Precision As You Need: Stochastic Computing Is a Dense Adaptive Quantizer

Haoran Jin[*,†], **Kangqi Zhang**[*], **Jirong Yang**, **Barry Lyu**, **Qiuyi Ding**, **Ruijie Gao**, **Nathan Bleier**

Department of Electrical Engineering and Computer Science, University of Michigan
{allenjin, zhkangqi, yjrcs, barrylyu, dingqy, ruijieg, nbleier}@umich.edu

Abstract

Matrix multiplications dominate the inference cost of modern transformer-based vision models, yet existing efficiency techniques such as post-training quantization and mixed-precision inference are largely limited to the small set of fixed-width formats—INT4, INT8, BF16, and FP16—supported by conventional accelerators. We revisit *stochastic computing* (SC) as a way to lift this constraint: viewed as a *dense adaptive quantizer*, SC controls precision by bit-stream length L rather than a fixed datapath, while each multiplication reduces to a single AND/XNOR gate. We build a GPU library that emulates SC matrix multiplication at scale, exposes stream lengths as first-class kernel arguments, and evaluates SC end-to-end on image classification, object detection and instance segmentation, class-conditional image generation, and visual world-model planning. On top of this substrate, we develop a dynamic per-row mixed-precision policy that assigns stream length per token or group at matched average budget, requires no retraining, and uses the same SC hardware across schedules. Our results show that SC can preserve near-full-precision quality, while mixed precision improves accuracy at a fixed average stream budget. These results provide software-level feasibility evidence that SC can serve as a dense-precision substrate for fine-grained mixed-precision inference on modern vision transformers.

1 Introduction

The computational footprint of modern neural networks has grown faster than the efficiency gains provided by conventional hardware scaling. Across vision, language, generation, and world-modeling workloads [Dosovitskiy et al., 2020, Brown et al., 2020, Touvron et al., 2023, Peebles and Xie, 2023, Ha and Schmidhuber, 2018, Hafner et al., 2023], matrix multiplications dominate inference cost, complicating deployment in latency- and energy-constrained settings, including edge devices, real-time control systems, and power-limited datacenters.

The dominant response has been low-bit arithmetic that trades efficiency at the cost of accuracy. Post-training quantization (PTQ) maps weights and activations to INT8, INT4, or lower precision, exploiting the empirically observed robustness of trained networks to numerical perturbations [Nagel et al., 2021, Dettmers et al., 2022, Frantar et al., 2023, Xiao et al., 2023]. Mixed-precision methods extend this idea by assigning different bit-widths or data types to different layers, tensors, or weight groups, thereby recovering accuracy in cases where a uniform low-precision format is too restrictive [Wang et al., 2019, Dong et al., 2019, Rouhani et al., 2023, Chen et al., 2025b].

Conventional mixed precision, however, is constrained by hardware: bit-parallel accelerators typically provide a distinct datapath or mode for each supported precision, so deployed accelerators expose only

[*]Equal contribution.

[†]Corresponding author: Haoran Jin (allenjin@umich.edu).

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

a small set of formats, such as INT4, INT8, BF16, and FP16. This sparse, hardware-defined precision ladder limits the granularity of precision, thereby constraining the effectiveness of mixed-precision inference.

Stochastic computing (SC) provides a fundamentally different precision substrate: precision is determined by execution time rather than by a fixed hardware format. In SC, a real value is represented by the ratio of 1s in a stochastic bit stream [Gaines, 1969, Alaghi and Hayes, 2013b]. Multiplication of two such streams reduces to a single bitwise AND gate in the unipolar case or XNOR gate in the bipolar case.

SC has three properties well-suited to neural-network inference: (i) **Area and energy efficiency.** A multi-bit multiplier is replaced by a single logic gate, enabling substantial reductions in area and energy relative to binary integer MACs [Alaghi and Hayes, 2013b, Alaghi et al., 2018, Ren et al., 2017, Wu and San Miguel, 2022, Guo et al., 2023, Weller et al., 2021]. (ii) **Dense, hardware-shared precision.** SC precision is governed by the bit-stream length L, which can be *any* positive integer rather than only a conventional word size. The same hardware can compute at different precisions by terminating streams at different lengths, allowing a single accelerator to support a dense set of precision choices and fine-grained mixed-precision schedules. (iii) **Inherent error tolerance.** A bit flip in an SC stream perturbs the decoded value by only $2/L$ in the bipolar encoding we use ($1/L$ for unipolar streams), whereas an error in a high-order bit of a fixed-point or integer word can induce a much larger deviation [Alaghi et al., 2018, Qian et al., 2011]. This property is particularly relevant for unreliable computing environments, including space computing.

Despite these advantages, prior work on SC for deep learning has remained limited in scope. Most studies evaluate small CNNs or simplified transformer models [Ren et al., 2017, Sim et al., 2017, Wu and San Miguel, 2022, Guo et al., 2023]. To our knowledge, no prior work has performed an end-to-end evaluation of SC across modern transformer-based AI workloads or quantitatively demonstrated how its dense precision substrate can be used for fine-grained mixed-precision inference.

This work addresses that gap by treating SC as a *dense adaptive quantizer*. We show that this perspective enables competitive accuracy with substantially reduced compute across four families of modern models. Our contributions are: (i) **A scalable GPU library for SC simulation.** We present, to our knowledge, the first GPU implementation that can effectively emulate SC matrix multiplication at scale. The library implements bit-stream generation, AND/XNOR-based multiplication, and counter-based decoding as fused Triton kernels, and exposes per-row and per-group bit-stream lengths for algorithmic exploration. (ii) **An end-to-end evaluation across vision-centric transformer workloads.** We evaluate SC inference on standard vision tasks—classification, detection, and instance segmentation—as well as diffusion-based image generation and learned world-model rollouts. To our knowledge, this is the first end-to-end study showing that an SC-based compute pipeline can preserve task-level performance across various transformer-based vision workloads. (iii) **A fine-grained mixed-precision algorithm.** We introduce a runtime mixed-precision scheme that assigns a separate bit-stream length to each row in every matrix multiplication, guided by lightweight operator-specific saliency metrics.

The GPU kernel is available at `https://github.com/CrucibleComputingGroup/scmp_kernels`.

2 Related Works

2.1 Stochastic Computing

Stochastic computing (SC) [Gaines, 1969] encodes real values as the density of 1s in probabilistic bit streams, reducing multi-bit multiplication to a single AND/XNOR gate and scaled addition to a multiplexer, and realizing division and common activations such as $\tanh$ with small finite-state machines [Gaines, 1969, Alaghi et al., 2018, Liu et al., 2021]; applications include LDPC decoding and image processing [Alaghi and Hayes, 2013b]. For neural networks [Liu et al., 2021], SC-DCNN [Ren et al., 2017], the SC multiplier of Sim and Lee [2017], and the scalable SC accelerator of Sim et al. [2017] apply SC to CNNs, and recent hardware substrates such as uSystolic [Wu and San Miguel, 2022] and Cambricon-U [Guo et al., 2023] push SC closer to modern accelerator scale.

2.2 Post-training Quantization and Mixed Precision

Post-training quantization (PTQ) compresses pretrained networks by mapping weights and activations to low-bit formats without retraining [Nagel et al., 2021]. For LLMs, OPTQ [Frantar et al., 2023], LLM.int8() [Dettmers et al., 2022], SmoothQuant [Xiao et al., 2023], and AWQ [Lin et al., 2024] enable low-bit transformer inference by addressing weight quantization and activation outliers. For vision and generative models, PTQ4ViT [Yuan et al., 2022], Q-Diffusion [Li et al., 2023a], and Q-DiT [Chen et al., 2025a] adapt PTQ to ViTs, UNet-based diffusion models, and Diffusion Transformers, respectively. Mixed-precision methods further assign different precisions to different layers, operators, or tensors. HAQ [Wang et al., 2019] learns per-layer bit-widths with reinforcement learning and hardware-simulator feedback, while HAWQ-V3 [Yao et al., 2021] and BRECQ [Li et al., 2021] search sensitivity-aware bit-width assignments under hardware constraints. MX formats [Rouhani et al., 2023] share one scale across small blocks of narrow elements and can pair different formats for weights and activations, while MPQ-DM [Feng et al., 2025] and MixDiT [Kim et al., 2025] specialize mixed precision for diffusion models and Diffusion Transformers.

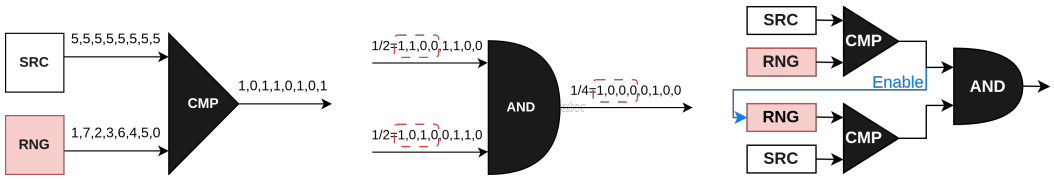
3 Preliminaries and Problem Formulations

3.1 Stochastic Computing Basics

Encoding. Stochastic computing (SC) represents a real value by the probability that each bit of a single-bit binary sequence is 1 [Gaines, 1969, Alaghi and Hayes, 2013b]. A *unipolar* stream encodes $v \in [0,1]$ by a length-L sequence $Z = (z_1, \ldots, z_L) \in \{0,1\}^L$ in which each bit is 1 with probability v. A *bipolar* stream encodes $v \in [-1,1]$ with $\Pr[z_\ell = 1] = (v+1)/2$. Since neural-network activations and weights are signed, we use bipolar encoding throughout.

Bitstream generation. Stochastic number generation (SNG) is implemented in hardware as an integer comparator driven by a random number source [Zhong et al., 2022]. At each cycle ℓ, a pseudo-random number generator (RNG) draws a uniform integer r_ℓ from the same b-bit range as the input, and the comparator emits $z_\ell = 1$ if r_ℓ is below the input value, else $z_\ell = 0$ (Figure 1(a)). For a signed integer $\hat{x} \in [-q_{\max}, q_{\max}]$ with $q_{\max} = 2^{b-1} - 1$ encoding the bipolar value $v = \hat{x}/q_{\max} \in [-1,1]$, the input is first offset onto the unsigned RNG range as $\hat{x} + q_{\max} \in [0,\ 2q_{\max}]$ before comparison, which approximately realizes the bipolar encoding above. Decoding inverts the encoding: from the count $c = \sum_\ell z_\ell$, the estimate of v is $\tilde{v} = (2c - L)/L$ [Gaines, 1969].

The choice of RNG drives both the convergence of $\tilde{v} \to v$ at finite L [Zhong et al., 2022] and the cross-correlation between any two streams produced for the same matrix multiplication; the latter is a major source of SC error [Alaghi and Hayes, 2013b, Wu et al., 2020] and is addressed by the C-BSG construction we adopt (Appendix A.1).



(a) Bitstream generation (b) Bitstream multiplication (c) C-BSG multiplier

Figure 1: Circuit primitives of stochastic computing. **(a)** A SNG implemented as an integer comparator: at each cycle the source value is compared against an RNG draw, emitting a single bit. **(b)** Stochastic multiplication via a bitwise gate: the unipolar product is realized by an AND gate whose output bit-probability factors as the product of the input marginals when the two streams are independent. The red circle denotes early termination. **(c)** Conditional bitstream generation (C-BSG): the second RNG is gated by the first stream so the conditional distribution of z^B matches its marginal, which drives the stochastic cross-correlation toward zero and suppresses correlation-induced bias in the bipolar XNOR multiplier [Wu et al., 2020, Wu and San Miguel, 2022].

SC matrix multiplication. SC enters the inference pipeline only at the integer matrix-multiply step. We assume the inputs to a matrix multiplication $C = AB^\top$ with $A \in \mathbb{R}^{N \times D}$ and $B \in \mathbb{R}^{M \times D}$

have already been quantized by a standard PTQ pipeline [Nagel et al., 2021] with per-row symmetric absmax scales, as in the vector-wise quantization of LLM.int8() [Dettmers et al., 2022] and the per-token and per-channel quantization of SmoothQuant [Xiao et al., 2023]: $s_i^A = \max_d |A_{id}|/q_{\max}$ and $s_j^B = \max_d |B_{jd}|/q_{\max}$ produce b-bit signed integer matrices $\hat{A}, \hat{B}$ with $\hat{A}_{id}, \hat{B}_{jd} \in [-q_{\max}, q_{\max}]$ such that $A_{id} \approx s_i^A \hat{A}_{id}$ and $B_{jd} \approx s_j^B \hat{B}_{jd}$. In the subsequent integer matrix multiplication, an SC primitive replaces the multi-bit integer multiplier at each inner-dimension position. Let D denote the inner dimension shared by $\hat{A}$ and $\hat{B}$ and let $d \in \{1, \ldots, D\}$ index it. Map each integer to a normalized bipolar value $u_{id}^A = \hat{A}_{id}/q_{\max} \in [-1, 1]$, $u_{jd}^B = \hat{B}_{jd}/q_{\max} \in [-1, 1]$, and encode each as a bipolar stream of length L. For two such streams the bipolar product is captured by the bitwise XNOR gate: in expectation, the rescaled XNOR count over L cycles equals $u_{id}^A u_{jd}^B$ [Alaghi and Hayes, 2013b] (the analogous unipolar AND construction is illustrated in Figure 1(b)). Summing over the inner dimension d and rescaling by the per-row scales then gives the SC dot-product estimator

$$\widehat{C}_{ij} \;=\; s_i^A \, s_j^B \, q_{\max}^2 \left[\frac{2}{L} \sum_{\ell=1}^{L} \sum_{d=1}^{D} \mathrm{XNOR}\big(z_{id,\ell}^A,\, z_{jd,\ell}^B\big) \;-\; D \right], \qquad \mathrm{XNOR}(a,b) \equiv 1 - (a \oplus b), \tag{1}$$

which, for independent paired streams, is unbiased for the quantized product. Throughout this paper, we generate the paired streams using the state-of-the-art *conditional bitstream generation* (C-BSG) construction [Wu et al., 2020, Wu and San Miguel, 2022], which drives the stochastic cross-correlation of each stream pair toward zero and so suppresses the correlation-induced bias of (1); we defer the full derivation, including the formal stochastic cross-correlation definition, to Appendix A.1.

Precision and early termination. The stream length L is the precision knob. A longer stream resolves each operand more finely, so the error of (1) shrinks as L grows; Section 3.2 quantifies this error and the network's tolerance to it. Crucially, L can be *any* positive integer, not only a power of two, so the same datapath exposes a dense, hardware-shared precision ladder; a scheduler can *terminate early* for less salient outputs by stopping at $L' < L$, trading quality for proportional time and energy savings. As a concrete example, an INT8 operand ($b = 8$) reaches roughly 8-bit resolution at $L = 2^b = 256$ cycles; stopping at $L' = 192$ leaves $\log_2 L' \approx 7.58$ effective bits, an operating point between INT7 and INT8 that the same datapath reaches simply by stopping early.

3.2 Error Tolerance in Neural Networks

Neural networks routinely tolerate bounded numerical error from quantization, pruning, low-rank approximation, and noisy hardware, so their workloads are natural candidates for SC. To quantify the relevant tolerance, we replace each exact output cell $Y_{ij} = \langle A_{i,:}, B_{j,:} \rangle$ with a Gaussian-perturbed surrogate $\widehat{Y}_{ij} = Y_{ij}(1 + \alpha Z_{ij})$, $Z_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$, with α calibrated as a function of stream length L to match measured SC error on representative tensors. We apply this surrogate to all DiT-XL/2 [Peebles and Xie, 2023] matmuls (Table 1).

Table 1: Gaussian surrogate of SC error on every DiT-XL/2 matmul: noise level α calibrated per stream length L, and the resulting latent MSE (before VAE decoding), image MSE, and image PSNR (after decoding).

stream length L (sc_len)	equiv. INT bits $\log_2 L$	noise α	latent MSE	image MSE	image PSNR (dB)
32	5.00	0.0726	3.2×10^{-2}	4.8×10^{-3}	23.2
64	6.00	0.0363	8.1×10^{-3}	1.3×10^{-3}	29.0
96	6.58	0.0242	4.0×10^{-3}	7.2×10^{-4}	31.4
128	7.00	0.0182	3.8×10^{-3}	6.4×10^{-4}	32.0
192	7.58	0.0121	9.7×10^{-4}	2.0×10^{-4}	37.0
256	8.00	0.0091	7.9×10^{-4}	1.6×10^{-4}	37.9

The sweep confirms that the neural workload has a usable fluctuation margin for SC: $L = 256$ gives 37.9 dB PSNR, $L = 192$ gives 37.0 dB, and $L = 128$/96 still give 32.0/31.4 dB; quality drops at

$L = 64$ and degrades substantially for $L \leq 32$. This tolerance is the key hardware opportunity. Since SC exposes precision as the runtime stream length L, including non-power-of-two operating points such as $L = 96$ and $L = 192$ without changing the circuit, we can spend longer streams only where the model is sensitive.

4 Methodology

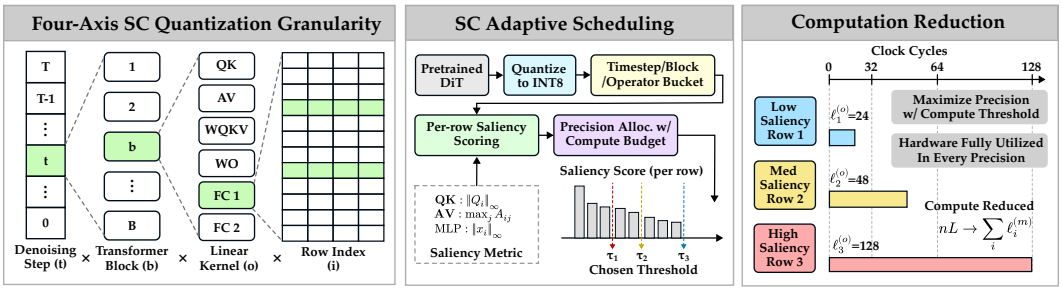


Figure 2: Overview of our adaptive fine-grained quantization and scheduling flow.

4.1 Hardware Model

SC replaces a multi-bit multiplier with a one-bit gate (AND for unipolar, XNOR for bipolar) and pays precision temporally: a length-L stream runs the same datapath for L cycles.

Gate-level advantage. We synthesize four multipliers (SC vs. binary, signed vs. unsigned) in FreePDK45 [Stine et al., 2007] at 400 MHz with the NanGate45 cell library at the typical corner and the nominal 1.1 V, and measure post-synthesis power with switching activity annotated from uniform random inputs applied every cycle; Table 2 reports area, power, their ratios, and the per-operation energy reduction at $L = 2^k$. The unipolar AND is $268\times$ smaller and $453\times$ lower-power than an unsigned INT8 multiplier, and the bipolar XNOR is $156\times$ smaller and $399\times$ lower-power than its signed counterpart. After the L-cycle penalty, the SC gate alone is still 1.6–$1.8\times$ lower-energy at k=8 and 6–7$\times$ at k=6.

Table 2: SC vs. binary multiplier comparison at 400 MHz. Energy/op reduction at L=2^k is the INT8 multiplier's energy per operation divided by the SC gate's energy over 2^k cycles, $P_{\mathrm{INT8}}/(2^k P_{\mathrm{SC}})$.

	SC gate		INT8 mult.		INT8/SC ratio		Energy/Op Reduction			
Pair	Area (μm^2)	Power (μW)	Area (μm^2)	Power (μW)	Area	Power	$k=8$	$k=7$	$k=6$	$k=5$
AND/UMUL	1.06	0.37	285.4	168.7	268×	453×	1.77×	3.54×	7.08×	14.2×
XNOR/MUL	1.60	0.63	249.2	249.7	156×	399×	1.56×	3.13×	6.25×	12.5×

Full-accelerator evidence. Table 2 prices arithmetic only. To include bit-stream generation, accumulation, control, and data movement, we run ViT inference on uSystolic [Wu and San Miguel, 2022], a published unary systolic-array accelerator, with its cycle-accurate simulator and energy model. Against INT8 on the binary systolic array that the same simulator models, uniform SC at L=128 uses $2.58\times$ less on-chip energy and SC MP at $\bar{L}$=64 uses $4.60\times$ less. The gate-level gap shrinks but persists at chip scale: Cambricon-U reports $2.1\times$ lower energy and $2.2\times$ smaller area than an INT8 binary accelerator [Guo et al., 2023].

From compute cycles to energy and latency. For matmul m with per-row stream length $\ell_i^{(m)}$, we take SC compute cost as $C_{\mathrm{SC}} \propto \sum_m \sum_i \ell_i^{(m)}$. Because every stream cycle drives the same gate, SC compute energy is linear in C_{SC}; latency is linear in C_{SC} at fixed lane count when rows of different lengths are packed onto the lanes without idle cycles, which we assume. Under this assumption, two

schedules with matched $\sum \ell_i$ have matched compute energy and latency, and reducing the average stream length from L to $\bar{\ell}$ yields up to $L/\bar{\ell}$ savings on both axes (e.g., $\bar{\ell}$=128 or 64 at L=256 gives $2\times$ or $4\times$). Section 4.4 spends this budget by shortening streams on less sensitive rows.

4.2 GPU Kernel for SC Matrix Multiplication

GPU execution path. Existing CPU SC simulators are too slow to insert into every transformer matmul, while the C-BSG datapath is a regular tiled AND-and-popcount computation that fits GPU execution. We implement the uSystolic-style C-BSG temporal-unary multiplier from Section 3.1 as fused Triton kernels [Tillet et al., 2019]; in our benchmark, this path is about $95{,}000\times$ faster than direct CPU SC simulation. The kernel treats stream length as data, so per-row length vectors can be changed at runtime without recompilation.

Datapath: cumulative-indicator C-BSG. Each matmul begins with fused quantization: operands are mapped to the int8 grid ($q_{\max} = 127$), converted to one of $V = 257$ comparison boundaries, and emitted with row scale and sign in one pass. The C-BSG stream is represented by a precomputed cumulative-indicator table $K[b, \ell] = \mathbf{1}[r_\ell \le b]$ over Sobol-8 values [Sobol', 1967, Joe and Kuo, 2008, Li et al., 2024]; the compact kernel indexes K by the weight and activation boundary tensors and accumulates the bipolar SC estimator as tiled AND-and-popcounts over $D\times L$ binary comparisons. The table is built once per layer at $L_{\max} = 2^8$, cached on device, and reused by taking prefixes for shorter streams. This same launch receives a row-length vector $\boldsymbol{\ell} \in \mathbb{Z}^N_{\ge 0}$, so row i stops at $\boldsymbol{\ell}_i$ and different rows can run at different effective precisions for Section 4.4.

4.3 Sensitivity Analysis

Probing where SC error matters. SC trades a multi-bit multiplier for one-bit logic, but its arithmetic is only statistically correct: the C-BSG kernel of Section 4.2 suppresses correlation bias, yet reaching fixed-point accuracy can require stream lengths in the hundreds or thousands [Wu and San Miguel, 2022, Guo et al., 2023]. Paying this cost uniformly is wasteful—prior mixed-precision work finds quantization sensitivity concentrated in a few layers and layer types [Zhao et al., 2024] and in a few large-magnitude activation channels and attention heads [Kim et al., 2025]. We therefore probe *where* SC precision must be spent. For each operator class $o \in \mathcal{O} = \{\text{qk}, \text{av}, \text{qkv_proj}, \text{out_proj}, \text{fc1}, \text{fc2}\}$ and block $b \in \{0, \dots, B-1\}$, we route only that matmul through C-BSG at stream length L, leaving all others in full precision, and measure

$$\mathrm{S}(o, b, L) \;=\; \mathbb{E}_{x\in\mathcal{D}} \left\| h^\star(x) - \widehat{h}^{(o,b,L)}(x) \right\|_2, \tag{2}$$

where $h^\star$ and $\widehat{h}^{(o,b,L)}$ are the full-precision and single-SC backbone feature tensors on a small calibration set $\mathcal{D}$.

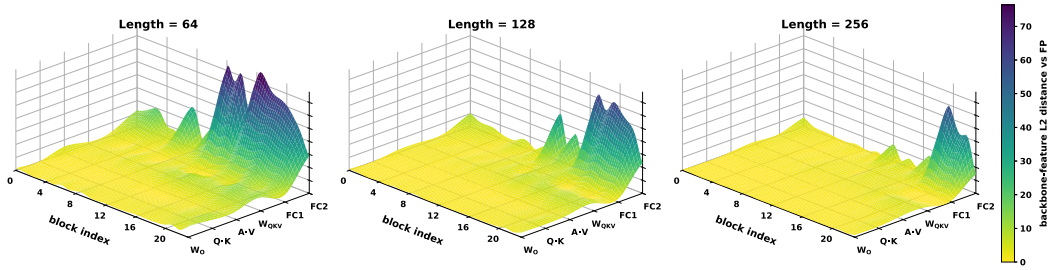


Figure 3: SC sensitivity is highly non-uniform across operator class, block depth, and stream length: replacing a single (o, b) matmul with SC perturbs the final ViT-L/14 feature far more for late FC2 blocks than for any other tile, motivating the mixed-precision policy of Section 4.4.

Case study on ViT-L/14 + ImageNet. We instantiate the probe on a pretrained DINOv2 ViT-L/14 backbone ($B = 24$, $|\mathcal{O}| = 6$) with $|\mathcal{D}| = 100$ ImageNet images at three stream lengths $L \in \{64, 128, 256\}$, yielding the sweep in Figure 3. The results reveal two patterns. First, FC2

dominates the high-sensitivity region across stream lengths, indicating SC error is far from uniform across operator classes. Second, within FC2 sensitivity concentrates in later blocks: at $L = 64$, late-FC2 accounts for 19 of the top-20 most sensitive (o, b) tiles. These motivate reserving binary fallback for the most sensitive tiles and using the mixed-precision policy of Section 4.4 to allocate longer SC streams to sensitive operators and shorter streams to less sensitive ones.

4.4 Per-Row Mixed-Precision SC

Calibration mode. Section 4.2 exposes stream length as a per-row runtime argument, so mixed precision reduces to choosing the stream length assigned to each output row of each SC matmul. We make this choice through an offline *calibration mode*: for each target average stream length $\bar{L}$, the calibrator runs a small set of FP-teacher trajectories, replays each selected matmul with uniform SC at several candidate stream lengths, and records the row-wise deviation. The result is a JSON table of metric thresholds. Inference only evaluates the same lightweight row metrics, looks up the calibrated thresholds, and passes the resulting row-length vector to the SC kernel.

Row metrics. For each calibrated operator, the row metric is chosen from tensors already present in the forward pass. Motivated by the activation-magnitude saliency of AWQ [Lin et al., 2024] and SmoothQuant [Xiao et al., 2023], which measure it per input channel to protect or rescale weights, for QK we use the row-wise query magnitude, $g_i^{\mathrm{QK}} = \|Q_i\|_\infty$. For AV, we use the peak attention probability, $g_i^{\mathrm{AV}} = \max_j A_{ij}$. For INPUT_PROJ, PROJ, MLP_FC1, and MLP_FC2, we use the row-wise activation magnitude, $g_i = \|X_i\|_\infty$. We use the maximum magnitude $a_{\max}$ rather than an average because the per-row SC quantizer is scaled by the row range: the largest entry sets the comparison boundary scale and also bounds the largest contribution any coordinate can make to the dot-product error. Thus $a_{\max}$ is a cheap upper-envelope proxy for row dynamic range, and it is already available from the fused per-row scale extraction. Within each operator call, metrics are normalized to $[0, 1]$ as

$$\tilde{g}_i = \frac{g_i - \min_j g_j}{\max_j g_j - \min_j g_j}.$$

Rows with larger $\tilde{g}_i$ are treated as more precision-sensitive and therefore receive longer streams after thresholding.

Threshold fitting. Let $L_1 > L_2 > \cdots > L_K$ be the stream-length ladder supplied to the calibrator. For each operator, timestep bucket, and layer-depth bucket, the calibrator measures row-wise cosine error against the FP teacher at every L_k and solves a budgeted Lagrangian assignment:

$$k_i(\lambda) = \arg\min_k \big(e_{ik} + \lambda L_k\big),$$

where e_{ik} is the measured FP-vs-uniform-SC error. Binary search on λ matches the target budget, and the resulting level counts are converted into non-uniform thresholds

$$1 \geq \tau_1 \geq \tau_2 \geq \cdots \geq \tau_{K-1} \geq 0.$$

At inference, row i receives the first level whose threshold it exceeds, $k^\star(i) = \min\{k : \tilde{g}_i \geq \tau_k\}$, or the shortest stream if it falls below all thresholds; the kernel then uses $\ell_i = L_{k^\star(i)}$. The JSON table is keyed by operator, timestep bucket, and layer bucket, with operator-level fallback for sparse buckets.

5 Experiments

5.1 Classification

Setup. We evaluate DINOv2 ViT-L/14 [Oquab et al., 2023] on full ImageNet-1k [Deng et al., 2009] val (N=50,000), comparing SC—with and without our residual *compensation* (Appendix A.2)—against the *QwT* [Fu et al., 2025] baseline (INT quantization [Li et al., 2023b] with closed-form residual compensation). For SC we further compare uniform-L against mixed precision (*MP*) at matched bit budget $\sum_i \ell_i$; for this classifier, MP assigns one stream length to each (op, block) cell through a calibration search, a coarser granularity than the per-row policy of Section 4.4. All SC runs keep the 30 most-sensitive (op, block) cells from the Section 4.3 profile in FP. Bit budgets follow the W/A convention of Chen et al. [2025a].

Table 3: **ImageNet-1k classification (full val, N=50,000).** *W/A* is weight/activation bits: the underlying INT setting for QwT; for SC, $W{=}A{=}\log_2 \bar{L}$ with $\bar{L}$ the average per-row stream length. SC: stochastic computing; MP: mixed-precision stream lengths at the same average budget; QwT: INT quantization with the closed-form residual compensation of Fu et al. [2025]; compensation: its SC counterpart (Appendix A.2).

Method	W/A	top-1 (%)	top-5 (%)	Δ vs FP
FP (DINOv2 ViT-L/14)	16/16	86.40	97.65	0.00
QwT	8/8	84.57	96.56	−1.83
SC uniform	8/8	85.57	97.40	−0.83
SC uniform + compensation	8/8	85.70	97.47	−0.70
SC uniform	$\log_2 192/\log_2 192$	84.67	97.11	−1.73
SC uniform + compensation	$\log_2 192/\log_2 192$	85.42	97.43	−0.98
SC MP	$\log_2 192/\log_2 192$	85.07	97.23	−1.33
SC MP + compensation	$\log_2 192/\log_2 192$	85.47	97.40	−0.93
QwT	4/8	83.48	96.37	−2.92
SC uniform	7/7	78.94	94.61	−7.46
SC uniform + compensation	7/7	82.87	96.46	−3.53
SC MP	7/7	79.84	94.90	−6.56
SC MP + compensation	7/7	83.26	96.45	−3.14

Results. At 8/8, SC + compensation exceeds QwT by 1.13 pt; at 7/7, SC MP + compensation comes within 0.22 pt of QwT at 4/8, a different weight/activation split (Table 3). MP beats uniform-L at matched average budget by recovering the asymmetry Section 4.3 flagged—raising MLP_FC2 to L=192 in every active block at $\bar{L}$=128 (L=256 in 40% of blocks at $\bar{L}$=192) and letting the cheap projections absorb $L{\in}\{64, 96\}$ in ∼36% of blocks. Compensation lifts SC by +0.13 pt at L=256 up to +3.93 pt at L=128. The closed-form fit of Fu et al. [2025] carries over to SC because our seed-fixed Sobol streams make the per-element error a fixed function of (w, x), as deterministic rounding does for INT, so the residual map fitted at calibration is the one met at inference (Appendix A.2). The 30 FP cells compensate for one quantization scale per row, which compresses small values in wide inputs such as the 4096-wide input of MLP_FC2: with 128-element quantization groups, every matmul runs in SC, and uniform SC at 8/8 comes within 0.4 points of INT8 with the same groups (Appendix A.3).

5.2 Object Detection and Instance Segmentation

Setup. We evaluate EVA-01 ViT-g + cascade Mask R-CNN [Fang et al., 2023, Li et al., 2022, He et al., 2017, Cai and Vasconcelos, 2021] on COCO [Lin et al., 2014] val (N=5000, square-padded to 1024^2, soft-NMS, BEiT [Bao et al., 2021] pos. interp.). SC rows skip the 30 most-sensitive of 240 (op, block) tiles to FP (Section 4.3); MP allocates within the active set per Section 4.4. We compare SC (uniform-L, MP, ± compensation) against QwT-INT [Fu et al., 2025] at matched $W{=}A$.

Table 4: **COCO val object detection & instance segmentation, EVA-01 ViT-g + cascade Mask R-CNN (N=5000, 1024^2).** *W/A* as in Table 3. AP_b/AP_m: box/mask average precision.

Method	W/A	AP_b	AP_m	Δ_b/Δ_m
FP (EVA-01 ViT-g)	16/16	62.06	52.39	0.00/ 0.00
QwT	8/8	61.20	51.59	−0.86/ − 0.80
SC uniform	8/8	61.98	52.18	−0.08/ − 0.21
SC uniform	$\log_2 192/\log_2 192$	61.69	51.85	−0.37/ − 0.54
SC uniform + compensation	$\log_2 192/\log_2 192$	61.82	52.06	−0.24/ − 0.33
QwT	7/7	61.13	51.51	−0.93/ − 0.88
SC uniform	7/7	58.84	48.68	−3.22/ − 3.71
SC uniform + compensation	7/7	59.72	49.81	−2.34/ − 2.58
SC MP	7/7	60.45	50.76	−1.61/ − 1.63

Table 5: **ImageNet class-conditional generation with DiT-XL/2,** 256×256**.** All rows use the same sampler, 50 denoising timesteps, CFG=4.0, and 10,000 generated images (10 per ImageNet class). $\bar{L}$ is the realized average stream length over SC-executed matmuls, and KID is reported as $10^3\times$KID. IS: Inception Score; FID and sFID: Fréchet Inception Distance on pooled and spatial features; KID: Kernel Inception Distance.

Method	$\bar{L}$	IS↑	sFID↓	KID$\times10^3$ ↓	FID↓
FP	–	285.95	24.75	13.47±1.24	24.81
SC MP	64	279.76	19.93	9.77±1.05	17.33
SC uniform	96	284.62	20.91	11.11±1.13	18.47

Results. SC uniform at $8/8$ is essentially FP-equivalent on boxes (61.98 vs. 62.06 AP_b) and clearly outperforms QwT $8/8$ (61.20). At the $\log_2192$ tier, compensation gives a small but consistent gain over raw uniform SC (61.82/52.06 vs. 61.69/51.85 AP_b/AP_m). The harder regime is $7/7$: uniform SC drops to 58.84/48.68, while compensation recovers $+0.88/+1.13$ AP and MP recovers $+1.61/+2.08$ AP, making MP the stronger lever at this budget.

5.3 Image Generation

Setup. We evaluate SC on class-conditional ImageNet [Deng et al., 2009] generation with DiT-XL/2 at 256×256 resolution [Peebles and Xie, 2023]. All attention and MLP matrix multiplications are routed through the GPU C-BSG kernel of Section 4.2 except 5 of the 168 eligible (op, block) cells (3.0% of cells, 8.1% of eligible matmul MACs), which run exact W8A8; the quantization and stream-length schedules follow the same bit-budget convention used in Sections 5.1 and 4.4. All rows use the same sampler with 50 denoising timesteps and classifier-free guidance scale CFG=4.0. We generate 10,000 images in total, with 10 samples for each of the 1000 ImageNet classes. We report IS, sFID, KID [Bińkowski et al., 2021], and FID against the ImageNet validation reference set, and include the realized average stream length to make the compute budget explicit.

Result. Table 5 summarizes the image-generation results. The comparison is intended to test whether the SC schedules that preserve recognition accuracy also maintain generative sample quality at the reported stream budgets. At CFG=4.0, both SC schedules lower FID by 26–30% and KID by 18–27% relative to FP, while IS falls by at most 2.2%. This guidance scale is well above the scale of 1.5 behind DiT's best reported FID, and DiT notes that larger guidance scales increase visual fidelity and decrease sample diversity [Peebles and Xie, 2023]. We hypothesize that SC noise restores part of the diversity that strong guidance removes, which FID and KID reward. Because Table 5 covers a single guidance scale, this remains a hypothesis. Qualitative samples are shown in Appendix A.4, Figure 4.

5.4 Visual World-Model Planning

Setup. We test SC's feasibility in a closed-loop visual planner, where backbone errors compound through a learned dynamics model over long rollouts. We SC-quantize the full Dino-WM [Zhou et al., 2024] stack on `wall_single` (2D goal-reaching): a pretrained DINOv2 ViT-S/14 encoder and a 6-block latent predictor. We keep the $12/60$ most-sensitive encoder tiles and $7/36$ predictor tiles in FP and SC-quantize the rest. Planning uses CEM-MPC with the same budget for all rows of Table 6 (n_{evals}=30 episodes, max_iter=20, 900 samples, seed-fixed); SR resolution is $1/30\approx0.033$.

SC stays within sampling noise of FP at both budgets. At the ∼8-bit tier, all SC configurations reach FP's 0.833 SR plateau, and both uniform variants reach their plateau two CEM iterations earlier than FP. SC MP at $\log_2192$ reaches 0.933, solving three additional episodes within the 20-iteration horizon. With 30 single-seed episodes, the binomial standard deviation of the number of solved episodes at SR ≈ 0.83 is about two episodes, so we do not treat this gain as significant. At the 7-bit tier, uniform SC solves one episode fewer than FP (0.800) and SC MP matches FP (0.833). The direction matches Table 3, where MP improves on uniform SC at $7/7$.

Table 6: **Visual planning success rate on Dino-WM `wall_single`, matched CEM budget.** SR is the fraction of 30 episodes solved within 20 MPC steps; bit budget follows Table 3. SR: success rate; CEM: cross-entropy method; Plateau iter: the CEM iteration at which SR reaches its final value.

Method	W/A	SR	Plateau iter	Δ vs FP
FP (Dino-WM, ViT-S/14 enc. + 6-block pred.)	$16/16$	0.833	5	0.000
SC uniform	$8/8$	0.833	3	0.000
SC uniform	$\log_2 192/\log_2 192$	0.833	3	0.000
SC MP	$\log_2 192/\log_2 192$	0.933	16	$+0.100$
SC uniform	$7/7$	0.800	16	-0.033
SC MP	$7/7$	0.833	13	0.000

6 Discussion

Limitations. Our evaluation covers only a limited set of configurations: for each task we use representative backbones and a small number of precision budgets, rather than an exhaustive sweep over models, seeds, and more aggressive low-bit settings. Thus, our results establish SC's feasibility in the evaluated regimes, but do not yet show how robust the conclusions are across a broader configuration space. The evaluated pipeline is a hybrid of SC and post-training quantization: operands come from a standard PTQ front end (Section 3.1), the most sensitive (op, block) matmuls stay outside SC, and the compensated rows adapt the residual fit of QwT [Fu et al., 2025]. The reported accuracies are those of this hybrid; with 128-element quantization groups, classification runs every matmul in SC (Appendix A.3). Generation and planning are each evaluated on one model and one task, DiT-XL/2 on ImageNet and Dino-WM on `wall_single` with 30 single-seed episodes. Broader generative and planning workloads are future work. In addition, the hardware benefits are estimated from gate-level synthesis and a cycle-accurate simulator rather than measured on silicon.

Broader impacts. This work supports broader exploration of SC for modern AI inference. Since specialized hardware requires substantial design effort, software-level feasibility evidence is important for deciding which arithmetic paradigms are worth pursuing in silicon. Our results suggest that dense, temporally controlled SC precision is a promising direction for future accelerator design. Cheaper inference also lowers the cost of running the same models for harmful uses.

7 Conclusion

This paper argues that stochastic computing is best viewed as a *dense adaptive quantizer* for modern transformer inference: precision is controlled by stream length rather than constrained to a small set of fixed word sizes. Our results highlight a simple principle for adaptive inference: precision should follow sensitivity, especially on COCO, where mixed precision gains 2.08 mask AP over uniform SC at the same average stream budget. To make this concrete at scale, we contribute a fused GPU library for SC matrix multiplication on modern vision transformers and a per-row mixed-precision algorithm that turns row-level sensitivity into a calibrated stream-length budget without retraining. Together, these results position SC as a viable substrate for future adaptive-precision accelerators, where a simple shared datapath can expose a much denser precision space than conventional fixed-format arithmetic.

Acknowledgments and Disclosure of Funding

This work was funded by the University of Michigan. The authors declare no competing interests.

References

Armin Alaghi and John P Hayes. Exploiting correlation in stochastic circuit design. In *2013 IEEE 31st International Conference on Computer Design (ICCD)*, pages 39–46. IEEE, 2013a.

Armin Alaghi and John P. Hayes. Survey of stochastic computing. *ACM Transactions on Embedded Computing Systems*, 12(2s):92:1–92:19, May 2013b. doi: 10.1145/2465787.2465794.

Armin Alaghi, Weikang Qian, and John P. Hayes. The promise and challenge of stochastic computing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(8):1515–1531, 2018. doi: 10.1109/TCAD.2017.2778107.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

Mikołaj Bińkowski, Danica J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans, 2021. URL `https://arxiv.org/abs/1801.01401`.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: High quality object detection and instance segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(5):1483–1498, 2021. doi: 10.1109/TPAMI.2019.2956516.

Lei Chen, Yuan Meng, Chen Tang, Xinzhu Ma, Jingyan Jiang, Xin Wang, Zhi Wang, and Wenwu Zhu. Q-DiT: Accurate post-training quantization for diffusion transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 28306–28315, June 2025a.

Yuzong Chen, Ahmed F. AbouElhamayed, Xilai Dai, Yang Wang, Marta Andronic, George A. Constantinides, and Mohamed S. Abdelfattah. Bitmod: Bit-serial mixture-of-datatype llm acceleration, 2025b. URL `https://arxiv.org/abs/2411.11745`.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems*, volume 35, pages 30318–30332, 2022. doi: 10.48550/arXiv.2208.07339. URL `https://proceedings.neurips.cc/paper_files/paper/2022/hash/c3ba4962c05c49636d4c6206a97e9c8a-Abstract-Conference.html`.

Zhen Dong, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 293–302, October 2019. doi: 10.1109/ICCV.2019.00038.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva: Exploring the limits of masked visual representation learning at scale. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19358–19369, 2023.

Weilun Feng, Haotong Qin, Chuanguang Yang, Zhulin An, Libo Huang, Boyu Diao, Fei Wang, Renshuai Tao, Yongjun Xu, and Michele Magno. MPQ-DM: Mixed precision quantization for extremely low bit diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 16595–16603, 2025. doi: 10.1609/aaai.v39i16.33823.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tcbBPnfwxS.

Minghao Fu, Hao Yu, Jie Shao, Junjie Zhou, Ke Zhu, and Jianxin Wu. Quantization without tears. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 4462–4472, 2025.

B. R. Gaines. Stochastic computing systems. In Julius T. Tou, editor, *Advances in Information Systems Science*, volume 2, pages 37–172. Springer, Boston, MA, 1969. doi: 10.1007/978-1-4899-5841-9_2.

Hongrui Guo, Yongwei Zhao, Zhangmai Li, Yifan Hao, Chang Liu, Xinkai Song, Xiaqing Li, Zidong Du, Rui Zhang, Qi Guo, Tianshi Chen, and Zhiwei Xu. Cambricon-U: A systolic random increment memory architecture for unary computing. In *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 424–437, Toronto, ON, Canada, 2023. Association for Computing Machinery. doi: 10.1145/3613424.3614286.

David Ha and Jürgen Schmidhuber. World models, 2018.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2023.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

Stephen Joe and Frances Y. Kuo. Constructing sobol sequences with better two-dimensional projections. *SIAM Journal on Scientific Computing*, 30(5):2635–2654, 2008. doi: 10.1137/070709359.

Daeun Kim, Jinwoo Hwang, Changhun Oh, and Jongse Park. MixDiT: Accelerating image diffusion transformer inference with mixed-precision MX quantization. *IEEE Computer Architecture Letters*, 24(1):141–144, 2025. doi: 10.1109/LCA.2025.3560786.

Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-Diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 17535–17545, October 2023a.

Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. In *European conference on computer vision*, pages 280–296. Springer, 2022.

Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. BRECQ: Pushing the limit of post-training quantization by block reconstruction. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=POWv6hDd9XH.

Zexi Li, Haoran Jin, Kuncai Zhong, Guojie Luo, Runsheng Wang, and Weikang Qian. Scgen: A versatile generator framework for agile design of stochastic circuits. In *2024 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6, 2024. doi: 10.23919/DATE58400.2024.10546649.

Zhikai Li, Junrui Xiao, Lianwei Yang, and Qingyi Gu. Repq-vit: Scale reparameterization for post-training quantization of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17227–17236, 2023b.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device llm compression and acceleration. In *Proceedings of Machine Learning and Systems*, volume 6, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/hash/42a452cbafa9dd64e9ba4aa95cc1ef21-Abstract-Conference.html.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

Yidong Liu, Siting Liu, Yanzhi Wang, Fabrizio Lombardi, and Jie Han. A survey of stochastic computing neural networks for machine learning applications. *IEEE Transactions on Neural Networks and Learning Systems*, 32(7):2809–2824, July 2021. doi: 10.1109/TNNLS.2020.3009047.

Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart van Baalen, and Tijmen Blankevoort. A white paper on neural network quantization, 2021.

Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*, volume 63 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1992. ISBN 9780898712957. doi: 10.1137/1.9781611970081.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4195–4205, October 2023.

Weikang Qian, Xin Li, Marc D. Riedel, Kia Bazargan, and David J. Lilja. An architecture for fault-tolerant computation with stochastic logic. *IEEE Transactions on Computers*, 60(1):93–105, 2011. doi: 10.1109/TC.2010.202.

Ao Ren, Zhe Li, Caiwen Ding, Qinru Qiu, Yanzhi Wang, Ji Li, Xuehai Qian, and Bo Yuan. Sc-dcnn: Highly-scalable deep convolutional neural network using stochastic computing. In *Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '17, page 405–418, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450344654. doi: 10.1145/3037697.3037746. URL `https://doi.org/10.1145/3037697.3037746`.

Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, Stosic Dusan, Venmugil Elango, Maximilian Golub, Alexander Heinecke, Phil James-Roxby, Dharmesh Jani, Gaurav Kolhe, Martin Langhammer, Ada Li, Levi Melnick, Maral Mesmakhosroshahi, Andres Rodriguez, Michael Schulte, Rasoul Shafipour, Lei Shao, Michael Siu, Pradeep Dubey, Paulius Micikevicius, Maxim Naumov, Colin Verrilli, Ralph Wittig, Doug Burger, and Eric Chung. Microscaling data formats for deep learning, 2023.

Hyeonuk Sim and Jongeun Lee. A new stochastic computing multiplier with application to deep convolutional neural networks. In *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6, 2017. doi: 10.1145/3061639.3062290.

Hyeonuk Sim, Dong Nguyen, Jongeun Lee, and Kiyoung Choi. Scalable stochastic-computing accelerator for convolutional neural networks. In *Proceedings of the 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 696–701, Chiba, Japan, 2017. doi: 10.1109/ASPDAC.2017.7858405.

I. M. Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, January 1967. doi: 10.1016/0041-5553(67)90144-9.

James E Stine, Ivan Castellanos, Michael Wood, Jeff Henson, Fred Love, W Rhett Davis, Paul D Franzon, Michael Bucher, Sunil Basavarajaiah, Julie Oh, et al. Freepdk: An open-source variation-aware design kit. In *2007 IEEE international conference on Microelectronic Systems Education (MSE'07)*, pages 173–174. IEEE, 2007.

Philippe Tillet, H. T. Kung, and David Cox. Triton: an intermediate language and compiler for tiled neural network computations. In *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*, MAPL 2019, page 10–19, New

York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450367196. doi: 10.1145/3315508.3329973. URL https://doi.org/10.1145/3315508.3329973.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models, 2023.

Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. HAQ: Hardware-aware automated quantization with mixed precision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8612–8620, June 2019. doi: 10.1109/CVPR.2019.00881.

Dennis D. Weller, Nathaniel Bleier, Michael Hefenbrock, Jasmin Aghassi-Hagmann, Michael Beigl, Rakesh Kumar, and Mehdi B. Tahoori. Printed stochastic computing neural networks. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 914–919, 2021. doi: 10.23919/DATE51398.2021.9474254.

Di Wu and Joshua San Miguel. uSystolic: Byte-crawling unary systolic array. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 12–24, April 2022. doi: 10.1109/HPCA53966.2022.00010.

Di Wu, Jingjie Li, Ruokai Yin, Hsuan Hsiao, Younghyun Kim, and Joshua San Miguel. Ugemm: Unary computing architecture for gemm applications. In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, pages 377–390, 2020. doi: 10.1109/ISCA45697.2020.00040.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 38087–38099. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/xiao23c.html.

Zhewei Yao, Zhen Dong, Zhangcheng Zheng, Amir Gholami, Jiali Yu, Eric Tan, Leyuan Wang, Qijing Huang, Yida Wang, Michael Mahoney, and Kurt Keutzer. HAWQ-V3: Dyadic neural network quantization. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 11875–11886. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/yao21a.html.

Zhihang Yuan, Chenhao Xue, Yiqi Chen, Qiang Wu, and Guangyu Sun. PTQ4ViT: Post-training quantization for vision transformers with twin uniform quantization. In *Computer Vision – ECCV 2022*, volume 13672 of *Lecture Notes in Computer Science*, pages 191–207. Springer, 2022. doi: 10.1007/978-3-031-19775-8_12.

Tianchen Zhao, Xuefei Ning, Tongcheng Fang, Enshu Liu, Guyue Huang, Zinan Lin, Shengen Yan, Guohao Dai, and Yu Wang. Mixdq: Memory-efficient few-step text-to-image diffusion models with metric-decoupled mixed precision quantization. In *European Conference on Computer Vision*, pages 285–302. Springer, 2024.

Kuncai Zhong, Zexi Li, Haoran Jin, and Weikang Qian. Exploiting uniform spatial distribution to design efficient random number source for stochastic computing. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, ICCAD '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392174. doi: 10.1145/3508352.3549396. URL https://doi.org/10.1145/3508352.3549396.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024.

A Supplementary Material

A.1 Correlation and the C-BSG Construction

This appendix expands the brief reference to conditional bitstream generation (C-BSG) in Section 3.1: it gives the formal definition of stochastic cross-correlation, motivates why a non-zero correlation biases the SC estimator in (1), and presents the UGEMM/C-BSG construction that we use throughout this paper to drive the correlation toward zero by construction.

Correlation between the paired streams, which violates the independence assumption behind (1), is a major source of SC error [Alaghi and Hayes, 2013b, Wu et al., 2020]. Following Alaghi and Hayes [2013a] and Wu et al. [2020], let a, b, c, d count the cycles at which the bit pair (z_ℓ^A, z_ℓ^B) equals $(1,1)$, $(1,0)$, $(0,1)$, $(0,0)$ respectively (with $a+b+c+d=L$); the *stochastic cross-correlation* $\mathrm{SCC}(Z^A, Z^B) \in [-1,+1]$ is a normalized form of $ad-bc$ that vanishes exactly when the two streams are uncorrelated ($ad=bc$), with full piecewise definition

$$\mathrm{SCC}(Z^A, Z^B) \;=\; \frac{ad-bc}{D}, \qquad D \;=\; \begin{cases} L\,\min(a+b,\,a+c) - (a+b)(a+c), & \text{if } ad > bc, \\ (a+b)(a+c) - L\,\max(a-d,\,0), & \text{otherwise.} \end{cases} \tag{3}$$

The two-case denominator normalizes $ad-bc$ by its maximum attainable absolute value subject to the marginals $(a+b)$ and $(a+c)$, so that $\mathrm{SCC}=+1$ (-1) is reached exactly when the overlap of 1s between Z^A and Z^B is maximal (minimal) for those marginals.

Pseudo-random generators such as LFSRs converge at the slower $\mathcal{O}(L^{-1/2})$ rate [Alaghi et al., 2018], and conventional stream generation does not in general guarantee uncorrelated streams [Wu et al., 2020]. We instead use scrambled Sobol low-discrepancy sequences [Sobol', 1967, Joe and Kuo, 2008, Li et al., 2024]: deterministic sequences in $[0,1)^d$ constructed by XOR-ing bit-indexed direction numbers, with L^∞-discrepancy $\mathcal{O}(L^{-1}\log^d L)$ that is asymptotically tighter than the $\mathcal{O}(L^{-1/2})$ rate of i.i.d. samples [Niederreiter, 1992]. Even so, Sobol RNGs reduce $|\mathrm{SCC}|$ but do not eliminate it at finite L [Wu et al., 2020], and the residual non-zero correlation biases the estimator of (1).

We address this gap with the UGEMM [Wu et al., 2020, Wu and San Miguel, 2022] construction. For two bipolar streams z^A, z^B encoding $u^A, u^B \in [-1,1]$, the output bit value $z_{\mathrm{out}} = \mathrm{XNOR}(z^A, z^B)$ that decodes to $u^A u^B$ requires the output probability to satisfy [Wu et al., 2020]

$$\Pr[z_{\mathrm{out}}=1] \;=\; \Pr[z^A=1]\,\Pr[z^B=1] \;+\; \Pr[z^A=0]\,\Pr[z^B=0], \tag{4}$$

i.e., a function of the input *marginals*. A bitwise XNOR over two streams, however, fires whenever the two bits agree, so its output probability is the *joint* probability of agreement, which factors via the chain rule as [Wu et al., 2020]

$$\begin{aligned} \Pr[z_{\mathrm{out}}=1] \;&=\; \Pr[z^A=1,\, z^B=1] \;+\; \Pr[z^A=0,\, z^B=0] \\ &=\; \Pr[z^A=1]\,\Pr[z^B=1 \mid z^A=1] \;+\; \Pr[z^A=0]\,\Pr[z^B=0 \mid z^A=0], \end{aligned} \tag{5}$$

which matches (4) only when both conditional probabilities equal the corresponding marginal:

$$\Pr[z^B=1 \mid z^A=1] \;=\; \Pr[z^B=1], \qquad \Pr[z^B=0 \mid z^A=0] \;=\; \Pr[z^B=0]. \tag{6}$$

Conditional bitstream generation (C-BSG) [Wu et al., 2020, Wu and San Miguel, 2022] targets (6) directly (Figure 1(c)): rather than generating Z^A and Z^B from two independent RNGs, the bits of Z^B come from generators whose updates are gated by Z^A [Wu et al., 2020], so that the conditional distribution of z^B given z^A matches its marginal. This drives SCC toward zero [Wu and San Miguel, 2022], so the XNOR-based bipolar multiplier avoids correlation-induced bias, up to a residual at finite L.

A.2 Residual Compensation in the SC Datapath

The QwT recipe we adapt. QwT [Fu et al., 2025] attaches one lightweight residual *compensation* module to each transformer block of an already-quantized network. For block b with quantized output $y_b^Z(x) = l_b^Z(x)$ and full-precision teacher output $y_b(x) = l_b(x)$, the module

$$\widehat{y}_b(x) \;=\; y_b^Z(x) \;+\; W_b\,x \;+\; b_b \tag{7}$$

adds an affine correction trained to minimize the per-block residual $\sum_{x\in\mathcal{C}} \| y_b(x) - y_b^Z(x) - W_b x - b_b \|_2^2$ on a small calibration set $\mathcal{C}$ of $\sim$512 ImageNet training images. The minimizer is the ordinary least-squares solution $W_b = (Y_b - Y_b^Z)\, X_b^\top (X_b X_b^\top)^{-1}$, with the bias absorbed by augmenting X_b with a row of ones, so calibration is one closed-form solve per block and runs in minutes. By construction $W_b{=}0$ recovers the uncompensated network, and Fu et al. [2025] use it as a fallback: they keep a block's closed-form fit only when its coefficient of determination satisfies $R^2{>}0$ and set $W_b{=}0$ otherwise, because a few modules ($< 5\%$) hurt accuracy. Fu et al. [2025] report ImageNet gains of up to $+7.8$ pt at W4A4 with RepQ-ViT (ViT-B, $68.5 \rightarrow 76.3$) and of $+0.1$ to $+0.3$ pt at W6A6.

Our SC-side compensation recipe. We adopt the same per-block, residual structure as (7), with three modifications motivated by the SC residual being stochastic-but-deterministic (see the paragraph on deterministic SC noise below) rather than an integer rounding map. First, we use $|\mathcal{C}|{=}1024$ ImageNet train images and a ridge-regularized solve $W_b = (Y_b{-}Y_b^Z)\, X_b^\top (X_b X_b^\top + \lambda I)^{-1}$ with $\lambda{=}10^{-4}$, which we found necessary to keep W_b well-conditioned in low-L blocks where X_b^Z has reduced effective rank. Second, we fit and store W_b *per head* (16 heads $\times$ $d_{\text{head}}{=}64$) rather than as a single $d{\times}d$ matrix, which reduces parameter count and lets the compensator follow the head-aligned SC kernels of Section 4.2. Third, we gate admission of W_b into the inference path by a *disjoint-split cosine test*: we fit two independent solutions $W_b^{(A)}, W_b^{(B)}$ on disjoint 512-image calibration halves under one fixed Sobol configuration and admit $W_b = \frac{1}{2}(W_b^{(A)} + W_b^{(B)})$ only when $\cos\big(W_b^{(A)}, W_b^{(B)}\big) > \tau$ ($\tau{=}0.5$ for blocks $0{-}22$, $\tau{=}0.8$ for the final block). Failed gates leave the block uncompensated. The gate exists because the SC residual fluctuates at finite calibration size, and an admitted W_b that happens to fit calibration noise can degrade rather than improve held-out accuracy; the cross-split test is designed to reject such fits. Empirically the gate admits $22/24$ blocks at $\bar{L}{\geq}128$ on ViT-L/14 (Section 5.1, Table 3).

The compensator is itself an SC matmul. A subtle but essential property of our pipeline is that the residual matmul $W_b x$ in (7) is *not* an FP linear layer; it is a second SC dot-product evaluated on the C-BSG kernel of Section 4.2 at uniform stream length $L_{\text{comp}}{=}256$ ($p{=}8$, bipolar polarity). Calibration solves for the FP-valued W_b in the closed form above, then materializes W_b as the weight tensor of an SC linear layer that consumes the block input x via the same fused per-row quantization kernel and the same cumulative-indicator K-table cache that the block's main matmuls use. The compensator therefore adds no FP matmul: a compensated block costs its main-path SC matmuls plus one auxiliary per-head SC matmul at fixed $L_{\text{comp}}{=}256$. This property concerns the compensator only: the evaluated classification pipeline still keeps the 30 most sensitive (op, block) cells in FP, so it remains a hybrid of SC and PTQ (Section 6).

Deterministic SC noise lets W_b *absorb* a fixed realization. The C-BSG kernel of Section 4.2 is stochastic in name only: its randomness is a length-$L_{\max}$ Sobol low-discrepancy sequence [Sobol', 1967, Joe and Kuo, 2008] indexed by a configuration (dim, seed, polarity) that is fixed at model load and reused on every forward pass. Consequently the per-element SC error $\widehat{\langle w, x\rangle} - \langle w, x\rangle$ is a deterministic function of (w, x) for a given configuration—it is "shot noise" only across hypothetical Sobol seeds, not across forward passes. We exploit this by using the *same* Sobol configuration during compensation calibration and during inference, so the residual map that W_b is fit against is the same map that W_b encounters at test time. In effect, W_b does not need to "average out" SC noise across forward passes; it fits a fixed deterministic perturbation, and the closed-form fit captures the part of it that is approximately linear in x, up to the per-head structure, the ridge term, the finite calibration set, and the compensator's own SC error at $L_{\text{comp}}{=}256$. The same reasoning covers INT-side QwT [Fu et al., 2025]: deterministic rounding makes the INT residual a fixed function of the block input, as seed-fixed Sobol streams do for the SC residual, so a closed-form least-squares fit applies to either.

Scope: feasibility, not formal analysis. We frame this section strictly as a *feasibility* demonstration: a closed-form fit, a cross-split admission gate, and an SC-realized compensator together form a plausible recipe, and the empirical results of Section 5.1 (within 1 pt of FP at $\bar{L}{\geq}192$; at $\bar{L}{=}128$, within 3.2 pt with MP and 3.6 pt with uniform L) suggest the recipe is sound at the budgets we evaluate. We do *not* establish here (i) that the deterministic-Sobol fixed-point characterization of W_b is unique or stable across changes of Sobol configuration, (ii) conditions under which the cross-split

cosine gate is the optimal admission criterion versus, e.g., a residual-norm threshold, or (iii) the regime in which an SC-realized W_b at $L_{\text{comp}}<256$ remains accurate enough to keep the cumulative compensation profitable. Compensated results below $\bar{L}=128$ are not reported here; where and why the SC residual leaves the linear-compensation regime is left to future work.

A.3 Grouped Quantization Without Fallback

The classification experiments of Section 5.1 keep the 30 most sensitive (op, block) cells in FP. These cells compensate for one quantization scale per row, which compresses small values in wide inputs, most severely the 4096-wide post-GELU input of MLP_FC2. Splitting each row of the reduction dimension into groups of 128 elements, each with its own symmetric scale and SC partial sum, removes the need for them: every group reuses the same XNOR/popcount datapath, and its precision is still set by the stream length.

Table 7 runs every matmul of the 24 transformer blocks of DINOv2 ViT-L/14 in SC, with no FP or INT fallback and no compensation. The INT baseline is asymmetric round-to-nearest quantization with the same 128-element groups. All rows are scored on one fixed 10,000-image subset of ImageNet-1k val, on which FP reaches 86.10%, and MP thresholds are calibrated on 128 images outside that subset. These runs use sign–magnitude streams, which reach the resolution of a length-L bipolar stream in $L/2$ cycles [Wu and San Miguel, 2022]. At 6 and 6.58 bits, MP also keeps the 2% of each linear layer's input channels with the largest calibration activations at a longer stream length, charged to the same average budget.

Table 7: DINOv2 ViT-L/14 top-1 accuracy (%) with 128-element quantization groups and every transformer-block matmul in SC, on a fixed 10,000-image ImageNet-1k val subset. W/A as in Table 3.

W/A	INT	SC uniform	SC MP
8/8	85.96	85.57	–
7/7	85.90	83.74	84.91
$\log_2 96/\log_2 96$	–	82.75	83.78
6/6	85.69	77.89	81.77

With groups, uniform SC at 8/8 reaches 85.57% with every matmul in SC, 0.39 points below INT8 with the same groups.

A.4 Image Generation Samples

Figure 4 shows qualitative samples for the image-generation setting in Section 5.3.

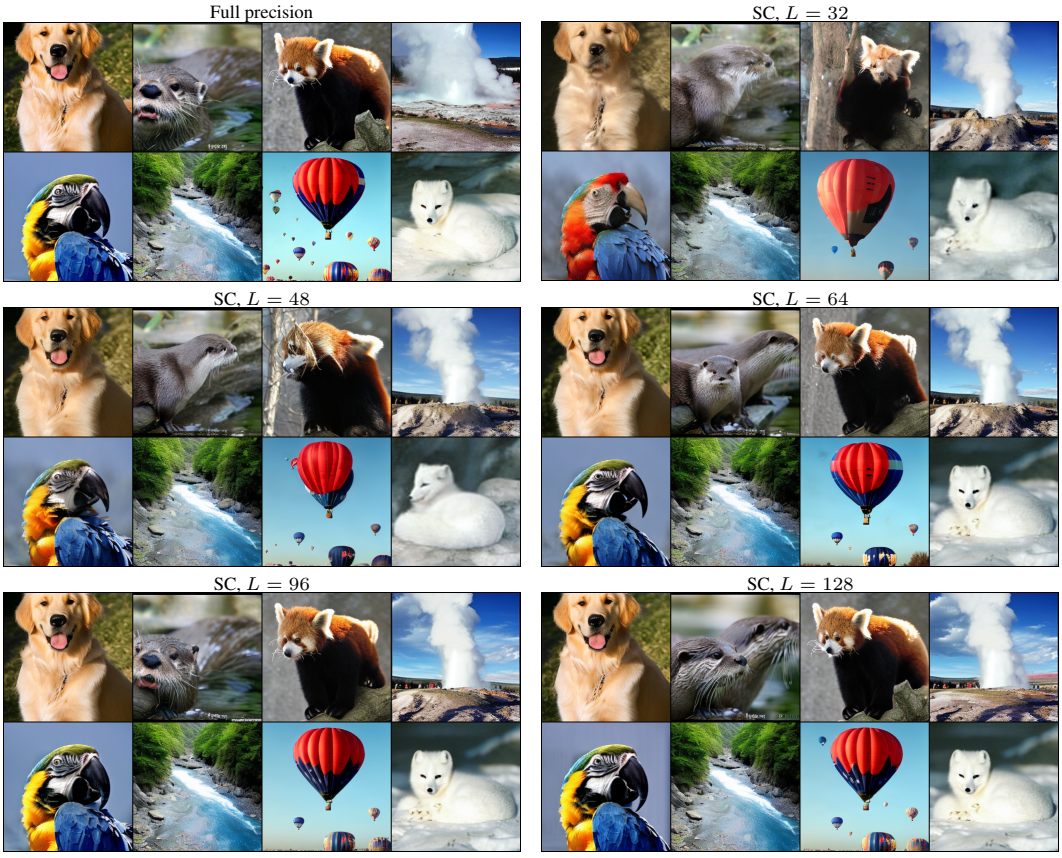


Figure 4: Sample images generated by full precision and SC with uniform stream lengths $L \in \{32, 48, 64, 96, 128\}$. The eight prompts are: "golden retriever", "otter", "lesser panda", "geyser", "macaw", "valley", "balloon", and "arctic fox".

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The contributions stated in the abstract and Section 1 are supported by the tables in Sections 5.1–5.4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6 contains a dedicated Limitations paragraph covering the empirical scope, the hybrid SC/PTQ pipeline, the single-model generation and planning evaluations, and that hardware benefits are estimated from gate-level synthesis and a cycle-accurate simulator rather than measured on silicon.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper presents no numbered theorems requiring proof; the closed-form compensation derivation in Appendix A.2 is constructive (standard ridge regression over seed-fixed Sobol streams).

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 4 specifies the kernel, sensitivity probe, and MP threshold-fitting algorithm; Sections 5.1–5.4 list the backbone, split, sampling or planning settings, and SC schedule of each experiment, and Section 4.3 and Appendix A.2 give the calibration-set sizes of the sensitivity probe and the compensation fit.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The SC GPU kernel library is publicly available at `https://github.com/CrucibleComputingGroup/scmp_kernels`, with installation instructions and kernel examples. We release only the GPU kernel library; the task evaluation scripts and the calibration and compensation code are not released.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Sections 5.1–5.4 list dataset splits, image sizes, sampler/CEM configurations, and SC schedules, with bit-budget conventions anchored on Table 3; Section 4.3 and Appendix A.2 give the calibration-set sizes of the sensitivity probe and the compensation fit.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

Justification: Table 5 reports uncertainty for KID, but the main task comparisons are not accompanied by repeated-run confidence intervals or significance tests. Section 6 describes the limited evaluation over seeds.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: We do not tabulate per-experiment compute; SC inference runs on a single Pro6000-class GPU and the full per-task sweep fits in a small 4-GPU pool over a few days.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses only public datasets and public pre-trained weights, involves no human subjects, and conforms to the Code of Ethics; dual-use considerations are discussed in Section 6.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 6 discusses the positive impact (cheaper inference and better-informed hardware investment) and the negative one (cheaper inference also lowers the cost of misusing existing vision models).

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper releases no new pre-trained models, datasets, or generative artifacts; misuse risk is bounded by the same considerations that already apply to the public models we evaluate on.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Models and baselines are credited by citation in the relevant sections, and the datasets are ImageNet-1k [Deng et al., 2009] and COCO [Lin et al., 2014]. ImageNet-1k is used under its non-commercial research terms of access, COCO annotations under CC BY 4.0 with images under the Flickr Terms of Use, DINOv2, QwT, and RepQ-ViT under Apache-2.0, EVA-01, the Dino-WM code, uSystolic-Sim, and Triton under MIT, and DiT under CC BY-NC 4.0. We redistribute no underlying weights or data.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The released SC GPU kernel library includes a README with installation instructions, API documentation, and usage examples. It is distributed under the GPL-3.0 license, included as the LICENSE file of the repository.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human-subjects research was conducted, so no IRB review was required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not part of the core methodology; any usage was confined to writing assistance, which the policy does not require declaring.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.