Zhou, X., Wen, H., Wu, Y., and Zou, Y. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *CoRR*, abs/1606.06160, 2016. URL http://arxiv.org/abs/1606.06160.

A Appendix

A.1 Range Calibration Algorithms

Scale algorithm is not applied for E5M2 due to its large dynamic range. However, the scale is very crucial for E4M3 to help make sure all data is recorded in E4M3 data range. Mainly, there are three classic scale algorithms used in INT8 quantization. Percentile and KL can help INT8 clip the min-max of observed data range to skip outliers and improve the accuracy of data representation(Jiang, 2021). However, they may have different behavior on FP8 due to the special data distribution of FP8.

In Figure 10, we show a demo to explain the shortage of KL when using FP8. The demo uses a tensor with some outliers around 6 and after KL process, the clipped max value is 2. The lines at the bottom show FP8 mapped data with different max values. The upper line have a large data range from 0-6 while the other line have more representations for small values. We expect the lower line have a better representation than the upper one, but it actually have a large MSE than the upper one. We can observer that the density of the last block in the lower line is much sparse than the upper one, while the enhanced small value representations do not help a lot in MSE.

As mentioned early, the FP8 has advantages of representing larger range of values and obtaining better accuracy at lower range because of denser representation on the contrary to the uniform representation at the whole range of INT8. FP8 format is represented by exponent bits (e) and mantissa bits (m). Here, we use E(e)M(m) as FP8 representation to demonstrate our point. To calculate the density of number for E(e)M(m), we choose a simplified method that uses the differentials between two points with exact same mantissa of value 1 but with a difference of 1 in exponent as $[1 \times 2^n, 1 \times 2^{n+1})$. We know that for any range with such endpoints, the number of values being represented is always 2^m. Therefore, we can calculate the density on this range is as:

$$D_{E(e)M(m)} = 2^m/(2^{n+1} - 2^n) = 2^{m-n} \tag{3}$$

As is well known, any decimal number N can be represented by binary number with exponent $Floor[log_2 N]$. Hence, the density of E(e)M(m) representation in decimal system is:

$$D_{E(e)M(m)} = 2^{m-Floor[log_2 N]} \tag{4}$$

It's clearly shown that the smaller the number N the denser the number of values being represented. On the contrary, the larger the number N the sparser the number of values being represented. Therefore, we always prefer to examine the histogram of our tensor's value and make sure always to represent the high frequency part of our tensor on the lower range on FP8 with higher density, which is in sharp contrast to INT8 with uniform density. Also shown in the density expression, the more the mantissa the denser the number of values being represented as expected.

Operator level means we have to fallback some operators to high precision to let the quantized model meed the accuracy goal. Theoretically, the more operators converted to low precision, the worse the precision will be. Usually, there are special operator types that have a big impact on accuracy, such as LayerNorm. Also, there are some individual operators that have the most impact on accuracy, such as the first and last operators.

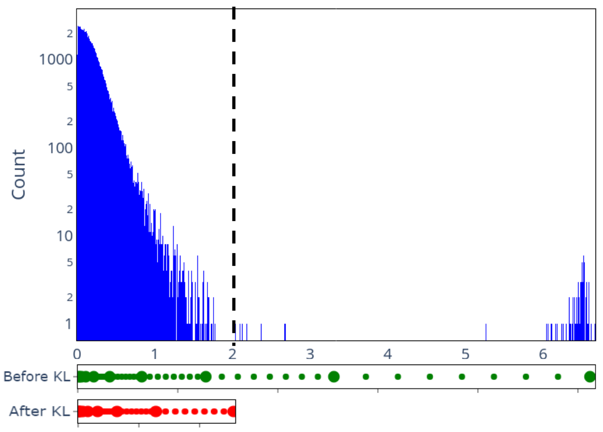


Figure 10. A KL Demo for FP8 mapping

The tuning strategy we proposed allows an automatic tuning for the best accuracy, performance or Pareto optimal. The search space is based on the combination of all tune-able parameters by default. Typically, a customized search space based on our experiment result can help narrow down the search space.

A.2 More Image Generation Samples from Stable Diffusion

Besides the sample generated with the prompt "A photo of an astronaut riding a horse on Mars", we also generate two another images with different prompts as shown in Figure 11 and 12.

A.3 Text Generation Samples from BLOOM

Table 7 shows generated text of BLOOM on different data formats and quantization approaches.

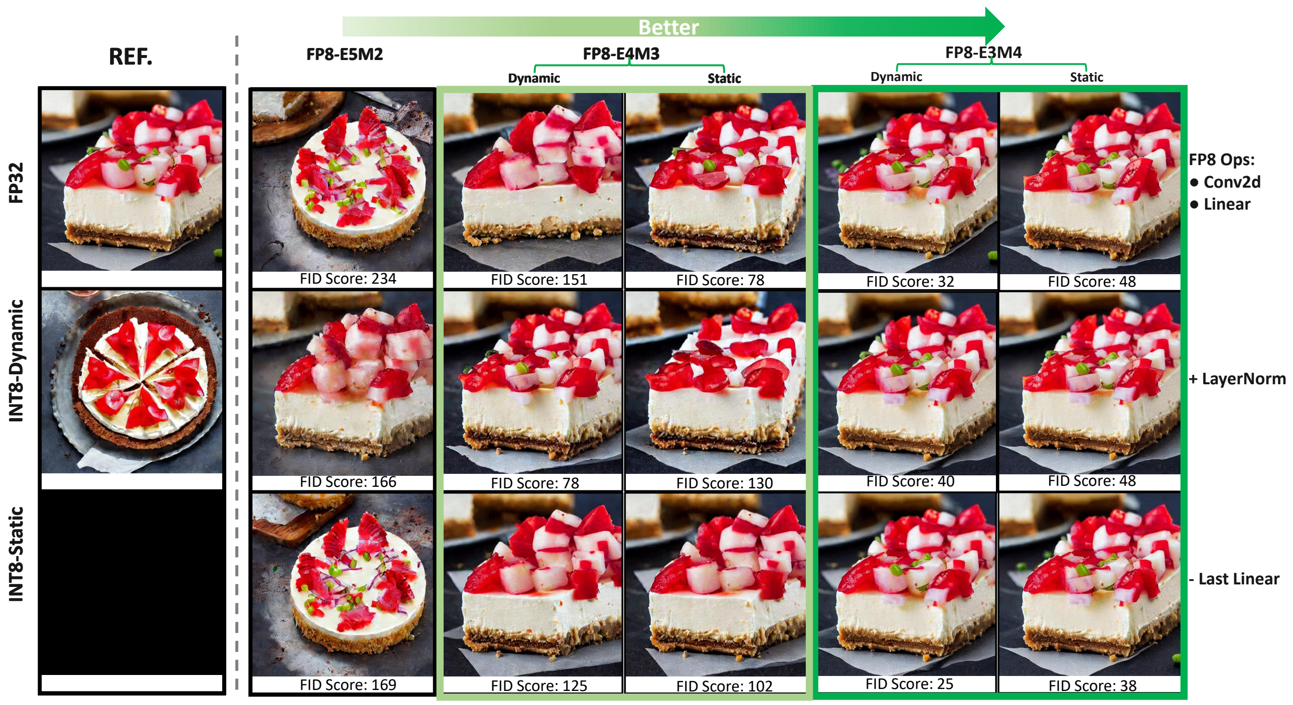


Figure 11. Stable Diffusion with Prompt: "A delicious ceviche cheesecake slice"

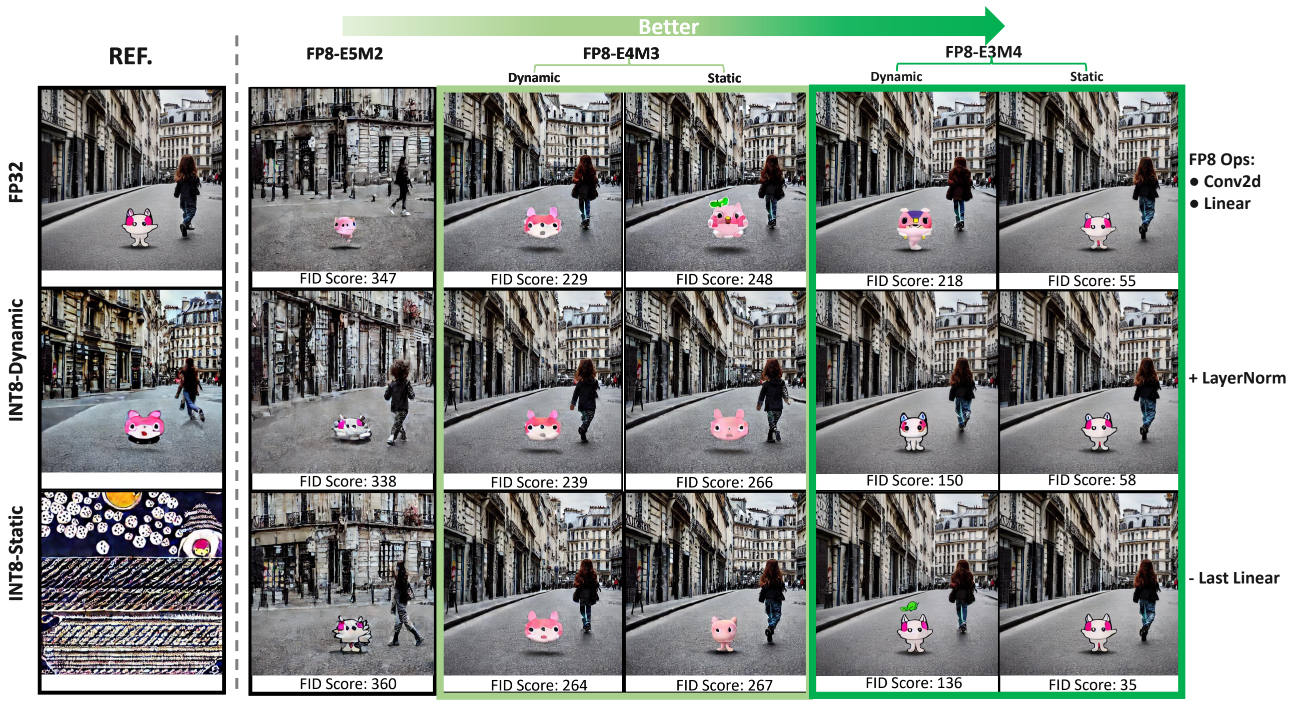


Figure 12. Stable Diffusion with Prompt: "The spirit of a tamagotchi wandering in the city of Paris"

Table 7. Generated Text of BLOOM (32 Input Tokens and 100 Output Tokens)

Prompt: Once upon a time, there existed a little girl, who liked to have adventures. She wanted to go to places and meet new people, and have fun.
Output (FP32): One day, she decided to go on a trip. She packed her suitcase and went to the airport. When she got there, she found out that there was no flight to her destination, so she decided to take a bus. When she got there, she found out that there was no bus to her destination, so she decided to take a train. When she got there, she found out that there was no train to her destination, so she decided to take a car. When she got
Output (INT8): This little girl was very adventurous. One day she decided to go on a trip to a faraway country. When she got there the little girl saw many strange things. She saw many strange people. She saw many strange animals. She saw many strange sights. She saw many strange smells. She saw many strange sounds. She saw many strange sights. She saw many strange smells. She saw many strange sounds. She saw many strange sights. She saw many strange smells. She saw many
Output (E5M2): One day, she decided to go on a trip with her parents. They took her to the seaside, where they had a wonderful time. After that, they went on a trip to the mountains, where they had a wonderful time. Then they went on a trip to the city, where they had a wonderful time. After that, they went on a trip to the countryside, where they had a wonderful time. Then they went on a trip to the zoo, where they had a wonderful time
Output (E4M3 Dynamic): One day, she decided to go on a journey. She packed her suitcase, and set off on her journey. She was very excited about her trip, and she had a lot of things to do. First, she went to the supermarket, and bought a lot of food. Then, she went to the bakery, and bought a lot of bread. Then, she went to the butcher, and bought a lot of meat. Then, she went to the florist, and bought
Output (E4M3 Static): One day, she decided to go on a trip. She packed her suitcase and went to the airport. When she got there, she found out that her flight was delayed. She waited for a long time, and when she finally got on the plane, she found out that it was going to take a long time to get to her destination. When she finally got to her destination, she found out that her flight had been canceled. She waited for a long time, and when she
Output (E3M4 Dynamic): One day, she decided to go on a trip with her parents. They took her to the zoo, where she saw a lion, a tiger, a bear, and a monkey. The lion was the biggest, the tiger was the fiercest, the bear was the strongest, and the monkey was the smartest. The lion ate the tiger, the tiger ate the bear, the bear ate the monkey, and the monkey ate the lion. The lion, the tiger, the bear, and
Output (E3M4 Static): One day, she decided to go on an adventure. She packed her suitcase and went to the airport. She boarded a plane and flew to New York City. There, she met a man, and they had a great time together. They went to a restaurant and ate delicious food. Then, they went to the park and had a great time together. They went to the zoo and had a great time together. They went to the beach and had a great time together. They went to
Output (FP8 Mixed): One day, she decided to go on a trip. She packed her suitcase and went to the airport. When she got there, she took a taxi to her hotel. When she got to her room, she unpacked her suitcase and went to the bathroom. She washed her face, brushed her teeth, and put on her makeup. When she was done, she put on her clothes and went to the restaurant. When she got there, she sat down at a table and ordered a