Accelerating Diffusion Transformer via Increment-Calibrated Caching with Channel-Aware Singular Value Decomposition

Supplementary Material

6. Detailed Illustration of CA-SVD

Algorithm 2 Channel-activation-aware Singular Value Decomposition

Input: Calibration set $\mathcal{D}$, DiT model $\epsilon_\theta(\cdot)$, number of timesteps T, noise scheduler β_t
Output: Scale matrix S_i and S_o of each linear layer l of $\epsilon_\theta(\cdot)$

for each linear layer l of $\epsilon_\theta(\cdot)$ **do**
  Initialize zero vector S_i and S_o
end for
for each $z_0 \in \mathcal{D}$ **do**
  $t \sim \mathcal{U}[1, T]$
  $z_t \sim \mathcal{N}(z_t, \alpha_t z_0, \sigma_t^2)$
  Compute $\epsilon_\theta(z_t, t))$ and
  for each linear layer l of $\epsilon_\theta(\cdot)$ **do**
    Accumulate the magnitude of inputs and outputs to S_i and S_o
  end for
end for
Turn S_i and S_o into diagonal matrices
return Outputs

As show in Algorithm 2, to enhance increment-calibrated caching with a extended version of ASVD [45], we obtain the distribution of activations of reverse process from a calibration set, which is composed of a small number of clean images. Just like the training process of DDPM [14], the reparameterization trick is utilized to model the distribution of noised data at any given timestep, which is streamed to DiT model. During the execution of DiT model, the accumulated activation magnitude is calculated, which will used to scale weight matrix before SVD as Eq. (5).

7. Implementation Details

We implement our methods based on the source codes and pre-trained models of DiT [30] and PixArt-α [4]. All the comparative experiments are conducted under the same conditions to ensure fairness, where the hardware platform and random seed are both unified. For the class-conditional image synthesis on ImageNet dataset, all the results are based on 8$\times$ NVIDIA RTX 4090D GPUs with a global batch size of 128 and data precision of TF32. For the results of text-to-image generation, we employ one 4090D GPU and set the batch size to 2. All the images are generated with the precision of FP16.

8. Theoretical analysis

To explain why the proposed method works, we analyze the induced error of both naive caching and increment-calibrated caching. To simplify the discussion, we ignore the error accumulation effect. Assuming the output of linear layer l at step s will be reused at step m, the error Δy of increment-calibrated caching can be formulated as,

$$\begin{aligned}\Delta y &= Wx_{m,l} - (Wx_{s,l} + W_r(x_{m,l} - x_{s,l})) \\ &= (W - W_r)(x_{m,l} - x_{s,l}) = \Delta W \Delta x\end{aligned} \tag{7}$$

where W is the original weight, W_r is the approximated weight of rank r, and $x_{t,l}$ denotes the input at step t. The upper bound of MSE can be represented as $\|\Delta W\|_2^2 \|\Delta x\|_2^2$. According to the property of SVD, a larger r always tends to decrease $\|\Delta W\|_2^2$. Note that the proposed method will be reduced to naive caching when r equals 0, therefore the proposed method can outperform naive caching by limiting the upper bound of MSE.

9. Visualization

To visually demonstrate the effectiveness of the proposed method, We provided generation images based on non-caching, naive caching and the proposed increment-calibrated caching with different sampler settings in Fig. 8, Fig. 9 and Fig. 10. We found naive caching tends to blur the details or Change the posture of an object, especially when a small step number is given. The proposed method can effectively correct these distortions and generate images of higher quality with marginal computation cost.

Figure 8. Visualization of the proposed method evaluated on DiT-XL/2 with 100-step DDIM. The cfg scale is set to 4.

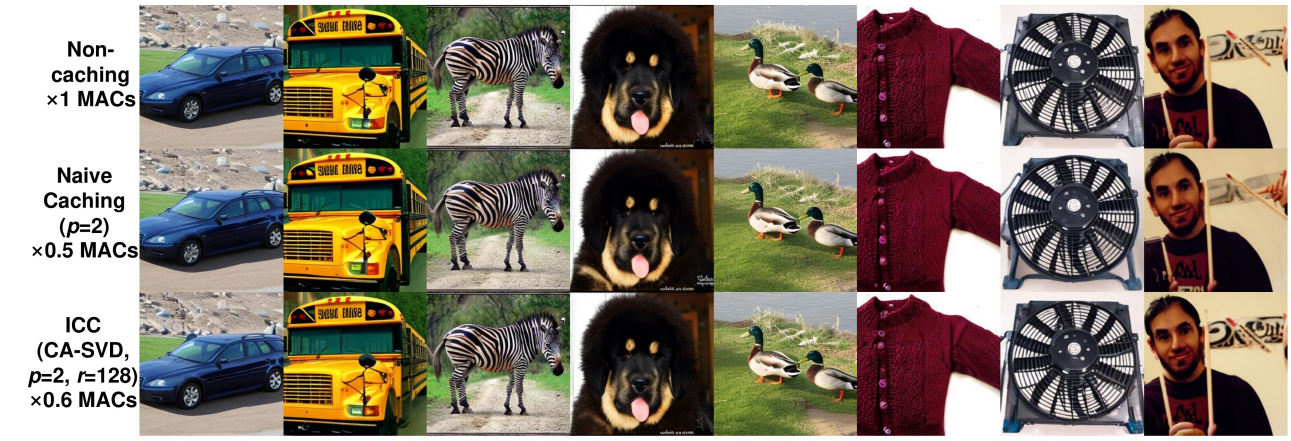

Figure 9. Visualization of the proposed method evaluated on DiT-XL/2 with 50-step DDIM. The cfg scale is set to 4.

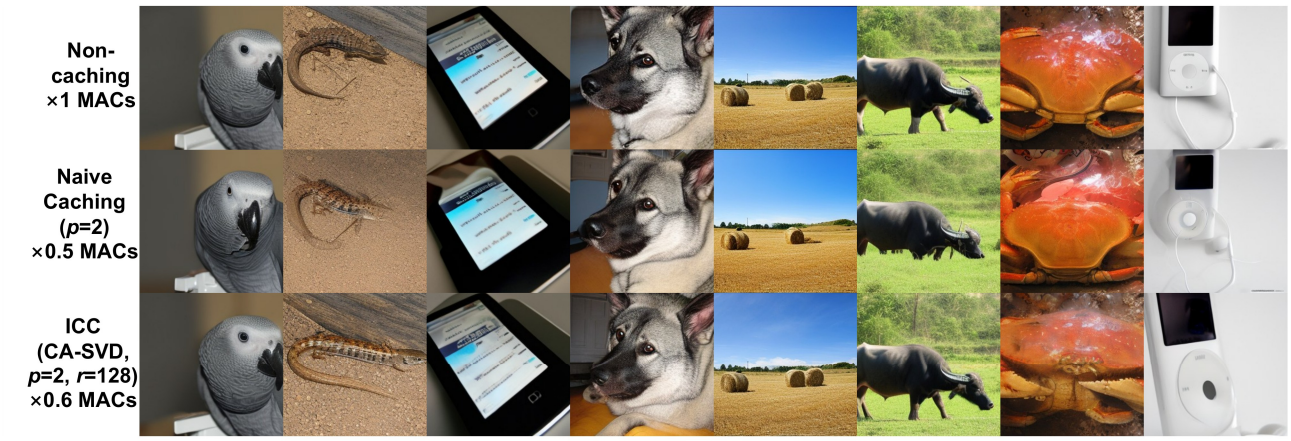

Figure 10. Visualization of the proposed method evaluated on DiT-XL/2 with 20-step DDIM. The cfg scale is set to 4.