SSVQ: Unleashing the Potential of Vector Quantization with Sign-Splitting

Supplementary Material

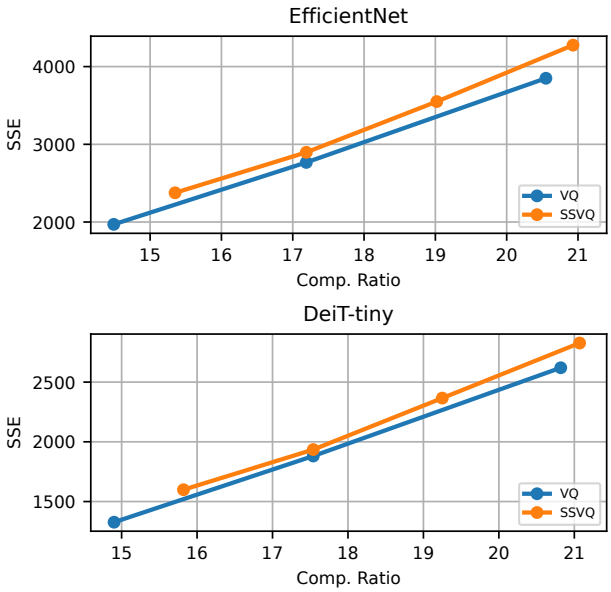

Figure 9. Comparison of initial clustering error between VQ and SSVQ.

9. Analysis of initial clustering error

Although storing the sign bit requires an additional 1 bit, our method does not introduce extra storage overhead. After extracting the sign bits, we can cluster the absolute values of the weights. Naturally, the similarity among non-negative weights is higher, so we only need fewer bits for cluster indexing. In Fig. 9, we compare the initial clustering errors of SSVQ and VQ. The results show that under the same compression rate, their clustering errors are comparable, which validates our hypothesis.

10. Further analysis with dimension

We further investigate the impact of quantization dimensionality. While prior work [19, 28, 39, 44] has demonstrated that increasing quantization dimensionality improves model accuracy under fixed index bits, this approach introduces significant scalability challenges. Specifically, when doubling the dimensionality from d to $2 \times d$ with constant index bits, the codebook size expands exponentially by 2^d times. This substantial expansion makes the codebook a critical factor in compression ratio computation, potentially offsetting the benefits of higher-dimensional quantization.

The codebook storage impact is particularly pronounced in compact architectures like MobileNet and DeiT-Tiny. As shown in Fig. 5, VQ achieves optimal performance at d=4, while SSVQ performs best at d=8. For larger models (DeiT-Small and ConvNext-Tiny), VQ demonstrates better accuracy-compression trade-offs at d=8. To further explore SSVQ's scalability, we present additional results at d=16 for these larger architectures, revealing its superior adaptability to higher-dimensional quantization.

11. Effect of Hessian weighted k-means

SqueezeLLM [15] employs Hessian-weighted initialization. Specifically, its objective function is designed to minimize the final task loss caused by quantized weights

$$\delta L = L(W) - L(\hat{W}) \tag{10}$$

We can achieve this objective through Hessian matrix approximation. Assuming that $\hat{W}$ represents a small perturbation on W, we perform a second-order Taylor expansion on Eq. 10 to obtain:

$$\delta L \approx g^T(W - \hat{W}) + \frac{1}{2}(W - \hat{W})^T H(W - \hat{W}) \tag{11}$$

Here, g is the gradient, and H is the Hessian matrix of the weights (the second derivative of the Loss with respect to the weights). Assuming the pretrained model has converged, the gradient can be approximated as zero. Furthermore, we can approximate H by retaining only its diagonal elements (which typically dominate).

$$\delta L \approx \sum_i H_{ii} \|\mathbf{w}_i - \mathbf{c}_{A(i)}\|^2 \tag{12}$$

We observed that this approach significantly benefits LLMs, but fails to demonstrate clear advantages over standard k-means on vision-related tasks, especially after fine-tuning. Thus, we retain standard k-means in other experiments.

12. Detailed quantization settings.

In this section, we listed the detailed quantization settings of our experiments, specifically k and d. Classification tasks (Tab. 10), detection and segmentation tasks (Tab. 11), image generation and nlp tasks (Tab. 12).

13. More results on image generation tasks

13.1. Comparison on visual similarity

To provide a comprehensive evaluation of generation consistency, we introduce visual similarity metrics as complementary assessment criteria. Specifically, we employ

SSVQ, d=8					
k	8	16	32	64	128
MobileNet-v2	22.9	20.7	18.6	16.6	-
MobileNet-v3	23.2	21.3	19.6	18.1	16.7
EfficientNet-lite0	23.0	20.9	19.0	17.2	-
ConvNext-Tiny	23.2	21.3	19.6	18.1	16.8
DeiT-Tiny	23.1	21.1	19.2	17.5	15.8
DeiT-Small	23.2	21.3	19.6	18.1	-
VQ, d=4					
k	32	64	128	256	-
MobileNet-v2	-	20.1	16.6	-	-
MobileNet-v3	23.5	18.7	-	-	-
EfficientNet-lite0	-	20.6	17.2	-	-
ConvNext-Tiny	-	21.2	18.1	-	-
DeiT-Tiny	-	20.8	17.5	-	-
DeiT-Small	-	21.2	18.1	-	-
VQ, d=8					
k	128	256	512	1024	2048
MobileNet-v2	-	-	20.9	16.2	-
MobileNet-v3	19.8	-	-	-	-
EfficientNet-lite0	-		17.8	-	-
ConvNext-Tiny	-	-	25.6	21.6	18.1
DeiT-Tiny	-	24.7	18.6	-	-
DeiT-Small	-	-	-	25.1	20.7

Table 10. k&d settings and corresponding compression ratios of classification tasks (Fig. 5). Notably, results with compression ratios $\leq$16× or unacceptably low accuracy are omitted from the figure.

Models	Methods	k	d	C.R.
Yolo-v9	VQ	64	4	19.67
	SSVQ	16	8	20.47
Gelan-C	VQ	64	4	20.98
	SSVQ	16	8	21.15

Table 11. k&d settings and corresponding bits/weight of detection and segmentation tasks (Tab. 5).

three widely-adopted measures: SSIM (structural similarity), LPIPS (learned perceptual image patch similarity), and PSNR (peak signal-to-noise ratio), maintaining consistency with the evaluation protocol used in EMF [18]. As demonstrated in Tab. 13, our SSVQ framework achieves an average improvement of 10% in visual similarity metrics compared to the standard VQ baseline, highlighting its enhanced capability in preserving visual fidelity.

Methods	k	d	Index bit	Mask bit	Total bit
VQ	1024	4	2.5	0	2.5
VQ	256	4	2	0	2
SSVQ	64	4	1.5	1	2.5
SSVQ	256	8	1	1	2

Table 12. k and d settings and corresponding bits/weight of generation (Tab. 6) and NLP tasks (Tab. 7).

Prompts	Methods	LPIPS↓	SSIM↑	PSNR ↑
Stable Diffusion v1-4				
COCO Prompts	EMF [18]	0.77	0.34	11.1
	VQ	0.56	0.44	12.5
	SSVQ	0.51	0.49	13.4
SD Prompts	EMF [18]	0.66	0.42	12.8
	VQ	0.56	0.46	13.5
	SSVQ	0.49	0.51	14.8
Stable Diffusion v2-1				
COCO Prompts	EMF [18]	0.66	0.39	11.5
	VQ	0.59	0.44	12.7
	SSVQ	0.48	0.50	13.8
SD Prompts	EMF [18]	0.72	0.30	10.9
	VQ	0.64	0.46	13.3
	SSVQ	0.52	0.48	14.4
Stable Diffusion v3				
COCO Prompts	EMF [18]	0.72	0.38	9.6
	VQ	0.59	0.42	10.4
	SSVQ	0.54	0.48	11.0
SD Prompts	EMF [18]	0.75	0.28	6.80
	VQ	0.64	0.34	9.40
	SSVQ	0.58	0.40	9.90

Table 13. Visual similarity results on three stable diffusion models. ↓ means lower is better. ↑ means higher is better

13.2. Visual results

To provide intuitive visual comparisons, Fig. 10 illustrates the qualitative results under extreme low-bit quantization. Our analysis reveals that aggressive quantization in diffusion models leads to three characteristic degradation patterns: (1) structural distortion in scene layout, (2) color bleeding and object disappearance, and (3) object blending artifacts. Notably, our SSVQ framework demonstrates remarkable robustness against these degradation effects, significantly mitigating image quality deterioration even at ultra-low bit rates.

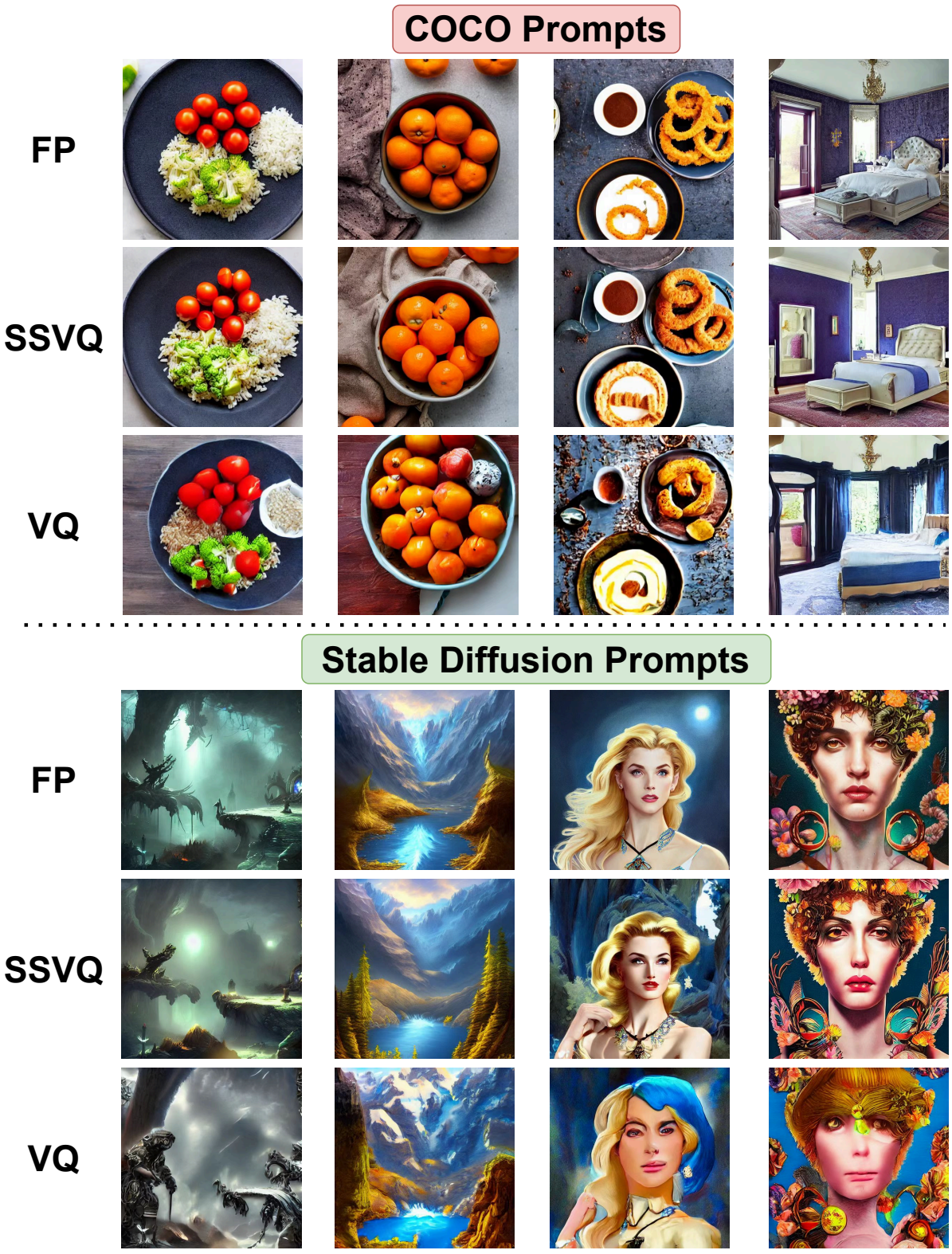

Figure 10. Visualization of image generation results of full-precision and quantized Stable Diffusion models. Qualitative comparisons under COCO and Stable Diffusion prompt styles demonstrate SSVQ's superior generation quality and prompt consistency over conventional VQ.

Models	batch size	lr	lr_mask	epochs	θ	Optimizer
Classification						
MobileNet-V2	256	2e-3	2e-4	10	1	AdamW
MobileNet-V3	256	2e-3	2e-4	10	1	AdamW
EfficientNet-Lite0	256	2e-3	2e-4	10	1	AdamW
ConvNext-Tiny	256	2.5e-4	2.5e-4	10	4	AdamW
DeiT-Tiny	256	2.5e-4	2.5e-4	10	6	AdamW
DeiT-Tiny	256	2.5e-4	2.5e-4	10	4	AdamW
Semantic Segmantation						
DeepLab-V3	8	5e-4	5e-4	100	10	AdamW
Object Dection						
YOLO-V9	16	1e-3	1e-3	10	4	AdamW
Instance Segmantation						
GELAN-C-SEG	32	1e-3	1e-4	10	8	AdamW
Image Generation						
Stable Diffusion v1.4	12	1e-4	2e-4	10000 iters	1	AdamW
Stable Diffusion v2.1	12	1e-4	2e-4	10000 iters	1	AdamW
Stable Diffusion v3	2	1e-4	5e-5	10000 iters	1	AdamW

Table 14. Detailed hyper parameter settings of our experiments

13.3. Training hyper parameters

Detailed training hyper parameters, including batch size, learning rate, threshold, optimizer, and epochs are summarized in Tab. 14.