BiGain: Unified Token Compression for Joint Generation and Classification Supplementary Material

Jiacheng Liu[1,*], Shengkun Tang[1,*], Jiacheng Cui[1], Dongkuan Xu[2], Zhiqiang Shen[1†]
[1]VILA Lab, MBZUAI [2]North Carolina State University
Code: https://github.com/Greenoso/BiGain

Supplementary Material

Contents

†Corresponding author.
*Equal contribution.

A. Frequency-Aware Token Reduction and Diffusion Classification

In this section we provide a simple stylized model showing how frequency-aware token reduction can tighten the diffusion-classifier decision, viewed through a margin–variance trade-off on the paired Monte Carlo estimator introduced in Sec. 3.1.1 of the main paper.

A.1. Diffusion Classifier and Paired Difference

Recall the diffusion-classifier score [3]:

$$S(x,c) \;=\; \mathbb{E}_{t,\epsilon}\,\|\epsilon-\epsilon_\theta(x_t,c,t)\|_2^2\,, \qquad x_t=\sqrt{\bar\alpha_t}\,x+\sqrt{1-\bar\alpha_t}\,\epsilon. \tag{S.1}$$

The prediction is $\hat c(x)=\arg\min_{c\in\mathcal{C}} S(x,c)$, implemented using *paired sampling*: for a shared Monte Carlo set $\mathcal{S}_{\mathrm{MC}}=\{(t_s,\epsilon_s)\}_{s=1}^{S_{\mathrm{MC}}}$ (reused for all classes as in Sec. 3.1.1) we approximate $S(x,c)$ by

$$\hat S_{S_{\mathrm{MC}}}(x,c)=\frac{1}{S_{\mathrm{MC}}}\sum_{s=1}^{S_{\mathrm{MC}}}\ell(x,c;t_s,\epsilon_s), \qquad \ell(x,c;t,\epsilon)=\|\epsilon-\epsilon_\theta(x_t,c,t)\|_2^2\,. \tag{S.2}$$

Fix the true class $c^\star$ and a distractor $\tilde c$. For one paired draw $(t,\epsilon)\sim p(t)p(\epsilon)$, we define

$$D(t,\epsilon)=\ell(x,\tilde c;t,\epsilon)-\ell(x,c^\star;t,\epsilon). \tag{S.3}$$

Let

$$\mu=\mathbb{E}[D],\sigma^2=\mathrm{Var}[D]. \tag{S.4}$$

With S_{MC} shared samples we obtain the paired estimator

$$\widehat\Delta_{S_{\mathrm{MC}}}=\frac{1}{S_{\mathrm{MC}}}\sum_{s=1}^{S_{\mathrm{MC}}}D(t_s,\epsilon_s), \qquad \mathbb{E}[\widehat\Delta_{S_{\mathrm{MC}}}]=\mu,\mathrm{Var}(\widehat\Delta_{S_{\mathrm{MC}}})=\frac{\sigma^2}{S_{\mathrm{MC}}}. \tag{S.5}$$

For a consistent classifier we have $\mu>0$, and misclassification against $\tilde c$ corresponds to the tail event $\widehat\Delta_{S_{\mathrm{MC}}}\le 0$.

By Cantelli's inequality, for any random variable Z with mean m and variance v, $\Pr(Z - m \le -a) \le v/(v + a^2)$ for $a > 0$. Applying this to $Z = \widehat{\Delta}_{S_{\mathrm{MC}}}$, $m = \mu$, $a = \mu$ yields

$$\Pr\big(\widehat{\Delta}_{S_{\mathrm{MC}}} \le 0\big) \;\le\; \frac{\sigma^2/S_{\mathrm{MC}}}{\mu^2 + \sigma^2/S_{\mathrm{MC}}} = f(r), r = \frac{\sigma}{\mu}, \tag{S.6}$$

where $f(r)$ is strictly increasing in r. Thus, tightening the Cantelli bound is equivalent to decreasing the ratio r.

A.2. Bandwise Decomposition

We now introduce a simple frequency-domain model of the paired difference. Let $\{\phi_k\}$ be an orthonormal 2-D DCT/Fourier basis over the spatial token grid. For each (t, ϵ) and class c, we expand the error as:

$$\ell(x, c; t, \epsilon) = \sum_k \omega_k(t) \left|\widehat{\epsilon}(k) - \widehat{\epsilon_\theta}(k; t, c)\right|^2, \tag{S.7}$$

where $\widehat{\cdot}(k)$ denotes the coefficient in band k and $\omega_k(t) \ge 0$ is a per-band reliability weight (e.g., arising from the ELBO weighting over t).

We define the bandwise paired difference as:

$$\Delta_k(t, \epsilon) := \left|\widehat{\epsilon}(k) - \widehat{\epsilon_\theta}(k; t, \tilde{c})\right|^2 - \left|\widehat{\epsilon}(k) - \widehat{\epsilon_\theta}(k; t, c^\star)\right|^2, \tag{S.8}$$

so that

$$D(t, \epsilon) = \sum_k \omega_k(t)\, \Delta_k(t, \epsilon). \tag{S.9}$$

Let

$$\begin{aligned} \mu_k &:= \mathbb{E}_{t,\epsilon}[\Delta_k(t, \epsilon)], \\ \sigma_k^2 &:= \mathrm{Var}_{t,\epsilon}[\Delta_k(t, \epsilon)], \\ w_k &:= \mathbb{E}_t[\omega_k(t)]. \end{aligned} \tag{S.10}$$

Assuming that cross-band covariances are weak,

$$\mathrm{Cov}\big(\Delta_i, \Delta_j\big) \approx 0, \qquad i \ne j, \tag{S.11}$$

we obtain the approximation

$$\begin{aligned} \mu &\approx \sum_k w_k \mu_k, \\ \sigma^2 &\approx \sum_k w_k^2 \sigma_k^2. \end{aligned} \tag{S.12}$$

Intuitively, discriminative classifiers benefit from high-frequency components (edges and textures) to refine the class margin μ. Coarse structures (low frequencies) provide stable semantic cues, while fine structures (high frequencies) often distinguish specific classes. Consequently, although high-frequency bands may exhibit larger variance under stochastic sampling, they can still carry strong class-specific information for the correct class. A reduction strategy that indiscriminately suppresses these frequencies, effectively acting as a low-pass filter with $H_P(k) \approx 0$ for high k, risks inducing a large margin loss $\Delta\mu$ that can outweigh any variance reduction $\Delta\sigma^2$.

A.3. Token Reduction as a Spectral Operator

We consider an attention block operating on a window of tokens, with pre-reduction features $z_i = s_i + n_i$, where s_i denotes structured signal and n_i zero-mean perturbations. A shape-preserving reduction operator P maps the window to a reduced representation (e.g., via merging or downsampling). Under a local linearization of the block, and for operators such as IE-KVD that are explicitly linear in each neighborhood, we approximate its effect via a *windowed frequency response* $H_P(k)$:

$$\begin{aligned} \mu' &\approx \sum_k w_k H_P(k)\, \mu_k, \\ \sigma'^2 &\approx \sum_k w_k^2 H_P(k)^2\, \sigma_k^2. \end{aligned} \tag{S.13}$$

This spectral model provides a useful approximation of how token-reduction operators reshape the band-weighted paired statistic. For strictly linear operators such as IE-KVD the frequency response $H_P(k)$ is exact, while for merging-based operators (e.g., L-GTM, ABM), it serves as a local linearization that captures their dominant low-pass-like behaviour.

In particular, any operator P whose effective response $H_P(k)$ (exact for IE-KVD, approximate for merging) acts as a *bandwise shrinkage rule*, with $H_P(k) \approx 1$ on margin-rich bands and $H_P(k) < 1$ on variance-heavy bands, will tend to reduce the ratio $r = \sigma/\mu$ in equation S.6. Both of our proposed modules, Laplacian-gated token merging (L-GTM) and Interpolate–Extrapolate KV-Downsampling (IE-KVD), are designed to approximate this behavior: they selectively smooth or merge tokens that are redundant while retaining tokens that appear informative, producing a frequency-selective shrinkage profile.

For IE-KVD, which is a linear filtered downsampling operator, the spectral interpretation is exact, for merging-based operators, it should be viewed as an approximation under local linearization. We emphasize that this analysis is a stylized model: it relies on linearization and approximate bandwise decorrelation, and is intended to clarify the design principle behind our frequency-aware token reduction rather than to provide a formal guarantee.

A.4. A Margin–variance Improvement Criterion

We define the changes in mean margin and variance as

$$\begin{aligned} \Delta\mu &:= \mu - \mu', \\ \Delta\sigma^2 &:= \sigma^2 - \sigma'^2. \end{aligned} \tag{S.14}$$

We are interested in when the ratio $r' = \sigma'/\mu'$ is smaller than $r = \sigma/\mu$, since by equation S.6 this implies a tighter Cantelli bound.

Theorem 1 (Spectral margin–variance improvement)**.** *Assume $\mu > 0$, $\mu' > 0$, $\sigma^2 > 0$, and that μ' and σ'^2 are defined as above. Then the post-reduction ratio r' is smaller than the original ratio r, $r' < r$, if and only if*

$$\Delta\sigma^2 \;>\; 2\,\frac{\sigma^2}{\mu}\,\Delta\mu \;-\; \frac{\sigma^2}{\mu^2}\,(\Delta\mu)^2. \tag{S.15}$$

Moreover, when $|\Delta\mu| \ll \mu$, the first-order sufficient condition

$$\Delta\sigma^2 \;>\; 2\,\frac{\sigma^2}{\mu}\,\Delta\mu \tag{S.16}$$

guarantees $r' < r$ and hence a strictly improved Cantelli bound.

Proof. Since $\mu > 0$ and $\mu' > 0$ we have $r, r' \geq 0$, so $r' < r$ is equivalent to $r'^2 < r^2$. The condition $r'^2 < r^2$ is equivalent to $\frac{\sigma^2 - \Delta\sigma^2}{(\mu - \Delta\mu)^2} < \frac{\sigma^2}{\mu^2}$. Multiplying both sides by the positive denominators and rearranging gives $\Delta\sigma^2\mu^2 - 2\sigma^2\mu\,\Delta\mu + \sigma^2(\Delta\mu)^2 > 0$, which after dividing by μ^2 yields equation S.15. Expanding the right-hand side of equation S.15 to first order in $\Delta\mu/\mu$ gives the sufficient condition equation S.16. □

A.5. Interpretation and Scope

Equation S.15, together with the bandwise decompositions of $\Delta\mu$ and $\Delta\sigma^2$, provides a spectral lens for comparing reduction strategies. The term $\Delta\sigma^2 = \sum_k w_k^2(1 - H_P(k)^2)\,\sigma_k^2$ captures the *variance savings*, while $\Delta\mu = \sum_k w_k(1 - H_P(k))\,\mu_k$ quantifies the corresponding *margin cost*. Achieving a tighter decision bound ($r' < r$) requires maximizing the former while keeping the latter small.

Standard token-merging approaches (e.g., ToMe [2]) can be interpreted, in our stylized spectral model, as performing spatial averaging: they tend to smooth fine-scale structure and thus resemble a low-pass filter with an effective response $H_P(k) \ll 1$ for high-frequency bands k. Because edges, textures, and other fine-scale structures often yield large μ_k (i.e., high-frequency bands are *margin-rich* for recognition), indiscriminate merging induces a substantial margin loss $\Delta\mu$. If this loss exceeds the allowable threshold in Theorem 1, the classifier's performance degrades despite any reduction in variance.

In contrast, the proposed **BiGain** is designed to favor this trade-off. Laplacian gating identifies regions where high-frequency content is relatively small, which in our spectral model corresponds to bands with small μ_k. In regions containing significant high-frequency structure (large μ_k), **BiGain** suppresses merging, corresponding to an effective response $H_P(k) \approx 1$ and hence $\Delta\mu \approx 0$ on margin-critical bands in our spectral model. This frequency-aware selection biases the shrinkage profile towards satisfying equation S.15, which is consistent with our observation that BiGain preserves discriminative utility far better than spectrally agnostic merging schemes.

B. Implementation Details

B.1. Datasets and Evaluation Protocols

B.1.1. Dataset Details

We evaluate on four widely-used benchmarks, summarized in Table 1. Following [3], ImageNet-1K is sub-sampled to 2,000 images for classification to reduce computational cost, while the full validation set is retained for generation experiments.

B.1.2. Diffusion Classifier Protocol

Diffusion-classifier. We follow the *Diffusion Classifier* framework [3], which scores a candidate conditioning c by the expected noise-prediction error $\mathbb{E}_{t,\epsilon}\left[\|\epsilon - \epsilon_\theta(x_t, c, t)\|_2^2\right]$ and selects the minimizer. This method is *training-free*, requiring no calibration or finetuning, and enables zero-shot classification directly from pretrained diffusion models. To enable evaluation on large label spaces, we use adaptive evaluation with staged pruning (detailed in Algorithm C.1). We adjust only `TrialList` and `KeepList` based on the size of the candidate set.

For completeness, we also evaluated velocity-prediction flow-matching models (FLUX [1]). Using the `FlowMatchEulerDiscreteScheduler` to construct affine mappings for recovering $\hat{\epsilon}_\theta$ and $\hat{x}_0$ within DDIM, the released FLUX.1-dev checkpoint performed only marginally better than random guessing under the diffusion-classifier protocol. To avoid adapter-specific confounds and ensure a fair comparison, we restrict all evaluations to standard noise-prediction models.

B.2. Multi-Label Classification Metric

We evaluate performance using mean Average Precision (mAP) for multi-label image classification, computed from a ranked list of labels for each image.

Given an image x with ground-truth label set $\mathcal{Y} \subset \mathcal{C}$, the diffusion classifier assigns each label $c \in \mathcal{C}$ an estimated classification error

$$\hat{L}(x, c) = \frac{1}{S}\sum_{s=1}^{S} \ell(x, c; t_s, \epsilon_s), \tag{S.17}$$

where $\ell(x, c; t_s, \epsilon_s)$ and the shared Monte Carlo set $\{(t_s, \epsilon_s)\}_{s=1}^S$ are defined in Sec. 3.1.1 of the main paper. Lower values of $\hat{L}(x, c)$ indicate higher confidence that label c is present.

All labels are ranked in ascending order of $\hat{L}(x, c)$.

Average Precision per image. Let $\{c_1, c_2, \dots\}$ denote the ranked label list. The Average Precision for image x

Table 1. Dataset statistics with official splits used in our experiments.

Dataset	Classes	Split	# Images (Cls.)	# Images (Gen.)
ImageNet-100 [12]	100	Val.	5,000	5,000
ImageNet-1K [9]	1,000	Val.	2,000	50,000
Oxford-IIIT Pets [5]	37	Test	3,669	3,669
COCO-2017 [4]	80	Val.	5,000	5,000

Table 2. Adaptive diffusion-classifier parameters per dataset. N_{stages} is the number of pruning stages; `TrialList` is the cumulative number of Monte Carlo trials per candidate by stage; `KeepList` is the number of candidates retained after each stage.

Dataset	N_{stages}	`TrialList`	`KeepList`
ImageNet-100	2	[5, 20]	[5, 1]
COCO-2017	2	[5, 20]	[5, 1]
Oxford-IIIT Pets	2	[5, 20]	[5, 1]
ImageNet-1K	3	[5, 20, 100]	[50, 10, 1]

is defined as

$$\mathrm{AP}(x) = \frac{1}{|\mathcal{Y}|} \sum_{k:\, c_k \in \mathcal{Y}} \frac{|\mathcal{Y} \cap \{c_1, \ldots, c_k\}|}{k}, \tag{S.18}$$

where the sum runs over ranks at which a ground-truth label appears.

Mean Average Precision. The final mAP is obtained by averaging $\mathrm{AP}(x)$ over all test images.

B.3. Model Configurations

B.3.1. Prompt Templates

For the classification task, following [3], we use ``a photo of a {class}'' for ImageNet and COCO datasets, and ``a photo of a {class}, a type of pet'' for Oxford-IIIT Pets.

For generation, we use the same templates except for COCO-2017, where we use the official validation captions.

B.3.2. Generation Setup

We standardize generation across both backbones. For Stable Diffusion 2.0 (UNet) [8], we use the EulerDiscreteScheduler with a scaled-linear beta schedule (beta_start 0.00085, beta_end 0.012, 1,000 training steps, epsilon prediction). For DiT-XL/2-512 [7], we use the DDIMScheduler with a linear beta schedule (beta_start 0.0001, beta_end 0.02, 1,000 training steps, epsilon prediction). In both cases, we sample for 50 steps at 512×512 resolution. We apply classifier-free guidance with a scale of 7.5 for Stable Diffusion 2.0 and 4.0 for DiT-XL/2-512. Unless otherwise stated, all experiments are conducted in FP16 precision. For evaluation, FID scores are computed using the `pytorch-fid` implementation [10].

B.4. Token Compression

B.4.1. Compression Settings

Guided by the ablation in Table 6, we apply compression exclusively to self-attention (SA) and leave cross-attention (CA) and MLP blocks intact to preserve prompt adherence. For merging-based operators, merging is performed inside each SA block and an explicit unmerge restores the original sequence length before the residual addition, ensuring dense outputs for downstream modules. For KV-downsampling operators, only keys and values are subsampled while queries remain full-length, removing the need for unmerge.

Stable Diffusion 2.0 (U-Net). We insert compression exclusively at the highest-resolution encoder layers, where the spatial token count, and thus attention cost is maximal. This targets the primary bottleneck while maintaining quality.

Diffusion Transformer (DiT-XL/2). To assess generality beyond U-Net architectures, we port the same operators to DiT-XL/2. Specifically, token compression is applied within the first 12 transformer blocks, comparing early (blocks 1–6) versus mid-early (blocks 7–12) reduction, while leaving later blocks, where class conditioning and fine structural details consolidate unchanged.

B.4.2. Baseline Implementation

For all token compression baselines, we use the official implementations and default parameters released by the authors, and run them under a common experimental protocol (see Sec. B.1.2 and Sec. B.3.2) to ensure fair comparison and avoid unintentional re-tuning. The only modification we introduce is to vary the token reduction ratio, so that each method can be fairly evaluated under different levels of compression.

B.5. Efficiency Evaluation

To measure the acceleration effect of our token reduction methods, we evaluate on the official Stable Dif-

fusion 2.0 implementation released by Stability AI [8]. All experiments are conducted on a single NVIDIA RTX 4090 GPU in half-precision (`float16`). We report wall-clock sampling time per image batch excluding the VAE encoding/decoding overhead, since our methods target the denoising backbone rather than the autoencoder. FLOPs are measured using `FlopCounterMode` from `torch.utils.flop_counter` [6]. The corresponding runtime and efficiency results are summarized in Table 3, which demonstrates the advantage of our method.

C. Algorithm

C.1. Adaptive Diffusion Classifier

Naïve diffusion classification requires evaluating all candidate classes, and thus its cost grows linearly with the number of classes. To mitigate this, we adopt the adaptive evaluation strategy introduced in the diffusion-classifier framework [3]. At each stage, we allocate a fixed budget of trials across the remaining classes, discard unlikely candidates based on their average error, and retain only the most promising ones. This progressive pruning concentrates computation on high-confidence classes, enabling more fine-grained Monte Carlo error estimation. The procedure is summarized in Algorithm 1.

Algorithm 1 Diffusion Classifier (Adaptive) [3]

Require: test image $\mathbf{x}$, conditioning inputs $\mathcal{C} = \{\mathbf{c}_i\}_{i=1}^{n}$ (e.g., text embeddings or class indices), number of stages N_{stages}, list `KeepList` of number of $\mathbf{c}_i$ to keep after each stage, list `TrialList` of number of trials done by each stage

Initialize $\texttt{Errors}[\mathbf{c}_i] = \text{list}()$ for each $\mathbf{c}_i$
Initialize $\texttt{PrevTrials} = 0$ ▷ How many times we've tried each remaining element of $\mathcal{C}$ so far
for stage $i = 1, \ldots, N_{\text{stages}}$ **do**
  for trial $j = 1, \ldots, \texttt{TrialList}[i] - \texttt{PrevTrials}$ **do**
    Sample $t \sim [1, 1000]$
    Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$
    $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x} + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}$
    for conditioning $\mathbf{c}_k \in \mathcal{C}$ **do**
      $\texttt{Errors}[\mathbf{c}_k]$.append$(\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}_k, t)\|^2)$
    end for
  end for
  $\mathcal{C} \leftarrow \underset{\mathcal{S}\subset\mathcal{C};\ |\mathcal{S}|=\texttt{KeepList}[i]}{\arg\min} \sum_{\mathbf{c}_k\in\mathcal{S}} \text{mean}(\texttt{Errors}[\mathbf{c}_k])$ ▷ Keep top $\texttt{KeepList}[i]$ conditionings
  $\texttt{PrevTrials} = \texttt{TrialList}[i]$
end for
return $\underset{\mathbf{c}_i\in\mathcal{C}}{\arg\min}\text{mean}(\texttt{Errors}[\mathbf{c}_i])$

C.2. Frequency-Aware Token Scoring

Spectral structure of latent features is important for both discriminative and generative ability. High-frequency tokens encode the information of edges, textures, and small objects, especially at the late denoise stage, which are indispensable for recognition. However, high-frequency tokens can also amplify the variance in the diffusion classifier since predictions are aggregated over Monte Carlo draws of timesteps and noise; excess high-frequency tokens inflate the per-timestep estimation variance. Moreover, different timesteps emphasize different bands, early denoising focuses on low frequencies (global structure) while later steps emphasize high frequencies (fine detail). Therefore, the compression schedule should be *spectrally balanced and temporally consistent* to avoid injecting avoidable variance across timesteps. The necessity of preserving a balanced spectrum is confirmed empirically in Table 4, where discarding either high- or low-frequency tokens severely harms classification.

Our **BiGain$_{\text{TM}}$** design follows this principle. Since token merging resembles a local low-pass filter, we encourage merging only in small, spectrally smooth neighborhoods, where low-frequency information can be safely aggregated, while protecting detail-rich tokens that anchor class-critical microstructures. This balanced policy removes redundancy without sacrificing classification accuracy or generation fidelity. Practically, we introduce a set of fast, training-free scoring heuristics to decide which tokens to *preserve* (high detail) and which to *merge* (smooth/redundant), and we apply them consistently across timesteps so that each per-timestep classifier score remains reliable and contributes coherently to the Monte Carlo ensemble.

Notation. Let $\boldsymbol{X} \in \mathbb{R}^{H\times W\times C}$ denote the hidden feature tensor (height H, width W, channels C). For spatial index (i, j), the token (channel vector) is $\boldsymbol{x}_{i,j} := \boldsymbol{X}_{i,j,:} \in \mathbb{R}^C$. The global mean token is $\boldsymbol{\mu} := \frac{1}{HW}\sum_{p=1}^{H}\sum_{q=1}^{W}\boldsymbol{x}_{p,q}$. For a 3×3 spatial kernel $\boldsymbol{L}$, $(\boldsymbol{X} * \boldsymbol{L})_{i,j,c}$ denotes 2-D convolution at (i, j) on channel c. Let $\mathbb{N}_4(i, j)$ be the (in-bounds) 4-neighborhood of (i, j) (up/down/left/right). The DFT of $\boldsymbol{x}_{i,j}$ at channel-frequency bin k is $\hat{\boldsymbol{x}}_{i,j,k} := \sum_{c=1}^{C}(\boldsymbol{x}_{i,j})_c\, e^{-2\pi\mathrm{i}\,(c-1)(k-1)/C}$ for $k \in \{1, \ldots, C\}$. We write $||\cdot||_p$ for the vector ℓ_p norm, $||\cdot|| \equiv ||\cdot||_2$, and $\langle \boldsymbol{a}, \boldsymbol{b}\rangle$ for the Euclidean inner product. We compute a scalar score $F_{i,j} \in \mathbb{R}$ per token, where larger values indicate detail-rich tokens and smaller values indicate smooth/redundant tokens. We list all functions of different metrics in Table 5.

For all heuristics except cosine-based ones, larger $F_{i,j}$ indicates stronger local variation and thus high-frequency detail. In contrast, for cosine similarity scores, *smaller* values correspond to tokens that deviate more from their neighbors or the global mean, and are therefore detail-rich.

Table 3. **Stable Diffusion 2.0 efficiency (batch size 4).** Wall-clock sampling time per *batch* (seconds) excluding VAE encode/decode. All rows use merge ratio $r = 0.7$.

Method	Time ↓ (s / batch)	Acceleration ↑ (%)	FLOPs ↓ (G)
Baseline (No Accel.)	11.98	–	804.26
SiTo [13]	8.71	27.30	748.49
ToMe [2]	7.37	38.48	704.87
Laplacian Gated Merge (Ours)	7.37	38.48	704.99
Cached Assignment Merge (Ours)	7.29	39.15	698.88
Adaptive Block Merging (Ours)	7.27	39.32	695.08

Table 4. **Classification results on frequency-based KV selection on ImageNet-100.** We compare the standard TODO strategy with frequency-aware variants that select tokens with the highest or lowest Laplacian scores globally. Retaining only high- or low-frequency tokens severely degrades classification performance, highlighting the need to preserve a balanced spectrum.

Downsampling strategy	Acc@1 ↑	KV token sparsity
Todo (Nearest-Neighbor) [11]	72.30	75%
Low-frequency tokens (lowest-laplacian)	45.58	75%
High-frequency tokens (Highest-laplacian)	26.56	75%

C.3. BiGain$_{\text{TM}}$

Algorithm 2 presents our frequency-aware token merging method. The core innovation lies in using spectral information to guide merge decisions, ensuring that token reduction preserves both generative fidelity and discriminative utility. The algorithm first applies a frequency scorer $\mathcal{F}$ (default: Laplacian filtering C.2) to identify local frequency content in the spatial feature map. Tokens with low frequency scores indicate smooth, homogeneous regions amenable to merging, while high scores correspond to edges, textures, and fine details critical for classification.

The destination selection step partitions the spatial layout into regular grids and identifies the lowest-frequency token within each grid as a merge destination. This strategy ensures spatial coverage while directing merging toward spectrally smooth regions. The remaining tokens form a source set, which is then assigned to destinations via bipartite matching based on cosine similarity. By selecting the top-r fraction of most similar pairs, the method preserves semantic coherence while respecting the frequency-based partitioning. After merging and processing through attention layers, an unmerge operation restores the original sequence length for architectural compatibility.

Algorithm 3 presents Adaptive Block Merge (ABM), a computationally efficient variant designed for high-resolution stages where token count is maximal. Rather than per-token assignment, ABM operates at block granularity. After computing frequency scores, the feature map is partitioned into blocks, and blocks are ranked by their frequency content. The lowest-scoring fraction r of blocks are identified as smooth regions and merged via averaging, while high-frequency blocks remain intact. This block-level decision reduces computational complexity of bipartite matching, providing speedup with little accuracy degradation as demonstrated in our Table 7.

Algorithm 2 BiGain$_{\text{TM}}$: Frequency-Aware Token Merging

Require: Tokens $\boldsymbol{X} \in \mathbb{R}^{N \times d}$, merge ratio r, grid size s, frequency scorer $\mathcal{F}$

function BIGAINMERGE($\boldsymbol{X}$, r, s, $\mathcal{F}$)
  $\boldsymbol{f} \leftarrow \mathcal{F}(\boldsymbol{X})$ ▷ Score tokens by frequency content
  $\mathbb{D} \leftarrow$ SelectDestinations($\boldsymbol{f}, s$) ▷ Lowest frequency per grid
  $\mathbb{S} \leftarrow \{1, \ldots, N\} \setminus \mathbb{D}$ ▷ Remaining tokens as sources
  $\mathcal{M} \leftarrow$ BipartiteMatch($\boldsymbol{X}_{\mathbb{S}}, \boldsymbol{X}_{\mathbb{D}}, r$) ▷ Similarity-based assignment
  $\boldsymbol{X}^{\text{merged}} \leftarrow$ Merge($\boldsymbol{X}, \mathcal{M}$) ▷ Combine assigned tokens
  $\boldsymbol{Z} \leftarrow$ Process($\boldsymbol{X}^{\text{merged}}$) ▷ Apply attention
  return Unmerge($\boldsymbol{Z}, \mathcal{M}$) ▷ Restore dimensions
end function

C.4. BiGain$_{\text{TD}}$

Algorithm 4 presents our Interpolate-Extrapolate KV-Downsampling method, which reduces attention complexity by downsampling keys and values while preserving queries at full resolution. This asymmetric approach maintains the model's ability to attend precisely to all spatial positions while reducing memory and computation. The key innovation is the controllable linear combination of nearest-neighbor and average pooling, allowing fine-grained control over the frequency-preservation trade-off.

Here we use the same interpolate-extrapolate operator

Table 5. Formulas of different metrics.

Metric Name	Formula
Global mean deviation	$F_{i,j} = \|\boldsymbol{x}_{i,j} - \boldsymbol{\mu}\|$
ℓ_1 norm	$F_{i,j} = \|\boldsymbol{x}_{i,j}\|_1$
ℓ_2 norm	$F_{i,j} = \|\boldsymbol{x}_{i,j}\|$
Channel variance	$F_{i,j} = \frac{1}{C}\sum_{c=1}^{C}\left((\boldsymbol{x}_{i,j})_c - \frac{1}{C}\sum_{c'=1}^{C}(\boldsymbol{x}_{i,j})_{c'}\right)^2$
Laplacian (ℓ_1)	$F_{i,j} = \frac{1}{C}\sum_{c=1}^{C}\|(\boldsymbol{X} * \boldsymbol{L})_{i,j,c}\|, \quad \boldsymbol{L} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$
Laplacian (ℓ_2)	$F_{i,j} = \sqrt{\frac{1}{C}\sum_{c=1}^{C}\left((\boldsymbol{X} * \boldsymbol{L})_{i,j,c}\right)^2}$
DFT spectral centroid	$F_{i,j} = \frac{\sum_{k=1}^{C} k\,\|\hat{\boldsymbol{x}}_{i,j,k}\|}{\sum_{k=1}^{C}\|\hat{\boldsymbol{x}}_{i,j,k}\|}$
DFT total amplitude	$F_{i,j} = \sum_{k=1}^{C}\|\hat{\boldsymbol{x}}_{i,j,k}\|$
Cosine similarity to neighbors	$F_{i,j} = \frac{1}{\|\mathbb{N}_4(i,j)\|}\sum_{(p,q)\in\mathbb{N}_4(i,j)}\frac{\langle \boldsymbol{x}_{i,j}, \boldsymbol{x}_{p,q}\rangle}{\|\boldsymbol{x}_{i,j}\|\,\|\boldsymbol{x}_{p,q}\|}$
Cosine similarity to global mean	$F_{i,j} = \frac{\langle \boldsymbol{x}_{i,j}, \boldsymbol{\mu}\rangle}{\|\boldsymbol{x}_{i,j}\|\,\|\boldsymbol{\mu}\|}$

Algorithm 3 Adaptive Block Merge (ABM): Fast **BiGain$_{\text{TM}}$** Variant

Require: Tokens $\boldsymbol{X} \in \mathbb{R}^{N\times d}$, block size b, merge ratio $r \in [0,1]$, scorer $\mathcal{F}$

function AdaptiveBlockMerge($\boldsymbol{X}$, b, r, $\mathcal{F}$)
  $\boldsymbol{f} \leftarrow \mathcal{F}(\boldsymbol{X})$ ▷ Compute frequency scores
  $\mathcal{B} \leftarrow$ BlockPartition($\boldsymbol{X}, b$) ▷ Partition into $b \times b$ blocks
  $\mathcal{B}_{\text{smooth}} \leftarrow$ SelectLowestFreq($\mathcal{B}, \boldsymbol{f}, r$) ▷ Select lowest r fraction blocks
  $\boldsymbol{X}^{\text{merged}} \leftarrow$ MergeBlocks($\boldsymbol{X}, \mathcal{B}_{\text{smooth}}$) ▷ Average selected blocks
  $\boldsymbol{Z} \leftarrow$ Process($\boldsymbol{X}^{\text{merged}}$) ▷ Apply attention
  return RestoreBlocks($\boldsymbol{Z}, \mathcal{B}_{\text{smooth}}$) ▷ Restore dimensions
end function

$D_{\alpha,s}$ as defined in Eq. 8 of the main paper. This operator blends nearest-neighbor sampling (preserving detail) with average pooling (smoothing), controlled by the parameter $\alpha \in \mathbb{R}$. Keys and values are downsampled as $\tilde{K} = D_{\alpha,s}(K)$ and $\tilde{V} = D_{\alpha,s}(V)$, while queries remain full resolution.

C.5. Additional Ablations and Results

We provide four targeted supplementary studies to further clarify the robustness and scope of our results: (i) robustness to the ImageNet-1K evaluation subset size, (ii) sensitivity of IE-KVD to α and its timestep schedule, (iii) compatibility between the merging and downsampling modules, and (iv) actual wall-clock speedups. Throughout this subsection, **BiGain$_{\text{TM}}$** refers to the merging module (L-GTM),

Algorithm 4 **BiGain$_{\text{TD}}$**: Interpolate-Extrapolate KV-Downsampling (IE-KVD)

Require: Tokens $\boldsymbol{X} \in \mathbb{R}^{N\times d}$, downsample factor s, interpolation-extrapolation factor $\alpha \in \mathbb{R}$

function BiGainDownsample($\boldsymbol{X}$, s, α)
  $\boldsymbol{Q} \leftarrow \boldsymbol{X}\boldsymbol{W}_Q$ ▷ Queries at full resolution
  $\boldsymbol{K} \leftarrow \boldsymbol{X}\boldsymbol{W}_K, \quad \boldsymbol{V} \leftarrow \boldsymbol{X}\boldsymbol{W}_V$ ▷ Keys and values
  $\tilde{\boldsymbol{K}} \leftarrow$ Interpolate/ExtrapolateDownsample($\boldsymbol{K}, s, \alpha$)
  $\tilde{\boldsymbol{V}} \leftarrow$ Interpolate/ExtrapolateDownsample($\boldsymbol{V}, s, \alpha$)
  $\boldsymbol{Z} \leftarrow$ Attention($\boldsymbol{Q}, \tilde{\boldsymbol{K}}, \tilde{\boldsymbol{V}}$)
  return $\boldsymbol{Z}$ ▷ Output remains at full resolution
end function

while **BiGain$_{\text{TD}}$** refers to the KV-downsampling module (IE-KVD). Unless otherwise stated, we follow the same evaluation protocol as in the main paper. For SD-2.0, we report diffusion-classifier Top-1 accuracy (Acc@1, %) and generation quality (FID). For merging-based methods, the merge ratio is denoted by r; for downsampling-based methods, the downsampling factor is denoted by s. For IE-KVD, α controls interpolation/extrapolation in KV downsampling; for generation we also consider linear schedules ($\alpha_{\text{start}} \to \alpha_{\text{end}}$). Wall-clock times are measured on a single RTX 4090 with FP16 and batch size 4 over 50 denoising steps, we report the U-Net step time averaged over 50 runs (VAE excluded).

Robustness to evaluation subset size. Table 6 addresses the concern that the ImageNet-1K classification results in the main paper are evaluated on a 2K subset for efficiency. We therefore expand the evaluation to 10K images in Table 6 and find that the relative ranking of methods remains

Table 6. **ImageNet-1K subset robustness.** Diffusion-classifier accuracy on 2K and 10K subsets with 95% Wilson confidence intervals.

Method	No accel.	ToMe (70%)	$\text{BiGain}_{\text{TM}}$ (70%)	ToDo (2×)	$\text{BiGain}_{\text{TD}}$ (2×)
Acc. (n=2000) (%) ↑	57.05 ± 2.17	37.35 ± 2.13	44.50 ± 2.19	55.75 ± 2.17	56.30 ± 2.17
Acc. (n=10000) (%) ↑	57.92 ± 0.97	39.85 ± 0.96	45.25 ± 0.98	56.50 ± 0.97	57.59 ± 0.97

Table 7. **IE-KVD α sensitivity.** SD-2.0 / Oxford-IIIT Pets diffusion classification accuracy (Acc., %).

α	0.0	0.5	0.8	0.9 (**$\text{BiGain}_{\text{TD}}$**)	1.0	1.2
Acc. (s=2) (%) ↑	77.02	78.74	79.97	81.52	81.30	64.27
Acc. (s=4) (%) ↑	71.26	75.63	77.48	78.03	77.46	45.46

Table 8. **IE-KVD schedule ablation.** SD-2.0 / Oxford-IIIT Pets generation quality (FID ↓) under different linear schedules.

Linear schedule ($\alpha_{\text{start}} \to \alpha_{\text{end}}$)	FID ↓
ToDo	33.52
IE-KVD: $0.2 \to 0.8$	35.56
IE-KVD: $0.5 \to 1.0$	33.95
IE-KVD: $0.7 \to 1.0$	33.89
IE-KVD: $0.8 \to 1.2$	32.19
IE-KVD: $0.0 \to 1.2$	32.54

unchanged, leading to the same qualitative conclusion. This indicates that the main findings are robust and not an artifact of the smaller subset.

Sensitivity of IE-KVD hyperparameters. Tables 7 and 8 examine the two main IE-KVD design choices: the interpolation–extrapolation parameter α and its timestep schedule. Accuracy is strongest around $\alpha \in [0.8, 1.0]$, showing that the default choice $\alpha = 0.9$ is not fragile, while overly aggressive extrapolation ($\alpha = 1.2$) degrades performance. For generation, FID varies only modestly across linear schedules, suggesting that IE-KVD is reasonably robust to the exact schedule.

Joint use of merging and downsampling. Table 9 studies two hybrid placements: L-GTM in the encoder with IE-KVD in the decoder, and the reverse. Both combinations remain stable for classification and generation, but neither exceeds the best single-module setting. This suggests that the two operators are compatible, although their gains are not simply additive.

Measured runtime. Finally, Table 10 reports actual U-Net step time in addition to FLOPs. This complements Table 3, which focuses on merging-based batch-time measurements, by providing a unified runtime comparison that also includes downsampling and hybrid settings. The empirical speedups follow the same trend as the FLOP reductions, confirming that the proposed operators translate into real inference-time gains.

Table 9. **Joint use of L-GTM and IE-KVD.** SD-2.0 / Oxford-IIIT Pets with r=0.7, s=2, α=0.9.

Method	Acc@1 ↑ (%)	FID ↓
No accel.	81.03	35.01
ToMe	65.76	38.35
ToDo	81.30	33.52
$\text{BiGain}_{\text{TM}}$	74.63	37.73
$\text{BiGain}_{\text{TD}}$	81.52	32.19
L-GTM(enc) + IE-KVD(dec)	79.53	34.84
IE-KVD(enc) + L-GTM(dec)	79.23	36.90

Table 10. **Wall-clock UNet step time.** SD-2.0 on RTX 4090 (FP16, batch=4, 50 steps; averaged over 50 runs; VAE excluded).

Method	Time ↓ (ms/step)	Speedup ↑ (×)	FLOPs ↓ (G)
No accel.	235.65	1.00	804.26
ToMe	144.31	1.63	704.87
ToDo	145.50	1.62	717.44
$\text{BiGain}_{\text{TM}}$	142.88	1.65	704.99
$\text{BiGain}_{\text{TD}}$	150.30	1.57	717.44
L-GTM(enc) + IE-KVD(dec)	147.43	1.60	716.23
IE-KVD(enc) + L-GTM(dec)	146.39	1.61	712.49

D. Use of Large Language Models

We used an LLM to help solely polish the writing of the paper, while all ideas and experiments are conceived and carried out entirely by the authors.

References

[1] Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, et al. Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv e-prints*, pages arXiv–2506, 2025. 3

[2] Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. In *Proceedings of the IEEE/CVF conference on*

computer vision and pattern recognition, pages 4599–4603, 2023. 3, 6

[3] Alexander C Li, Mihir Prabhudesai, Shivam Duggal, Ellis Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2206–2217, 2023. 1, 3, 4, 5

[4] 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. 4

[5] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012. 4

[6] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 5

[7] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023. 4

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 4, 5

[9] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 2015. 4

[10] Maximilian Seitzer. pytorch-fid: FID score for PyTorch. *GitHub repository*, 2023. Version 0.3.0. 4

[11] Ethan Smith, Nayan Saxena, and Aninda Saha. Todo: Token downsampling for efficient generation of high-resolution images. *arXiv preprint arXiv:2402.13573*, 2024. 6

[12] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European conference on computer vision*, pages 776–794. Springer, 2020. 4

[13] Evelyn Zhang, Jiayi Tang, Xuefei Ning, and Linfeng Zhang. Training-free and hardware-friendly acceleration for diffusion models via similarity-based token pruning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025. 6