

Chimera: Designing and Chinchilla-Scaling Hybrid Visual Diffusion Transformers

Chongjian Ge^{*§†}, Hanwen Jiang^{*§}, Tianyu Wang^{*§}, Jiuxiang Gu[§], Yiran Xu[§], Ziwen Chen[§], Shaoteng Liu, Jing Shi, Yicong Hong, Zefan Cai, Hailin Jin, Hao Tan[†]

Adobe Research

^{*}Equal Contribution, [§]Core Contribution, [†]Project Lead

Visual generation is entering a token-intensive regime: high-resolution images, long videos, and multimodal context make the quadratic cost of full attention in Diffusion Transformers increasingly prohibitive. Language models underwent a similar transition, but solutions developed for them cannot be transferred directly to visual diffusion models, which must preserve spatiotemporal locality and support bidirectional interactions across modalities. We present **Chimera**, a hybrid visual diffusion backbone co-designed with a principled scaling recipe. **Chimera** processes text, images, and video tokens in a single stream with three complementary mechanisms: Kimi Delta Attention (KDA) provides long-context state tracking with efficient $\mathcal{O}(N)$ computation complexity, interleaved Multi-head Latent Attention (MLA) enables direct global interaction, and modality-aware short convolutions capture local spatiotemporal context. Together, these components enable processing of a unified raster-ordered token sequence free of positional embeddings. Sparse Mixture-of-Experts (MoE) layers further expand model capacity while keeping activated compute under control. To scale this heterogeneous architecture, we introduce HeteroP, a module-wise scaling scheme that transfers hyperparameters across width and depth based on each tensor’s functional fan-in and the model’s depth. HeteroP produces a consistently tuned model family, enabling us to fit Chinchilla-style compute-optimal laws for activated model size, training-token count and image–video data ratio. Guided by these laws, we train an 11B-parameter **Chimera** with 2B activated parameters. Experiments yield three main findings. First, measured by pretraining diffusion loss, our dense backbone achieves $1.7\times$ the compute efficiency of a matched full-attention Wan-2.1 (2B) baseline, while the complete system raises this factor to $7.3\times$. Second, without length-specific fine-tuning, **Chimera** exhibits zero-shot length extrapolation from 5-second training clips to **30-second** videos, with only a 6.5% degradation in FID over the final five seconds. Third, the fitted scaling laws indicate that compute-optimal image pretraining allocates compute nearly evenly between activated model size and training-token count, whereas video pretraining modestly favors model size at higher compute budgets. Together, we hope these findings provide a principled foundation for designing and scaling efficient long-context diffusion architectures.

Date: July 25th, 2026



1 Introduction

Visual generation is becoming a long-sequence problem. A 4096×4096 image patchified at 16×16 already contains roughly 65K visual tokens, while videos, dense conditions, and multi-turn interactions increase the sequence length further [7, 63, 67]. We refer to this as the token-extensive regime, in which sequence length becomes a primary determinant of memory, latency, and training cost. Yet progress in this regime is constrained by two coupled gaps. Architecturally, the dominant Diffusion Transformer still relies on full self-attention [63], whose pairwise interactions scale quadratically with sequence length [81]. Methodologically, visual generation lacks a systematic framework for scaling such architectures: models and optimization recipes are often designed at a particular scale, without a controlled model family for establishing scaling laws that guide how diffusion-pretraining performance depends on model size, token budget, and data composition [50, 103].

Language models entered this regime earlier and offer useful precedents on both fronts. Architecturally, language models have increasingly moved beyond dense attention [81] toward sparse attention [14, 54, 97], compressed key-value representations [1, 52, 69], recurrent or linear sequence operators [29, 45, 47, 75, 94], and sparsely activated capacity [17, 25, 48]. Methodologically, hyperparameter transfer across model scales [18, 93] and compute-optimal

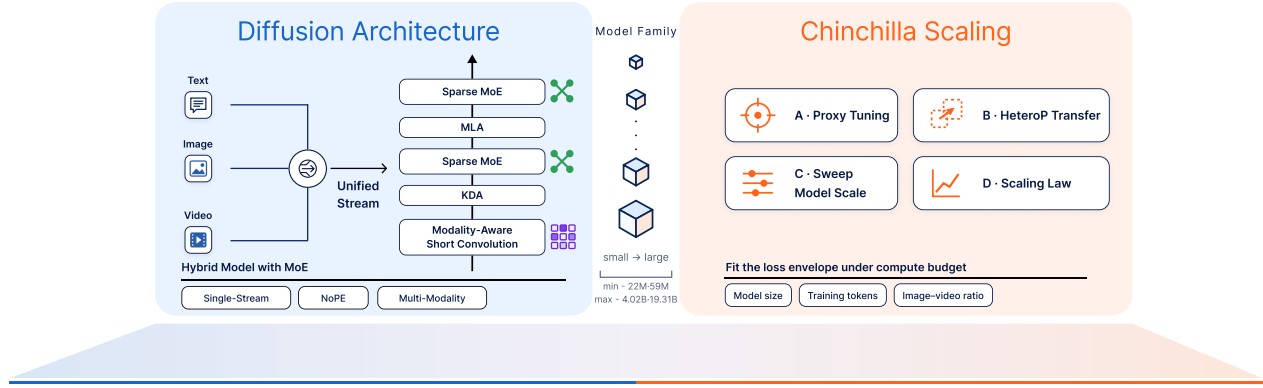


Figure 1 Overview of Chimera’s architecture and scaling framework. **Chimera** couples a hybrid-linear-global diffusion architecture (top left) with a Chinchilla-style scaling methodology (top right). The lower panels show both contributions: the architecture improves pre-training compute efficiency (A1) and long-sequence memory/latency efficiency (A2), while the scaling framework enables reliable cross-scale hyperparameter transfer (S1) and yields a predictable loss-compute frontier (S2).

scaling laws that relate model capacity, data, and training compute [36, 44] have turned model scaling into a systematic and increasingly predictable process.

However, neither component of the language-model recipe can be transferred directly to visual diffusion training. On the architectural side, visual diffusion must preserve local structure over two- and three-dimensional grids, enable direct interaction among heterogeneous text, image, and video contexts, and remain effective under a denoising objective [8, 33, 35, 43]. On the scaling side, hybrid visual backbones contain tensors whose functional input dimensions grow differently, or do not grow at all, as nominal model width increases, so standard rules based on a single global scaling ratio do not preserve comparable optimization dynamics [18, 93]. Moreover, visual training compute depends not only on the number of tokens, but also on data composition. The challenge is therefore twofold yet coupled: *to design an efficient context-modeling architecture tailored to the visual data structure, and to develop a systematic framework for scaling it and deriving its scaling laws.*

We present **Chimera**, a visual diffusion backbone co-designed with a principled scaling framework for token-extensive generation. As shown in Fig. 1, **Chimera** represents text, images, and videos as a unified token sequence for modeling. At the sequence scale, most layers use Kimi Delta Attention (KDA) [47] as a linear-complexity mechanism for content-adaptive long-range state tracking, while periodic Multi-head Latent Attention (MLA) layers [52] provide direct global token-to-token interaction with compressed key-value representations. At the local scale, we propose modality-aware short convolutions aggregate neighbors along the native axes of each modality before every KDA state update. KDA therefore writes locally enriched features, rather than isolated tokens from the flattened sequence, into its recurrent state. The local relative-offset cues supplied by the convolutions, together with the ordering inherent in KDA’s causal recurrence, make a simple temporal-major raster scan sufficient and allow us to dispense with explicit positional embeddings. Orthogonally, **Chimera** combines sparse Mixture-of-Experts (MoE), identity hyper-connections (iHC), and sandwich normalization for greater capacity at controlled activated compute and stable signal propagation.

On the training side, to address the challenge of transferring hyperparameters across scales in our heterogeneous architecture [18, 93], we introduce HeteroP, a hyperparameter-transfer parameterization that derives a separate scaling ratio for each tensor from its functional fan-in and model depth. HeteroP translates these tensor-wise ratios into module-specific transfer rules for learning rates, initialization, and residual scaling, allowing a recipe tuned on a small proxy model to transfer across model widths and depths. Together, these rules produce a consistently optimized and mutually comparable family of **Chimera** models (59M to 19.3B total parameters) without per-scale retuning. This

controlled model family enables fitting Chinchilla-style compute-optimal scaling laws [36] over activated parameter count and visual training tokens. We further extend these laws by treating the image–video data mixture as an explicit scaling variable, capturing how the compute-optimal allocation depends on modality composition.

Across three independent estimators, the training-curve envelope, IsoFLOP profiles, and parametric loss-surface fitting, our scaling laws consistently prescribe nearly balanced growth of activated model size and training tokens for image pre-training (Fig. 1-S2), with $N_{\text{opt}} \propto C^{0.48-0.52}$, while video pre-training shifts modestly toward model capacity at higher compute budgets, with $N_{\text{opt}} \propto C^{0.53-0.56}$. Guided by the fitted scaling laws, we train a **Chimera** model with 11B total parameters and 2B activated parameters given our training budget. Under matched training compute (Fig. 1-A1), it reaches the training loss of a full-attention Wan 2.1 (2B) baseline [82] with $7.3\times$ fewer FLOPs. **Chimera** is also competitive with strong image generators such as FLUX.1-dev and Z-Image-Turbo [8] on the GenEval and DPG-Bench benchmarks. Notably, our final model is trained only around 600 H100 Days, substantially less than prior method: Z-Image-Turbo, for example, reports a total training budget of approximately 12.4K H100 Days, 20 times of ours.

Beyond guiding the final model allocation, **Chimera**’s architecture brings concrete advantages in long-sequence generation. Benefiting from its position-embedding-free design, **Chimera** generates 30-second videos despite being trained only on 5-second clips, requiring no length-specific fine-tuning and extending the generation horizon by $6\times$ while increasing frame-wise FID by only 6.5%, compared with an increase of more than 50% for prior arts [82]. Moreover, compared with the matched MHA/MLA full-attention counterpart, our KDA/MLA backbone supports more than $1.68\times$ longer sequences within a single 80 GB GPU (NVIDIA A100-SXM4-80GB) and runs $2.14\times$ faster at 255k tokens (Fig. 1-A2).¹ Together, these results support jointly considering architecture and scaling for visual generation.

Our contributions are threefold. First, we introduce **Chimera**, a hybrid single-stream visual diffusion backbone for token-extensive generation. It represents text, image, and video as a unified sequence. Our model combines KDA linear attention with periodic MLA-based global attention; uses modality-aware causal short convolutions to support a single raster-order scan without positional embeddings; and adopts MoE layers, identity hyper-connections, and sandwich normalization for capacity and training stability. Second, we develop, to our knowledge, the first comprehensive hyperparameter-transfer parameterization (HeteroP) for heterogeneous visual diffusion backbones. HeteroP allows optimization hyperparameters tuned on a small proxy model to transfer reliably across model widths and depths. Third, we fit Chinchilla-style compute-optimal scaling laws for image and video pre-training over activated model size and visual training tokens, and extend the analysis to the image–video data composition.

2 Chimera Overview

We aim to modernize diffusion models, particularly DiT [63], for the token-extensive visual generation regime. This objective involves two tightly coupled sub-goals: designing architectures that balance efficiency and capacity, and understanding how their advantages evolve with model size, the number of training tokens, and training-data composition. Accordingly, we study architecture design and scaling jointly: the architecture defines a controlled model family, while the scaling analysis determines how model size, training tokens, and training-data composition should be allocated under a fixed compute budget. Section 3 details the architecture, while Sec. 4 presents the scaling solution.

Model Architecture. We summarize the key design choices of **Chimera** below, ordered from the overall framework to individual modules:

- **Single-stream Multimodal Backbone.** **Chimera** concatenates multimodal tokens from text, images, and video into one sequence and processes them with a single shared backbone. Unlike cross-attention or dual-stream designs [82, 87], this single-stream interface enables direct token-level interaction across modalities

¹We compare matched KDA/MLA and MHA/MLA 3:1 backbones with approximately 2B activated parameters, using batch size one, BF16 activations, 512 text tokens, and 18×28 visual tokens per temporal slice. For the roll-out-cache memory test, we allocate, touch, and retain the cache tensors on device during one complete denoiser forward. FlashAttention does not write the full $N\times N$ attention matrix to high-bandwidth memory: it streams blocks of Q , K , and V through on-chip SRAM, computes $QK^\top$ tile by tile, and accumulates the exact output using online softmax. This removes the quadratic attention workspace, but not the quadratic arithmetic. The 24 MHA layers therefore have sequence-dependent BF16 key–value cache memory of $\mathcal{O}(NHd_h) = \mathcal{O}(ND)$, where H and d_h are the number and width of the attention heads and $D = Hd_h$. In contrast, the 24 KDA layers retain H fixed-size FP32 recurrent states of shape $d_h\times d_h$, requiring $\mathcal{O}(Hd_h^2)$ memory independent of N ; this is the $D\times D$ state-memory term when D denotes the per-head state width. Both variants retain the same eight BF16 MLA latent key–value caches. We report peak allocated CUDA memory on one NVIDIA A100-SXM4-80GB and stop at the first out-of-memory input. For speed, we exclude cache construction, use FlashAttention for MHA, and report the median of three CUDA-synchronized post-warm-up forwards.

without modality-specific branches. It also simplifies scaling: a shared stack over one token sequence inherits language-model practices such as hyperparameter transfer and compute-optimal scaling laws.

- **Hybrid Linear and Global Attention.** **Chimera** uses KDA [47] in most layers for $\mathcal{O}(N)$ long-sequence computation and periodically interleaves MLA layers that restore explicit global token interactions through compressed key-value states. Together, they balance long-context efficiency and global modeling capacity. Unlike prior works that rely on complex scanning patterns for visual modeling [39, 56], **Chimera** applies linear attention through a **single temporal-major raster scan** for simple and uniform multimodal processing.
- **Minimal Modality-aware Inductive Bias and No Positional Embedding (NoPE).** Attention-based models typically use positional embeddings to encode the token positions. For visual data, however, positional embeddings can hinder extrapolation beyond the training length, and factorizing them across spatial (image) and temporal axes (video) requires ad-hoc design choices. The state-tracking property of KDA offers a simple alternative: with fine-grained channel-wise state decay and recurrent delta updates, KDA behaves like local sliding-window attention with *an adaptive window length and causal structure*, which removes the need for positional embeddings, as observed in recent language models [47]. To adapt this design to multimodal visual data, we retain a single minimal inductive bias: modality-aware short convolutions that serve as a “Canon Layer” [2] to capture local spatio-temporal dependencies among visual tokens. Order and locality thus come from the scan and the convolution, so the whole backbone processes multimodal tokens through one temporal-major raster scan without positional embeddings.
- **Sparse Capacity and Training Stability.** For capacity, **Chimera** adopts sparse MoE activation [25], which increases total parameters while controlling per-token computation. For stability, it pairs identity hyperconnections [90], which stabilize the residual streams, with sandwich normalization [22], which regulates the variance of sub-layer outputs.

Pre-training and Chinchilla Scaling. We study how **Chimera** scales during pre-training, the stage in which the model acquires its generative knowledge. We argue that the diffusion pre-training loss should be the reliable metric for measuring knowledge acquisition at this stage, while other vision-related evaluation metrics² are better treated as post-training metrics rather than pre-training objectives. This perspective allows us to treat visual pre-training analogously to language-model pre-training and fit compute-optimal scaling laws. We outline the key ingredients below.

- **Diffusion Loss as a Pre-training Metric.** Under a fixed diffusion objective and noise parameterization, diffusion loss provides a sampler-independent metric for comparing models across scales. Unlike downstream generation metrics such as FID, it does not depend on sampling algorithms, classifier-free guidance scales, or post-training choices. Moreover, diffusion can be interpreted as approximate autoregressive modeling in the frequency domain [20]: denoising from high to low noise levels predicts visual information from coarse to fine scales, which parallels next-token prediction under progressively longer contexts. This motivates its use for a more faithful pre-training metric than downstream generation scores.
- **Hyperparameter Transfer and Fair Scaling Comparison.** Hyperparameter transfer reduces the cost of hyperparameter search at the target scale and helps every model in a scaling study reach near-optimal performance. We build on μ P-style transfer [18, 93] across model scales, but **Chimera** poses an additional challenge: its heterogeneous modules do not share a single global scaling multiplier. We therefore propose HeteroP to replace this global-ratio approximation with a module-specific ratio derived from each parameter group’s functional fan-in, covering KDA, MLA, sparse MoE, identity hyperconnections, and diffusion conditioning. As a result, the data points used to fit scaling curves come from models trained with scale-appropriate and near-optimal hyperparameters, a factor often not explicitly controlled in prior scaling-law studies [36, 50, 86].
- **Fitting Chinchilla Scaling Laws.** Using this controlled model family, we fit Chinchilla-style laws over activated model parameters and training tokens to predict their compute-optimal allocation under a fixed compute budget. We also use IsoFLOP analysis to estimate how the optimal image-video mixture changes with compute.

²FID, DPG [40], GenEval [28], etc

3 Chimera Architecture

3.1 Preliminaries and Training Objectives

Chimera follows the rectified-flow framework with v -prediction and v -loss [51, 55]. Each image or video is encoded by a frozen VAE from [82] with temporal, height, and width compression factors of (4, 8, 8); a subsequent patchification layer converts the latents into the clean visual tokens z_1 . The text prompt is encoded separately by a frozen T5-style encoder [65], and we denote the resulting text-token sequence by c . During training, we sample Gaussian noise $z_0 \sim \mathcal{N}(0, I)$ and a timestep τ , and linearly interpolate between z_0 and z_1 to obtain the noised visual tokens $z_\tau = \tau z_1 + (1 - \tau)z_0$. The velocity target is $v_\tau = \frac{dz_\tau}{d\tau} = z_1 - z_0$. We train the denoising model with

$$\mathcal{L} = \mathbb{E}_{\tau, z_0, z_1} \|\text{Chimera}(c, z_\tau, \tau; \theta) - v_\tau\|_2^2, \quad (1)$$

where θ denotes the trainable parameters and the loss is calculated only at visual-token positions.

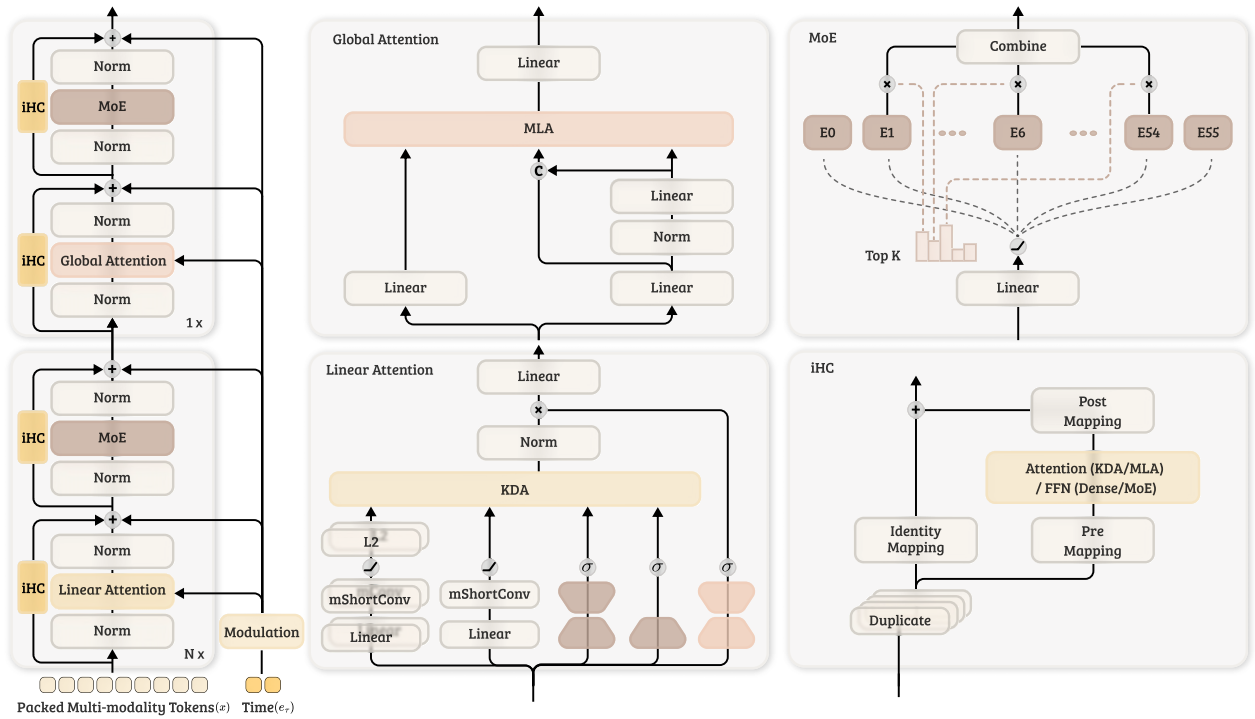


Figure 2 Illustration of the Chimera architecture. **Chimera** is a single-stream latent diffusion Transformer in which text tokens and noised visual tokens are concatenated into a unified sequence and processed jointly. Each block combines a sparse Mixture-of-Experts feed-forward network with either a KDA layer or a MLA layer. Before each KDA update, a modality-aware short convolution (mShortConv) mixes local context in each modality’s native geometry. Timestep information is injected through AdaLN modulation. Each block applies sandwich-style RMSNorm (pre- and post-normalization), while identity hyper-connections (iHC) wrap each sublayer to stabilize signal propagation.

3.2 Single-Stream Framework

Chimera adopts a single-stream framework in which text and visual tokens interact directly within one shared backbone. Its input sequence is

$$x = [P_{\text{text}}(c), P_{\text{vis}}(z_\tau)], \quad (2)$$

where P_{text} and P_{vis} are separate trainable projections into the shared hidden space, and padding tokens are removed from c before concatenation. The visual tokens are flattened frame by frame in temporal-major raster order, traversing each frame in row-major order, as formalized in Eq. 12. For the diffusion timestep τ , we construct a sinusoidal embedding

and project it to the model dimension, yielding the timestep embedding e_τ , which conditions each block through the modulation pathway defined below.

Chimera consists of a stack of Transformer-style blocks, each containing an attention sub-layer followed by a feed-forward sub-layer (FFN). The attention operator is either KDA or MLA, following the periodic schedule (Sec. 3.3). The first and last blocks use dense SwiGLU FFNs [70], while all intermediate blocks use sparse MoE FFNs (Sec. 3.5). Before each KDA update, a modality-aware short convolution (mShortConv) injects local structure (Sec. 3.4). Both sub-layers use sandwich normalization [22] and are wrapped with $M = 4$ stream identity hyper-connections (iHC; Sec. 3.6). With a slight abuse of notation, we continue to write x for this multi-stream state, whose structure is made explicit in Sec. 3.6. Timestep conditioning enters through adaptive layer normalization (AdaLN) modulation and residual gates. Omitting the layer index, one block computes:

$$(\delta_A, \gamma_A, g^A, g^F) = \text{MLP}_{\text{mod}}(e_\tau), \quad (3)$$

$$\tilde{x}_A = \text{iHC-Read}_A(x), \quad (4)$$

$$\hat{x}_A = \text{RMSNorm}_A^{\text{pre}}(\tilde{x}_A) \odot (1 + \gamma_A) + \delta_A, \quad (5)$$

$$a = \text{RMSNorm}_A^{\text{post}}(\text{Attn}(\hat{x}_A)), \quad (6)$$

$$x' = \text{iHC-Write}_A(x, g^A \odot a), \quad (7)$$

$$\tilde{x}_F = \text{iHC-Read}_F(x'), \quad (8)$$

$$f = \text{RMSNorm}_F^{\text{post}}(\text{FFN}(\text{RMSNorm}_F^{\text{pre}}(\tilde{x}_F))), \quad (9)$$

$$x'' = \text{iHC-Write}_F(x', g^F \odot f). \quad (10)$$

Here, $x, x', x'' \in \mathbb{R}^{L \times M \times d}$ denote the multi-stream states, whereas $\tilde{x}_A, \hat{x}_A, a, \tilde{x}_F, f \in \mathbb{R}^{L \times d}$. The iHC-Read operators collapse the stream axis from $\mathbb{R}^{L \times M \times d}$ to $\mathbb{R}^{L \times d}$, whereas iHC-Write distributes each sub-layer update back to the M streams. The timestep-conditioned vectors $e_\tau, \delta_A, \gamma_A, g^A, g^F \in \mathbb{R}^d$ are broadcast across the L token positions. The state x'' becomes the input to the next block. Equation 3 maps e_τ to an attention shift δ_A , an attention scale γ_A , and residual gates g^A and g^F for the two sub-layers. Thus, the attention input receives timestep-conditioned AdaLN modulation [63], while both residual updates are timestep-gated (Eqs. 7 and 10). With $M = 4$ parallel residual streams, both sub-layers share the same *read–compute–write* pattern over the streams. Each sub-layer starts with iHC-Read (Eqs. 4 and 8), which aggregates the M streams into one sub-layer input. For attention, this input is pre-normalized and timestep-modulated, passed to the attention operator (Eq. 5 and 6). The resulting update is then gated by g^A , and written back to all M streams by iHC-Write (Eq. 7). The FFN sub-layer also repeats this pattern (Eqs. 8–10). Within each sub-layer, Sandwich Normalization places RMSNorm both before ($\text{RMSNorm}^{\text{pre}}$) and after ($\text{RMSNorm}^{\text{post}}$) the operator to regulate the variance for training stability. We define the attention operator $\text{Attn}(\cdot)$ in Sec. 3.3, the feed-forward operator $\text{FFN}(\cdot)$ in Sec. 3.5, and the iHC read/write operators in Sec. 3.6.

After the final block, the M residual streams are averaged into one representation (Sec. 3.6), and only visual-token positions enter the output head for rectified-flow velocity prediction. Text tokens condition these predictions through token-level interactions in the shared backbone, while the diffusion loss applies only to visual tokens.

3.3 Hybrid Attention Mechanism with NoPE

Chimera periodically interleaves Kimi Delta Attention (KDA) [47] and Multi-head Latent Attention (MLA) [52], at a 3:1 KDA-to-MLA ratio. For a sequence of length L , each KDA layer requires only $\mathcal{O}(N)$ sequence complexity because it carries a constant-size recurrent state rather than materializing a full attention map. Each MLA layer retains full bidirectional self-attention with compressed key-value representations. The operators are therefore complementary: KDA provides efficient long-sequence computation, while MLA periodically enables unrestricted bidirectional global interaction. We detail both formulations and the no-positional-embedding (NoPE) design below.

$$\text{Attn}(\hat{x}_A) = \begin{cases} \text{KDA}(\hat{x}_A), & \text{Linear Attention,} \\ \text{MLA}(\hat{x}_A), & \text{Global Attention.} \end{cases} \quad (11)$$

KDA with a Single Scan Pattern. We traverse video latents frame by frame, using row-major order within each frame. For a latent grid of temporal length T , height H , and width W , this temporal-major raster order is

$$z^{i,j,k} \mapsto z^m, \quad m = iHW + jW + k, \quad (12)$$

where i , j , and k index the temporal, height, and width dimensions, respectively. An image is treated as a single-frame latent grid. We omit the timestep subscript τ below for clarity.

After text tokens are prepended to the flattened visual tokens, every KDA head processes $\hat{x}_A$ using the same scan order. The recurrent state update is causal in this order: at each visual position, the state summarizes the text prefix and previously scanned visual tokens. The update follows the KDA formulation of Kimi Linear [47], with fine-grained channel-wise state decay rather than a single head-wise scalar. For a single head, omitting its index, we have

$$(Q, K, V) = \text{mShortConv}(W_{\text{kv}}(\hat{x}_A)), \quad (13)$$

$$S_t = (I - \beta_t k_t k_t^\top) \text{Diag}(\alpha_t) S_{t-1} + \beta_t k_t v_t^\top, \quad (14)$$

$$[\text{KDA}(\hat{x}_A)]_t = W_o(\text{Sigmoid}(W_{\text{gate}}(\hat{x}_{A,t})) \odot \text{RMSNorm}(S_t^\top q_t)), \quad (15)$$

where $q_t \in \mathbb{R}^{d_q}$, $k_t \in \mathbb{R}^{d_k}$, and $v_t \in \mathbb{R}^{d_v}$ denote the query, key, and value vectors at scan position t in Q , K , and V , respectively, while $S_t \in \mathbb{R}^{d_k \times d_v}$ is the recurrent state after processing that position. The forget gate $\alpha_t \in (0, 1)^{d_k}$ applies an independent decay to each key channel, while the head-wise scalar $\beta_t \in (0, 1)$ controls both the erase and write terms. The discrete scan index ranges over $t = 1, \dots, L$ and is distinct from the diffusion timestep τ . The recurrent state is reset to $S_0 = 0$ at each packed-sample boundary. $\text{mShortConv}(\cdot)$ denotes the proposed modality-aware short convolution (Sec. 3.4), which supplies local structure and facilitates context-level information shortcut [2], while allowing the recurrent KDA update to retain a single scan order across modalities.

MLA for Global Context Modeling. KDA compresses the scanned history into a constant-size recurrent state. This state is efficient to maintain, but its bounded size limits how much information can be recalled exactly from long visual sequences. We therefore interleave a small fraction of full-attention layers to restore direct content-based interactions across all tokens. This hybrid design is more than a cost-capacity compromise. Recent analysis shows that, under standard complexity assumptions, alternating linear-attention and full-attention layers is strictly more expressive than either type alone on formal sequence task: linear attention contributes state tracking and full attention contributes recall, so their composition solves problems that neither solves by itself [57]. To keep these global layers memory-friendly, we instantiate them with MLA, which retains full bidirectional attention while compressing the key-value representations:

$$(Q, U, K^{\text{direct}}) = (W_q(\hat{x}_A), W_{\text{kv}}^{\text{down}}(\hat{x}_A), W_k^{\text{direct}}(\hat{x}_A)), \quad (16)$$

$$[K^{\text{lat}}, V] = W_{\text{kv}}^{\text{up}}(\text{RMSNorm}(U)), \quad (17)$$

$$K_i = [K^{\text{direct}}, K_i^{\text{lat}}], \quad i = 1, \dots, n_h, \quad (18)$$

$$\text{MLA}(\hat{x}_A) = W_o(\text{MHSA}(Q, K, V)). \quad (19)$$

Here, n_h denotes the number of attention heads. U is the compressed key-value latent. Both K^{lat} and V are decoded from U , whereas K^{direct} is projected directly from $\hat{x}_A$ and shared across heads. This reduces the key-value cache cost while retaining full bidirectional attention. Unlike standard MLA [52], we leave K^{direct} unrotated (NoPE).

NoPE v.s. RoPE. **Chimera** uses no explicit positional encoding (NoPE) in either KDA or MLA layers. To motivate this choice, we first examine what positional encoding contributes to attention. Absent explicit positional modulation, attention is a content-matching operation. That’s to say, its logits are inner products between learned query and key features, without direct dependence on token indices. Positional encodings add position-dependent inductive biases to this content pathway. For RoPE [74], the *de facto* choice in both language and visual generation models, these biases take three forms:

- **Position selection.** Some operations must activate at a specific relative token offset. A canonical example is the previous-token head, which serves as a building block of induction heads [23, 61]. RoPE enables such *token-index-aware/token-level-shortcut* operations by rotating each query-key channel pair at a fixed frequency, allowing trained heads to exploit high-frequency pairs for precise offset selection. By aligning the phases of its highest-frequency pairs, an attention head can make its logit peak at a chosen small offset [3].

- **Recency decay.** Nearby context should generally receive greater weight than distant context. Positional encoding schemes introduce this bias either explicitly, as with ALiBi’s distance-dependent penalty [64], or implicitly, as with RoPE. Under the conventional frequency spectrum and simplifying assumptions, RoPE’s rotations can induce a distance-decaying envelope on the attention logits. This *long-term decay* was a key motivation for RoPE [74].
- **Multidimensional encoding.** Multimodal data has inherently multidimensional positions: text tokens have a one-dimensional index, whereas video tokens use temporal, height, and width coordinates. Axial RoPE assigns rotary query-key pairs to these axes and rotates each subset by its coordinate [82, 85, 95].

All three biases are implemented by rotating content-bearing query-key features, forcing positional and semantic information to share query-key capacity. Trained models mitigate this competition by separating the two across the frequency spectrum [3]. Low-frequency pairs remain nearly rotation-invariant over the relevant context range, so their dot products are governed primarily by token content and preserve distance-robust semantic matching. Short-range position selection, by contrast, are concentrated in the high-frequency pairs.

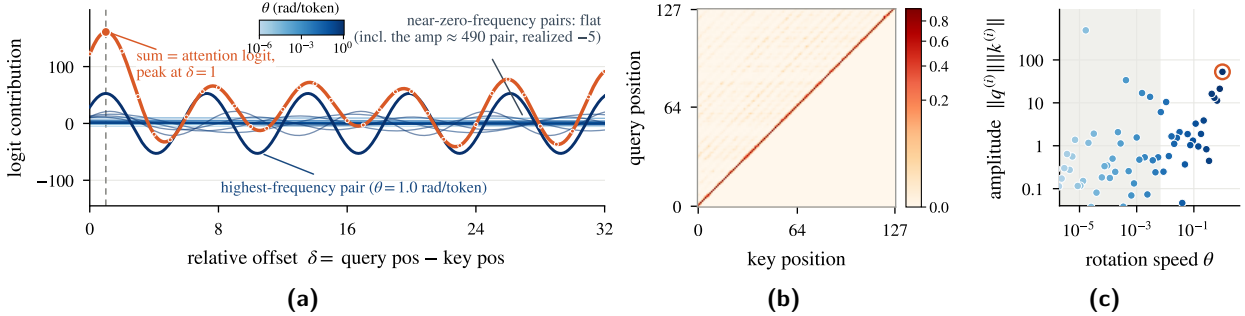


Figure 3 RoPE analysis of the strongest previous-token head in Qwen3-4B (layer 0, head 1). (a) Per-pair logit contributions (darker blue indicates faster rotation) and their sum (orange), which peaks exactly at $\delta=1$. (b) Attention map over the first 128 tokens (power-transformed color scale). (c) Pair amplitude versus rotation frequency: pairs in the shaded region rotate less than one full turn within the context window, and the circled highest-frequency pair creates the peak in (a).

To characterize this frequency-wise division of labor, we first analyze a pretrained RoPE-based language model, Qwen3-4B [91], whose previous-token heads provide a clear diagnostic of short-range position selection. We use a 1,024-token window of natural text. Because RoPE rotates each channel pair independently, every attention logit decomposes into a sum of per-frequency contributions, allowing the role of each frequency to be measured directly. Specifically, we average the queries and keys over the window and, for each rotary channel pair i , rotate the query component onto the $+x$ axis while absorbing both vector norms into the pair amplitude $|q^{(i)}| |k^{(i)}|$. The resulting pairwise cosine terms sum exactly to the attention logit, agreeing with the directly computed value to within 3×10^{-13} (Fig. 3a). We first examine the strongest previous-token head (i.e., layer 0, head 1), for which 98.5% of queries attend most strongly to the immediately preceding token (Fig. 3b). In this head, the highest-frequency pair, which rotates by approximately one radian per token, produces a logit that peaks exactly at offset one: its learned phase places the cosine maximum at $\delta \approx 1$. By contrast, the largest-amplitude pairs all lie near zero frequency and contribute almost uniformly across offsets (Fig. 3c). These slowly rotating pairs are therefore modulated primarily by content rather than position. The largest such pair rotates by less than 1° across the entire window, yet contributes little in this example because its averaged query and key vectors are nearly orthogonal. We then search across all heads for those that attend exactly n tokens into the past, quantified by the fraction of queries whose attention peaks at that offset. Such heads appear only at short distances: the best head attains a fraction of 0.56 at $n=2$, but only 0.07 at $n=10$. These results indicate that RoPE-based position selection operates almost entirely over short ranges.

We then apply the same analysis to visual generation models, FLUX.2 [4] and Wan2.2 [82], and observe the same division of labor. That’s to say, slowly rotating channel pairs preserve distance-robust semantic matching, whereas rapidly rotating pairs enable short-range position selection (protocol and full results in Appendix C).

From this perspective, RoPE implements each of its three inductive bias through rotary query-key channels, while introducing several limitations. (1) First, position selection consumes substantial attention capacity. In the previous-token head above, a single fast-rotating pair fixes the target offset, while the large-amplitude slow pairs contribute only a flat, offset-independent background. As a result, an entire attention head is effectively dedicated to a fixed n -token shift, yet reliable selection remains limited to very short ranges. (2) Second, recency decay is only implicit. The decaying

envelope arises as an average property of the summed rotations rather than from an explicit content-adaptive decay mechanism, and can be bypassed by trained heads [3]. (3) Third, layout encoding is hand-designed. The channel partition must be specified manually, each additional modality or positional axis further divides the available channels and reduces the frequency resolution per axis, and the resulting allocation remains fixed at design time.

Chimera instead provides each inductive bias through a dedicated mechanism, none of which consumes attention-channel capacity. Token order is supplied by the recurrent scan itself: KDA processes the sequence in temporal-major raster order, making its state updates inherently order-aware. (1) *Position selection* is handled by the modality-aware short convolution (Sec.3.4). A depthwise kernel mixes tokens at explicit index offsets, implementing index-aware operations with only a small number of parameters rather than dedicating an entire attention head to them. Related convolutional mechanisms have also been used to inject positional information into vision Transformers [15]. Because the analyses above show that RoPE-based position selection is primarily short-range, a short kernel is sufficient to replace it. (2) *Recency decay* is intrinsic to KDA. The channel-wise forget gate α_t (Eq. 14) decays the recurrent state at each scan step, linking each token to its predecessors over a learned, content-adaptive window rather than through a fixed spectral envelope. (3) *Multidimensional encoding* is determined by the structure of the convolution. Text tokens are mixed with a causal 1D kernel along the token index, whereas visual tokens are mixed with a 3D kernel over the temporal-height-width grid. Each modality’s native layout is therefore encoded directly by the operator, without partitioning the channels across positional axes. With these three positional biases assigned to dedicated modules, attention is free to perform its native function: content matching. MLA without RoPE corresponds to the limiting case in which every channel has zero rotary frequency, so its logits depend only on content [3]. The queries and keys in KDA likewise contain no positional phases. This design parallels recent hybrid language models that combine NoPE full attention with gated linear attention [47, 77]. **Chimera** applies it to multimodal data via modality-aware convolution.

Beyond this disentanglement, removing RoPE also avoids its dependence on training length. At relative offsets beyond those observed during training, nonzero rotary frequencies can produce phase configurations outside the training distribution. When a model trained on 5-second clips is used to generate 10- or 20-second videos, these unseen phase configurations can degrade [3, 46]. By contrast, our three mechanisms introduce no explicit positional phase tied to the training range: the convolution kernels have fixed support, and the same recurrent update is applied at each scan step. This, to some extent, removes one architectural obstacle to zero-shot extrapolation to longer sequences.

Overall, **Chimera** factors RoPE’s three positional functions into explicit, specialized modules: a short convolution for position selection, gated state decay for recency, and operator structure for layout encoding. Each inductive bias is provided in a direct and learnable form, without consuming attention capacity for simple positional operations, manually partitioning channels, or coupling the model to its training length.

3.4 Modality-aware Short Convolution

A short convolution injects local structure into the token features before each linear-attention update, following common practice in the linear-attention literature [47]. A plain one-dimensional convolution, however, mixes only tokens that are adjacent in the scan order and thus models locality only along a fixed-order axis. To recover the two- and three-dimensional locality of visual data, prior vision models pair such a convolution with multiple hand-designed scan directions [37, 39, 56]. We instead make the convolution *modality-aware*, so that it matches each modality’s native structure and captures local spatial-temporal correlations directly. This allows **Chimera** to retain a single scan pattern. Fig. 4 contrasts our modality-aware convolution this single scan pattern with the conventional pairing of a 1D convolution and multi-axis scanning.

Concretely, the modality-aware convolution uses modality-specific views of text and visual tokens while preserving their positions in the packed single-stream interface (Fig. 4). Text tokens are gathered from their packed positions and processed with a causal one-dimensional depthwise convolution over the text sequence. Visual tokens are reshaped into their spatial-temporal grid and processed with a depthwise 3D convolution, which models local correlations along the temporal, height, and width axes at once. An image is handled as the special case of a video with temporal length one. The convolutions are applied independently within each modality segment, so no kernel crosses a text-visual or packed-sample boundary. After convolution, the text and visual outputs are scattered back to their original positions in the packed sequence. Because the convolution itself captures multi-dimensional locality, KDA can keep a single temporal-major scan instead of the multi-directional scans discussed above.

Implemented naively, this *gather–convolve–scatter* procedure launches multiple kernels and materializes intermediate

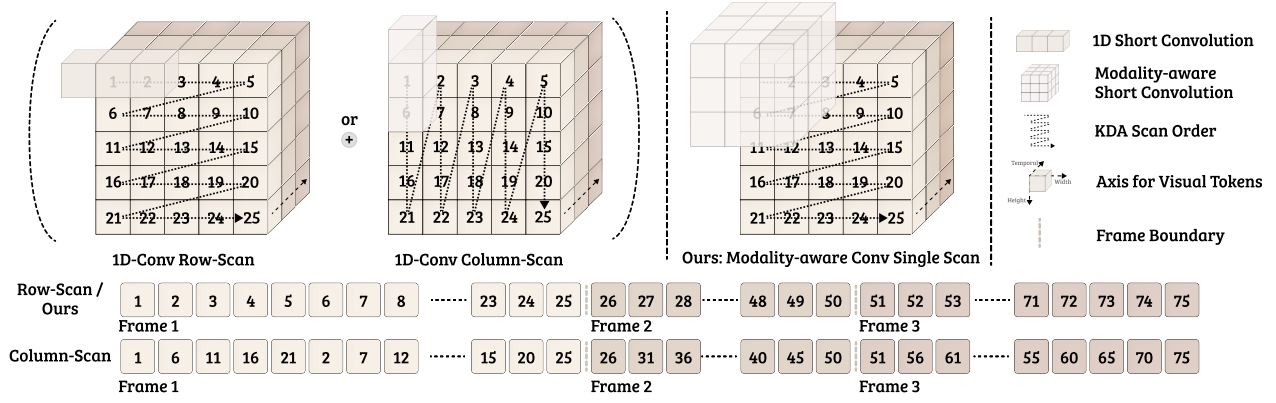


Figure 4 Conceptual comparison of local-mixing designs in vision modality. Conventional 1D short convolution is paired with multi-axis scanning for vision modeling, whereas our modality-aware short convolution models spatial-temporal locality under a single scan pattern

buffers per layer. We instead implement a fused Triton kernel that realizes the same text and visual convolutions directly over packed storage, avoiding explicit gather and scatter operations. In operator-level microbenchmarks against the unfused implementation on packed image (512^2 and 1024^2) and video (21-frame) batches, the fused kernel achieves a $2.2\text{--}2.3\times$ forward speedup and a $1.5\text{--}1.8\times$ forward-backward speedup, while reducing peak activation memory during the forward pass by up to $4\times$.

Along the temporal dimension, the 3D convolution uses causal padding, so that each visual token mixes only with tokens from the current and past frames. We make the short convolution *temporally causal* to keep its local mixing consistent with KDA’s causal temporal-major scan, for the following reasons:

- **Consistency with State Tracking.** The KDA operator updates a recurrent state while scanning the sequence, which aligns naturally with the temporal progression of video generation. A non-causal temporal convolution would introduce future-frame information into the local query, key, and value features before each KDA update, weakening this state-tracking interpretation. Causal padding keeps the local visual operator consistent with the causal dynamics of the KDA scan.
- **Compatibility with Sequential Generation.** A temporally causal convolution also facilitates adapting the pretrained backbone to downstream streaming generation tasks. For example, if the model is later converted into an autoregressive world model, the KDA pathway, including its short convolution, is already causal in time from pretraining, rather than relying on a causal constraint introduced only during post-training [38, 41].
- **Performance.** Empirically, causal temporal padding in the short convolution has little effect on generation quality compared with the non-causal alternative, so this design choice comes at negligible cost.

In summary, KDA follows a causal temporal-major scan, while the short convolution is causal along the temporal axis. This alignment preserves KDA’s state-tracking interpretation, while the convolution provides modality-specific local structure under a single scan pattern.

3.5 Mixture-of-Experts (MoE)

To enhance the capacity of **Chimera** without proportionally increasing activated computation per token, we replace the dense FFN with sparse MoE FFNs [17, 25] in all except the first and last blocks. This choice complements the hybrid KDA/MLA design: once the attention cost is reduced, FFN computation accounts for a larger share of per-token computation, so enlarging dense FFNs would erode the efficiency gains from hybrid attention. Sparse activation instead increases the total FFN parameters, and hence the model capacity, while keeping the activated computation per token matched to a dense baseline.

Concretely, an MoE FFN contains N routed experts $\{E_i\}_{i=1}^N$ and a router $r(\cdot)$. For each token input $u \in \mathbb{R}^d$, the router

produces expert scores, selects a small top- K set $\mathcal{T}(u)$, and combines only those expert outputs,

$$\mathcal{T}(u) = \text{TopK}(r(u) + b, K), \quad (20)$$

$$\text{MoE}(u) = \sum_{i \in \mathcal{T}(u)} g_i(u) E_i(u), \quad (21)$$

where K is the number of experts activated per token, $b \in \mathbb{R}^N$ is a non-gradient balancing bias applied only during expert selection, and $g_i(u)$ is the routing weight of expert i . In intermediate blocks, $\text{MoE}(\cdot)$ instantiates the $\text{FFN}(\cdot)$ operator in Eq. 9. We use *token-choice* routing: each token selects its activated experts independently. Following the fine-grained expert design of DeepSeekMoE [17], each routed expert is a small SwiGLU FFN whose hidden dimension is $1/K$ of the dense FFN hidden width. In our main configuration, every MoE layer has $N = 56$ routed experts and activates $K = 8$ experts per token, corresponding to an activation ratio of $8/56 = 1/7$. The total expert parameters are thus enlarged by roughly $56/8 = 7\times$, while the 8 activated experts together recover exactly the dense FFN hidden width, e.g., $8 \times 1024 = 8192$ in our 2B configuration. We use no shared experts, as we empirically find that the routed experts alone provide sufficient capacity in this visual diffusion setting. The MoE configuration, including expert count and activation ratio, is fixed across experiments, keeping our scaling-study runs directly comparable.

The router is a lightweight two-layer MLP producing sigmoid scores. After top- K selection, the selected scores are normalized and scaled by $\sqrt{K}$ to form the weights $g_i(u)$. To avoid routing collapse and expert-load imbalance during training, we adopt the auxiliary-loss-free balancing strategy [53, 84]: the non-gradient bias b enters only the top- K selection, while the weights $g_i(u)$ are always computed from the unbiased router scores. After each training step, the bias of overloaded experts is decreased and that of underloaded experts is increased by a small fixed step, according to the observed expert loads. This keeps the expert load balanced without adding any auxiliary loss to the diffusion objective.

To verify that this strategy indeed balances the load in our visual diffusion setting, we follow [84] and quantify the degree of load balance of an MoE layer by its *maximal violation* (MaxVio):

$$\text{MaxVio} = \frac{\max_{1 \leq i \leq N} \text{Load}_i - \overline{\text{Load}}}{\overline{\text{Load}}}, \quad (22)$$

where Load_i is the number of tokens routed to expert E_i in a training batch, and $\overline{\text{Load}}$ is the mean load over the N experts, i.e., the per-expert load under perfect balance. MaxVio equals 0 under a perfectly uniform load and reaches $N/K - 1 = 6$ in our configuration when all tokens select the same K experts. Fig. 5 tracks the batch-level MaxVio, averaged over all MoE layers, for two identical runs. With the balancing bias enabled, MaxVio drops below 1 almost immediately and stabilizes around 0.5 for the rest of training, i.e., even the most loaded expert receives only about $1.5\times$ its expected share of tokens. Disabling the bias instead leads to severe imbalance: MaxVio rises above 5 within the first few hundred steps and keeps drifting toward the collapse bound, with the most loaded expert attracting over $6\times$ its expected load. These results confirm that the bias-based mechanism maintains well-balanced expert utilization throughout training without adding a load-balancing term to the diffusion objective.

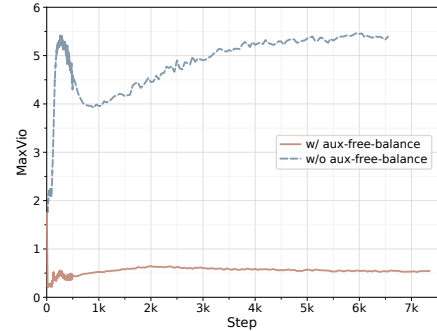


Figure 5 Expert load balance during training. We plot the batch-level MaxVio (Eq. 22), averaged over all MoE layers. With auxiliary-loss-free balancing, MaxVio stays around 0.5 throughout training; without it, MaxVio rises above 5, approaching the collapse bound of $N/K - 1 = 6$.

3.6 Identity Hyper-Connections (iHC)

Deep networks rely on residual connections to preserve signal propagation across depth. Hyper-Connections (HC) [104] generalize this design by expanding the residual state into multiple streams and learning how each sub-layer reads from and writes back to them. mHC [90] further constrains the residual-stream transition to be doubly stochastic, preserving the approximate identity mapping required for stable large-scale optimization. This multi-stream interface is particularly well suited to Chimera. KDA, MLA, modality-aware short convolution, and MoE perform complementary computations and produce updates with different statistics, yet a standard residual block forces every sub-layer to read from and write to the same state. As these heterogeneous updates accumulate with depth, a single stream can restrict how operator-specific

information is preserved and recombined. Multiple residual streams provide token-dependent read/write pathways that increase routing capacity and maintain stable feature statistics through an identity path.

We adopt Identity Hyper-Connection (iHC), a simplified variant of mHC that fixes the residual-stream transition to the identity while retaining token-dependent read and write mappings. At the input of the denoising backbone, iHC expands the standard residual state into M residual streams $R \in \mathbb{R}^{M \times d}$ by replication, where d is the hidden dimension and R is the per-token multi-stream state. Stacking these states across L token positions yields the sequence-level state $x \in \mathbb{R}^{L \times M \times d}$ used in Eqs. 4–10. After the final block, the M streams are averaged back into a single stream. Within each block, before each sub-layer, a read vector $h_{\text{pre}}(R) \in \mathbb{R}^M$ aggregates the streams into the sub-layer input,

$$\tilde{x} = \sum_{m=1}^M h_{\text{pre},m}(R) R_m, \quad (23)$$

which instantiates the iHC-Read operator in Eqs. 4 and 8. Let $\Delta \in \mathbb{R}^d$ denote the gated sub-layer update, with $\Delta = g^A \odot a$ for attention (Eq. 7) and $\Delta = g^F \odot f$ for the FFN (Eq. 10). A write vector $h_{\text{post}}(R) \in \mathbb{R}^M$ independently scales and adds the shared update to each residual stream,

$$R'_m = R_m + h_{\text{post},m}(R) \Delta, \quad m = 1, \dots, M, \quad (24)$$

which instantiates the iHC-Write operator. The token-dependent read and write vectors are generated by

$$\begin{bmatrix} h_{\text{pre}}(R) \\ h_{\text{post}}(R) \end{bmatrix} = \begin{bmatrix} \mathbf{1}_M \\ 2\mathbf{1}_M \end{bmatrix} \odot \sigma \left(\alpha_h W_h \text{RMSNorm}(\text{vec}(R)) + \begin{bmatrix} b_{\text{pre}} \\ b_{\text{post}} \end{bmatrix} \right). \quad (25)$$

Here, $W_h \in \mathbb{R}^{2M \times Md}$, $b_{\text{pre}}, b_{\text{post}} \in \mathbb{R}^M$, and α_h is a learnable scalar. $\mathbf{1}_M$ and $\mathbf{0}_M$ denote the all-ones and all-zeros vectors in $\mathbb{R}^M$, respectively, and σ is applied elementwise. This gives $h_{\text{pre}}(R) \in (0, 1)^M$ and $h_{\text{post}}(R) \in (0, 2)^M$. We initialize $b_{\text{pre}} = \text{logit}(1/M)\mathbf{1}_M$ and $b_{\text{post}} = \mathbf{0}_M$, so suppressing the input-dependent term yields $h_{\text{pre}} = \frac{1}{M}\mathbf{1}_M$ and $h_{\text{post}} = \mathbf{1}_M$. Because α_h is initialized to a small value and the streams begin as replicas of the input, each block starts close to a standard residual update. Input-adaptive multi-stream routing then emerges gradually during training. As described in Sec. 3.2, this *read–compute–write* pattern is applied separately around the attention and the FFN sub-layers, with $M = 4$ residual streams in our main model.

The key simplification relative to mHC lies in the residual-stream transition. In the general HC/mHC formulation, the write step additionally mixes the streams themselves through a residual transition matrix $H_{\text{res}} \in \mathbb{R}^{M \times M}$, i.e., $R'_m = \sum_{m'=1}^M (H_{\text{res}})_{mm'} R_{m'} + h_{\text{post},m}(R) \Delta$, where mHC learns a token-dependent H_{res} and projects it onto the doubly-stochastic manifold with Sinkhorn iterations [90]. iHC instead fixes $H_{\text{res}} = I_M$, yielding the write rule in Eq. 24. This removes the per-token $M \times M$ residual mixing matrix and the Sinkhorn projection required by mHC. Beyond maintaining, reading, and writing the M residual streams, the coefficient generator in Eq. 25 requires one RMSNorm, one $Md \rightarrow 2M$ projection, and elementwise gates per sub-layer. For fixed $M = 4$, this routing arithmetic scales linearly with d , compared with the quadratic-in- d projection cost of attention and FFNs. iHC therefore retains adaptive read and write mappings over multiple residual streams while preserving an exact identity residual transition.

4 Scaling Recipe

This section presents the scaling recipe for **Chimera**. Sec. 4.1 develops **HeteroP**, a module-wise hyperparameter-transfer parameterization that yields a consistently tuned model family across width and depth. Section 4.2 then uses this family to fit Chinchilla-style compute-optimal laws over model size and training tokens and extends the analysis to the image-video data ratio.

4.1 Hyperparameter Transfer

Studying the scaling behavior of a new architecture requires training a family of models of increasing scales. This immediately raises a practical question: which hyperparameters should be used at each scale? Re-tuning the learning rate, initialization, and weight decay independently at every scale is prohibitively expensive. Reusing the values tuned at a small scale is cheap but often suboptimal, because the optimal learning rate drifts with model width and depth under standard parameterizations [18, 93]. *Hyperparameter transfer* resolves this tension by reparameterizing the model and

optimizer so that a set of base hyperparameters remains approximately stable across width and depth. One can tune these base values on a small proxy model and then transfer them to larger models

Existing hyperparameter-transfer methods address complementary scaling axes. The maximal-update parameterization (μ P) provides width-transfer rules and has been validated on Transformers and ResNets [93], while CompleteP extends transfer across depth for Transformers [18]. Recent diffusion-specific work further adapts μ P to full-attention diffusion Transformers [103]. These methods provide the principles on which we build, but do not specify a complete parameterization for a heterogeneous visual diffusion backbone that jointly combines KDA linear attention, MLA global attention, modality-aware short convolutions, sparse MoE layers, iHC, sandwich normalization, and diffusion timestep conditioning. We call our resulting method **HeteroP** (Heterogeneous Parameterization).

Why transfer is necessary for a fair scaling study. Beyond reducing tuning cost, hyperparameter transfer makes a scaling study *meaningful*. A scaling curve compares the losses of models at different sizes to estimate how performance improves with compute. If each size is trained with scale-mismatched hyperparameters, differences in measured loss may reflect both model scale and suboptimal tuning, obscuring the true effect of scale. For example, a larger model trained with a suboptimal learning rate can even underperform a smaller one, making the resulting curve unreliable for design conclusions. Hyperparameter transfer reduces this confound effect, making differences in loss primarily reflect scale and architecture rather than tuning noise. Earlier diffusion scaling studies do not explicitly control this factor with a width- and depth-transfer parameterization [50, 63]. In Sec. 5.2, we verify that the optimal learning rate stays stable across width and depth under our recipe, but shifts under standard parameterization.

Why heterogeneous backbones require module-wise ratios. A homogeneous Transformer is often widened by multiplying a common hidden dimension, making a single global width ratio a reasonable shorthand when instantiating transfer rules. In **Chimera**, “width” is instead a collection of distinct dimensions. A projection may take as input the model hidden state, the MLA KV-compression latent, a KDA head, an MoE expert hidden state, the router state, or the timestep-conditioning state. These dimensions may scale with the model width, grow at different rates, or remain fixed. Applying the global model-width ratio to every tensor would therefore mis-scale parameters whose updates are controlled by another fan-in. HeteroP resolves this mismatch by computing the target-to-proxy ratio separately for every parameter group from the dimension that controls its activation and update scale. The result is a fine-grained, dimension-accurate parameterization that preserves the intended transfer rule for structurally heterogeneous backbones.

HeteroP Parameterization. Let $\mathcal{M}^{(0)}$ denote a small proxy model on which the base hyperparameters are tuned, and let $\mathcal{M}$ denote a target model to which they are transferred. The two models share the same architecture but may differ in width and depth, with $n_{\text{blk}}^{(0)}$ and n_{blk} blocks, respectively. For each target parameter group W , let $W^{(0)}$ denote its proxy counterpart. We write the transfer compactly as:

$$\bar{\mathbf{h}} = (\sigma_{\text{base}}^2, \eta_{\text{base}}, \lambda_{\text{base}}, \epsilon_{\text{base}}), \quad (26)$$

$$(m_W, m_L) = \left(\frac{\text{fan-in}(W)}{\text{fan-in}(W^{(0)})}, \frac{n_{\text{blk}}}{n_{\text{blk}}^{(0)}} \right), \quad (27)$$

$$\mathcal{P}_W = \mathcal{T}_{\rho(W)}(\bar{\mathbf{h}}; m_W, m_L). \quad (28)$$

Here, σ_{base}^2 , η_{base} , λ_{base} , and ϵ_{base} denote the base initialization variance, learning rate, weight decay, and AdamW ϵ . $\mathcal{P}_W$ collects the initialization, optimizer, forward, and residual-scaling settings for W , while $\rho(W)$ denotes its functional role. m_W is a module-specific width multiplier determined by the fan-in of W and is independent of the number of blocks, whereas m_L is the global depth multiplier determined by the block count. In tuning **Chimera**, unlike previous works that adopt a single global scale ratio, m_W may be determined by the model hidden width, the MLA KV-compression rank, KDA head width, MoE expert width, router width, or timestep-conditioning width. The depth multiplier m_L is shared because it is determined by the target-to-proxy block-count ratio. Our proxy has model width 512 and depth 4 (22M activated parameters, Table 4), while the largest model has model width 2048 and depth 32. Hence $m_W = 4$ for parameter groups whose fan-in is the model width, while other groups receive their own ratios, and $m_L = 8$. Operationally, HeteroP (i) tunes $\bar{\mathbf{h}}$ once on the proxy, (ii) assigns each target tensor a role $\rho(W)$ and computes m_W , and (iii) applies the role-dependent map $\mathcal{T}_{\rho(W)}$ together with m_L . Eqs. 26–28 define these three ingredients, and Table 1 gives their module-wise instantiation.

Table 1 Module-wise hyperparameter-transfer rules of HeteroP for Chimera, compared with standard parameterization (SP). We jointly control width and depth: m_W is the target-to-proxy fan-in ratio computed separately for parameter group W (e.g., from the model width, MLA KV-compression rank, KDA head width, MoE expert width, or router width), and m_L is the target-to-proxy depth ratio raised to the listed power. $\eta_{\text{base}}, \sigma_{\text{base}}^2, \lambda_{\text{base}}, \epsilon_{\text{base}}$ denote the learning rate, initialization variance, weight decay, and AdamW ϵ tuned once on the proxy model and transferred to all scales.

Module / Hyperparameter	SP	HeteroP
<i>Input adapters</i> (patch / text / timestep embedding)		
Init. Var.	σ_{base}^2	σ_{base}^2
LR (AdamW)	η_{base}	η_{base}
<i>Sandwich RMSNorm</i> (pre-/post-norm gains)		
Init. Var.	σ_{base}^2	σ_{base}^2
LR (AdamW)	η_{base}	η_{base}
<i>Hidden weights</i> (KDA/MLA projections, FFN & MoE experts, router)		
Init. Var.	σ_{base}^2	$\sigma_{\text{base}}^2 \cdot m_W^{-1}$
LR (AdamW)	η_{base}	$\eta_{\text{base}} \cdot m_W^{-1}$
Bias LR (AdamW)	η_{base}	η_{base}
WD (AdamW)	λ_{base}	$\lambda_{\text{base}} \cdot m_W$
Attention residual (KDA, MLA; iHC)	$\mathbf{X}^l + \text{Attn}(\text{RMSNorm}(\mathbf{X}^l))$	$\mathbf{X}^l + m_L^{-1} \cdot \text{Attn}(\text{RMSNorm}(\mathbf{X}^l))$
FFN / MoE residual (iHC)	$\mathbf{Z}^l + \text{FFN}(\text{RMSNorm}(\mathbf{Z}^l))$	$\mathbf{Z}^l + m_L^{-1} \cdot \text{FFN}(\text{RMSNorm}(\mathbf{Z}^l))$
<i>Final norm</i> (pre-readout)		
Init. Var.	σ_{base}^2	σ_{base}^2
LR (AdamW)	η_{base}	η_{base}
<i>Diffusion readout head</i> (velocity prediction)		
Init. Var.	σ_{base}^2	σ_{base}^2
LR (AdamW)	η_{base}	η_{base}
Forward	$\mathbf{X}^L \mathbf{W}_{\text{head}}^\top$	$\mathbf{X}^L \mathbf{W}_{\text{head}}^\top \cdot m_W^{-1}$
AdamW ϵ (residual blocks)	ϵ_{base}	$\epsilon_{\text{base}} \cdot m_W^{-1} \cdot m_L^{-1}$
AdamW ϵ (input adapters & readout)	ϵ_{base}	$\epsilon_{\text{base}} \cdot m_W^{-1}$

Width and depth transfer rules. The HeteroP rules in Table 1 combine module-wise hyperparameter transfer across both width and depth. Under width transfer, hidden matmul weights use initialization variance and learning rate scaled by m_W^{-1} , together with weight decay scaled by m_W . These initialization and learning-rate corrections preserve width-consistent activation and update scales, while the paired scaling $\eta \propto m_W^{-1}$ and $\lambda \propto m_W$ keeps $\eta\lambda$ invariant. Input adapters, normalization gains, biases, and the readout keep their base initialization and learning rate, matching the input/output treatment of μP . The readout is additionally multiplied by m_W^{-1} in the forward pass, and AdamW ϵ is scaled by m_W^{-1} to track the shrinking gradient scale.

HeteroP incorporates the depth correction of CompleteP [18]: each attention or FFN/MoE sub-layer output is scaled by m_L^{-1} before the corresponding iHC write, keeping the aggregate residual contribution on a depth-independent scale. Residual-block tensors receive the same additional m_L^{-1} correction to AdamW ϵ , while their learning rates require no additional depth correction, as listed in Table 1.

Module-wise assignments. Mapping HeteroP onto Chimera requires assigning each learnable tensor to a rule. For each parameter group, we identify its functional role and, where applicable, the dimension that determines its fan-in. The resulting assignment is summarized below.

- **Input adapters and readout.** The patch, text, and timestep embedding projections are input layers, and the velocity-prediction head is the output layer. Both keep the base learning rate and initialization, and the readout additionally uses the m_W^{-1} forward multiplier. The blockwise timestep-conditioning MLP in Eq. 3, however, follows the hidden rule; its input projection is parameterized by the width of e_τ .
- **Hybrid attention.** All projections that read the hidden state, including the KDA query-key-value projection and the MLA query and key-value down-projections, scale with the model width. The MLA key-value up-projection scales with the KV-compression rank, while the KDA gating and the output projections of both operators scale with the corresponding head widths. The short-convolution kernels and the KDA low-rank gate up-projections have a fixed fan-in per output channel. They therefore use $m_W = 1$, retaining the base learning rate and weight decay while using the residual-block ϵ scaling. The complementary gate down-projections read the hidden state, so they follow the hidden rule.

- **FFN and MoE.** The up- and gate-projections of dense SwiGLU FFNs and MoE experts scale with the model width, and the down-projections scale with their respective hidden widths. The router input projection likewise follows the model width, while the router hidden width is held fixed across scales. This keeps the routing logits on a consistent scale as the model grows. The routing sparsity is also fixed, as discussed below.
- **iHC and Sandwich Normalization.** The iHC gains and biases, together with the RMSNorm gains, are control tensors and follow the norm treatment: base initialization, base learning rate, and the hidden ϵ . The lightweight projection that produces the iHC read/write vectors reads the concatenated residual streams, whose size grows with the model width, so it follows the hidden rule. This preserves residual stability across width and depth.

What is held fixed when scaling. When constructing the model family, we vary the module-specific widths and block count represented by m_W and m_L , respectively, while holding the structural settings listed below fixed. The MoE routing configuration is constant across scales: every model uses 56 routed experts with 8 activated per token, an expert activation ratio of $1/7$ (Sec. 3.5). The MLA KV-compression ratio, defined as the compression rank divided by the model width, is likewise constant as width grows, so compression acts as a fixed architectural property rather than a hidden scaling variable.³ The same holds for the KDA-to-MLA layer ratio, the iHC residual-stream count, and the patchification factors. Fixing these settings prevents structural choices from being entangled with the scaling study and keeps all data points mutually comparable.

4.2 Chinchilla Laws for Chimera

HeteroP yields mutually comparable **Chimera** models with scale-appropriate hyperparameters. Using this family, we fit Chinchilla-style compute-optimal scaling laws [36] for visual diffusion.

Why a compute-optimal law is needed. A central practical question in scaling is allocation. Given a fixed training budget of C FLOPs, how large should the model N be, and on how many tokens D should it process? Training an overly large model for too few steps wastes capacity, whereas training an overly small model for too long spends compute on a saturated model. Hoffmann et al. [36] show that, for language models, the loss-minimizing split between N and D follows a predictable trade-off, and that ignoring it yields substantially undertrained models. For visual diffusion, this trade-off has not been characterized on a model family with hyperparameters controlled across scales through HeteroP (Sec. 4.1). Our goal is therefore to make the size-versus-data trade-off of **Chimera** explicit and predictive, so that the configuration of a large target run can be chosen *a priori* from the fitted law rather than by trial and error.

Formulation. We adopt the parametric form of Hoffmann et al. [36], modeling the training loss as a function of the model size N and the amount of training data D :

$$\hat{\mathcal{L}}(N, D) = E + \frac{A}{N^a} + \frac{B}{D^b}, \quad (29)$$

where E denotes the irreducible loss under the fixed data distribution and diffusion objective, *i.e.*, the Bayes-optimal diffusion loss on the data distribution, and the two power-law terms capture the finite-size penalties from limited capacity and limited data. Three measurement choices adapt this form to our setting. First, the loss $\hat{\mathcal{L}}$ is the **Chimera diffusion loss**, the rectified-flow v -prediction objective of Sec. 3.1, rather than a downstream generation score such as FID. Under the fixed objective and evaluation protocol used here, diffusion loss provides a sampler-independent metric for comparing pre-training across model scales. Second, because **Chimera** is sparsely activated, N counts *activated* parameters, so the law reflects the compute actually spent per token rather than the total parameter store.⁴ Third, D denotes the cumulative training-data volume, measured by the number of visual latent positions processed, and we relate it to compute through the standard accounting $C \approx 6ND$ for the dominant matmul cost,⁵ which lets us express the law and its optimum directly in terms of compute.

³Whether the optimal compression ratio is itself scale-dependent is an interesting question that we leave to future work. We use the proxy value.

⁴The load balance across experts does not affect computational cost, but it can affect model quality and hence the losses used for fitting. We therefore conduct this study under near-perfect load balance, as demonstrated in Fig. 5.

⁵This approximation remains valid for the hybrid backbone. Most layers use linear attention with constant per-token cost, and the few MLA layers attend over compressed key-value states, so parameter matmuls dominate the overall compute.

Fitting and usage. All runs share the same base AdamW configuration and a warmup-then-constant learning-rate schedule without decay. Thus, the loss measured at an intermediate step is not affected by a learning-rate decay phase tailored to that training duration. A single run can therefore provide comparable loss measurements across multiple data budgets D . We further ensure that all runs remain in the infinite-data regime: no run processes more than one epoch of the training corpus. Given a set of runs spanning model sizes and durations, we estimate the compute-optimal allocation with the three estimators of Hoffmann et al. [36]: the lower envelope of the loss-versus-compute curves, IsoFLOP profiles that compare losses across model sizes at fixed compute budgets, and a direct parametric fit of Eq. 29 that minimizes the Huber loss between predicted and observed log-losses over a grid of initializations. The fitted law yields the compute-optimal frontier $N_{\text{opt}}(C)$ and $D_{\text{opt}}(C)$ in closed form. We use this frontier to select the configuration of **Chimera**’s largest runs, and report the fits and their cross-estimator agreement in Sec. 5.3.

Extending the law to the image–video data ratio. Unlike scaling laws for single-modality text data, visual generation trains on multimodal data that mixes images and videos. The two differ in information content and in the number of tokens each sample contributes, so the mixture shifts both the learning signal and how the compute budget is spent. The standard law over (N, D) leaves this axis uncontrolled. We therefore extend the scaling law with an additional dimension, the image–video data ratio, and model how its optimal value changes as the training budget increases.

Specifically, we generalize the IsoFLOP methodology of Hoffmann et al. [36]. We train models of multiple sizes under multiple image–video ratios and record their diffusion losses at fixed compute budgets, tracking the image and video losses separately. At each budget, instead of fitting a one-dimensional quadratic profile over model size alone, we fit a quadratic surface over model size and data ratio for each loss. We then select the optimum of every fitted surface and fit the compute-optimal frontier to these per-budget optima, obtaining the optimal model size and image–video ratio as functions of compute. Sec. 5.4 presents the fitted surfaces and the resulting frontier.

5 Experiments

We evaluate **Chimera** along the two axes developed in this paper. We first validate the scaling methodology: Sec. 5.2 verifies HeteroP transfer across width and depth, Sec. 5.3 presents compute-optimal scaling laws for image and video generation, and Sec. 5.4 extends the law to the image-video data mixture. We then evaluate the models: Sec. 5.5 compares **Chimera** with strong baselines on generation quality, and Sec. 5.6 presents component ablations.

5.1 Implementation details.

All models instantiate the architecture of Sec. 3 and differ only in width, FFN width, and depth. Fig. 6 visualizes representative configurations spanning 55M to 4B activated parameters, of which the exact specifications are listed in Appendix B. As specified in Sec. 4.1, all structural ratios are held fixed across scales, including the MoE configuration, the KDA-to-MLA ratio, the residual-stream count, the MLA KV-compression ratio, and the patchification factors.

Frozen components. Visual inputs are encoded by a frozen causal 3D VAE [82] with 16 latent channels and $(4, 8, 8)$ compression along the temporal, height, and width axes. A clip of $4k + 1$ frames at resolution $H \times W$ is encoded into $k + 1$ latent frames of size $\frac{H}{8} \times \frac{W}{8}$, and a single image is the special case $k = 0$. Encoded latents are standardized with the per-channel mean and standard deviation of the VAE latent space, so the rectified-flow objective operates on approximately zero-mean, unit-variance targets. We then apply $(1, 2, 2)$ patchification. Text prompts are encoded by a frozen umT5-XXL encoder [16] with a maximum length of 512 tokens. Padding positions are dropped and the remaining embeddings are packed into a variable-length sequence that is prepended to the visual tokens through a separate trainable projection (Sec. 3.2).

Training recipe. All models are trained with the rectified-flow objective of Sec. 3.1, and the loss is the mean squared error on the velocity target, unweighted across timesteps. One timestep is drawn per sample and shared by all of its tokens, from a logit-normal distribution [24] that concentrates supervision on intermediate noise levels. Since a fixed timestep destroys more information in a longer token sequence, we then shift it toward higher noise by $\tau' = \tau / (\tau + s(1 - \tau))$, with $s = \sqrt{L_{\text{vis}}/256}$ for a sample of L_{vis} visual tokens, taking the 256^2 image bucket as the reference. Because s is computed per sample, images and video clips of different sizes are shifted independently within the same batch.

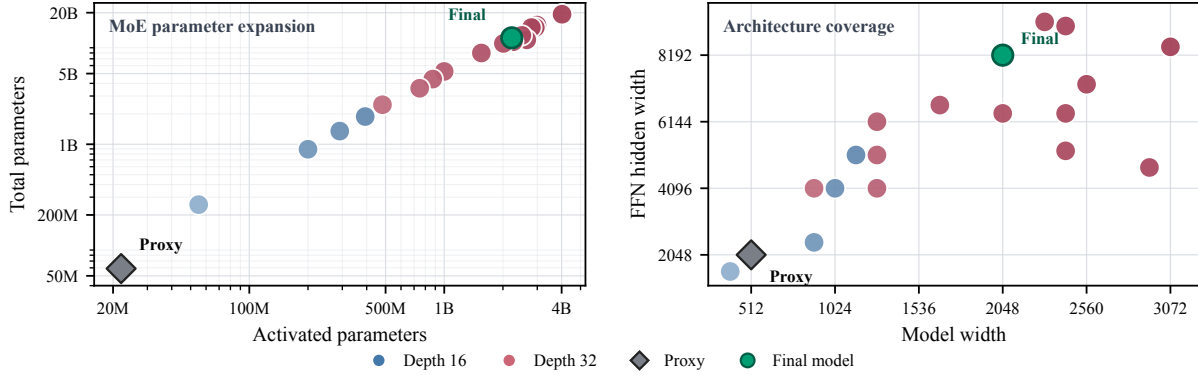


Figure 6 Representative Chimera model family. Left: total versus activated parameters, showing the parameter expansion. Right: model width versus FFN hidden width, showing the configurations covered by the scaling study. The family is not a strict fixed-aspect-ratio scaling ladder; instead, it spans feasible combinations of width, depth, and FFN capacity while keeping the high-level architecture fixed. We primarily fix the depth and study the scaling of width. All markers use the same diameter; hue denotes depth, while within-depth color intensity increases logarithmically with activated parameter count. The proxy and final training model are highlighted separately. Exact configurations are provided in Appendix B.

Text conditioning is dropped with probability 0.1 to enable classifier-free guidance at inference. We optimize with AdamW using a base learning rate $\eta_{\text{base}} = 10^{-3}$, betas (0.95, 0.95) [9, 27, 62], $\epsilon_{\text{base}} = 10^{-8}$, and a base weight decay $\lambda_{\text{base}} = 0.01$. HeteroP instantiates the parameter-group learning rates, initialization variances, weight decays, and AdamW ϵ values at each scale according to Table 1. The base learning rate warms up linearly from 10^{-6} over 2,000 steps and then remains constant. The constant schedule keeps intermediate checkpoints comparable across training durations, which the scaling study in Sec. 5.3 relies on.

Gradients are clipped to a global norm of 1.0. As a stability guard, we skip the optimizer and EMA update at any step whose pre-clipping gradient norm is non-finite or exceeds 10. Expert load is balanced by the auxiliary-loss-free bias update of Sec. 3.5, with the selection bias adjusted by 10^{-3} per step. No load-balancing term is therefore added to the training objective, and the reported loss is exactly the flow-matching loss above, which matters for the scaling study in Sec. 5.3 where this loss is the primary metric. We additionally maintain an exponential moving average of the trainable parameters with decay 0.9999, updated every step.

Data. We train on a large-scale curated visual corpus. Training uses multi-resolution pixel buckets: an early stage mixes 256^2 , 512^2 , and 1024^2 buckets with sampling ratios 4:2:2. Mixed-resolution batches yield highly variable sequence lengths, so a data sequence balancer [99] redistributes packed sequences across devices to balance the per-GPU load. Samples with aspect ratio above 4:1 or below 1:4 are discarded. During mixed image–video training, we retain the three image buckets and add approximately 5-second video clips at 180p and 360p, using loader sampling ratios 2:1:1:2:2 for 256^2 , 512^2 , 1024^2 , 180p video, and 360p video, respectively. Each clip contains 81 frames sampled at 16 fps and is assigned to an area-preserving crop bucket according to its aspect ratio. Additionally, we conduct all trials in Secs. 5.2 and 5.3 using a deterministic data loader and under an infinite-training-data regime.

Infrastructure. We train Chimera with Hybrid Sharded Data Parallel (HSDP) [101], sharding model states across the eight GPUs within each node while replicating them across nodes. HSDP is applied at block granularity to overlap parameter communication with computation. Parameters are gathered in bf16, gradients are reduced in fp32. We further apply activation checkpointing to every block. The dominant per-layer operators are implemented directly over the packed token representation using custom Triton kernels. These include a fused multimodal short convolution, chunked KDA with fused gating and normalization, and an MoE implementation based on expert-wise token sorting and grouped matrix multiplications.

Mixed image–video training can produce substantial load imbalance because different ranks may process samples with widely varying computational costs. We therefore introduce two lightweight workload balancers. For VAE encoding, ranks within each node exchange sample-shape metadata and estimate encoding cost from pixel count. Samples are assigned greedily in descending order of cost to the currently least-loaded rank, transferred in a single all-to-all, and encoded latents are returned to their original ranks and order through a second all-to-all. We additionally apply a DiT balancer before the first transformer block, following [99]. It estimates the cost of each packed sequence using a model

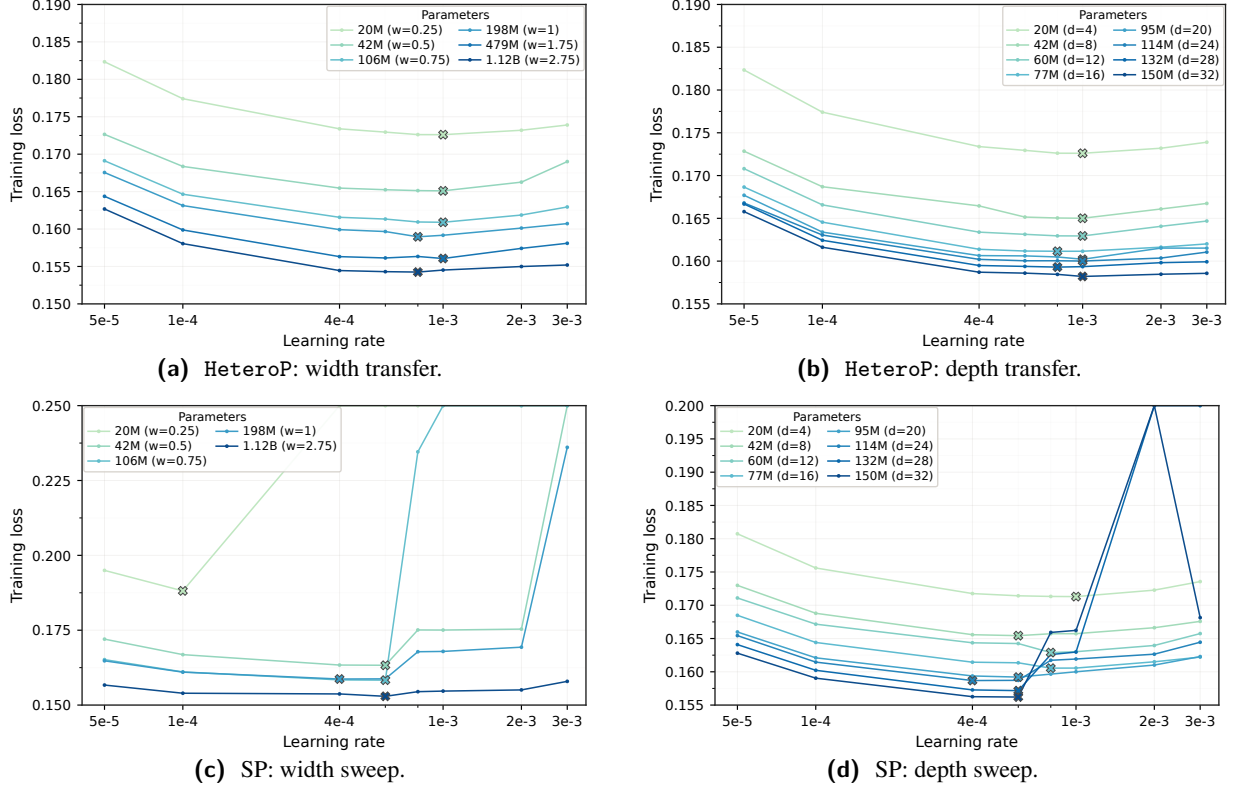


Figure 7 HeteroP transfers learning rates across width and depth. Top row: HeteroP; bottom row: standard parameterization (SP). Panels (a,c) vary width $0.25\times$ to $2.75\times$ (20M to 1.12B activated parameters) and panels (b,d) vary depth from 4 to 32 layers at fixed width (20M to 150M activated parameters). The crosses mark the minimum after 50k steps. The horizontal axis denotes the base learning rate η_{base} under our recipe and the unscaled global learning rate under SP. With HeteroP, the optimal learning rate stays at $\approx 10^{-3}$ across $56\times$ in model size and $8\times$ in depth. Under SP, it shifts with scale, and high-rate runs can become unstable.

that captures both the linear cost of KDA-dominated computation and the quadratic cost of periodic MLA layers, and redistributes complete sequences through a differentiable all-to-all so that outputs and gradients are routed back to their owners. Both assignments are recomputed at every step and applied only when the estimated reduction in the maximum per-rank workload exceeds a small threshold.

Evaluation protocol. We evaluate **Chimera** at two levels. For the scaling study, the primary metric is the **Chimera** diffusion training loss of Sec. 3.1. Under a fixed objective and noise parameterization, this loss is sampler-independent and therefore comparable across model sizes, training durations, and data mixtures, which the analyses in Sec. 5.3 rely on. For the final models, we additionally report downstream generation quality. We assess image generation on the GenEval [28] and DPG-Bench [40] benchmarks, and video generation with FID and FVD under the length-extrapolation protocol of Sec. 5.5. To keep comparisons controlled, we hold the sampling configuration fixed across the variants under test within each such comparison, so that measured differences reflect the models rather than the sampling settings. We detail the setup specific to each experiment in the corresponding subsection.

5.2 HeteroP Hyperparameter Transfer

We first verify the central premise of the scaling study: under HeteroP, the base learning rate η_{base} tuned on the proxy model remains near-optimal at larger scales.

Setup. We sweep the base learning rate η_{base} from 5×10^{-5} to 3×10^{-3} on two model families that vary one scaling axis at a time. Under our recipe, every model is trained for 50k steps at each candidate η_{base} , with the parameter-group learning rates and other scaled optimizer values instantiated according to Table 1. For comparison, we repeat the same sweeps under standard parameterization (SP), where the swept learning rate is applied globally without HeteroP’s transfer rules. The width sweep spans 20M to 1.12B activated parameters. The depth sweep varies the number of layers

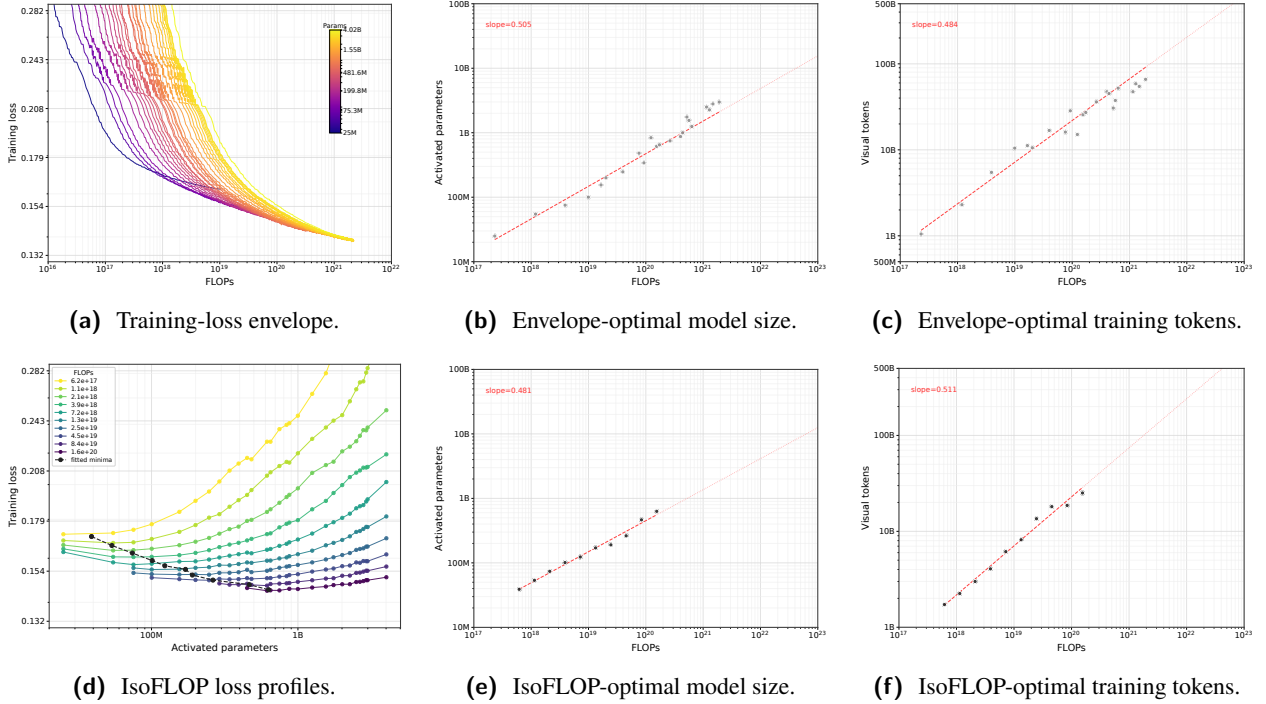


Figure 8 Compute-optimal scaling of Chimera for 256^2 image pre-training. **Top row, training-loss envelope:** (a) Diffusion training loss versus cumulative training FLOPs; the lower envelope identifies the best checkpoint at each compute budget. (b) Compute-optimal activated model size $N_{\text{opt}}(C)$ from the envelope-winning checkpoints. (c) The corresponding compute-optimal training tokens $D_{\text{opt}}(C)$, obtained using each model’s logged per-step FLOPs. **Bottom row, IsoFLOP profiles:** (d) Diffusion loss versus activated model size at fixed compute budgets; the marked profile minima identify the optimal model at each budget. (e) Compute-optimal activated model size inferred from these minima. (f) The corresponding compute-optimal training tokens. Dashed lines show the fitted power laws, with their exponents annotated in the scaling panels.

from 4 to 32, spanning 20M to 150M activated parameters. Across both sweeps, all other architectural and training settings are held fixed as specified in Sec. 5.1.

Results. Fig. 7 plots the training loss after 50k steps against the base learning rate η_{base} for both families. Two observations support the HeteroP transfer recipe. First, the optimal base learning rate is approximately stable across scales: every configuration attains its minimum near $\eta_{\text{base}} = 10^{-3}$, drifting by at most one point on the sweep grid across a $56\times$ range of model sizes (Fig. 7a) and an $8\times$ range of depths (Fig. 7b), and the loss basin around the optimum remains wide and flat at all scales. Second, the curves are cleanly ordered: at every learning rate, wider and deeper models achieve strictly lower loss with no crossovers, indicating that the parameterization preserves the benefit of scale rather than trading it for stability. Together, these results support transferring the base learning rate tuned on the proxy model across both axes, and we therefore fix $\eta_{\text{base}} = 10^{-3}$ for every run in the scaling study of Sec. 5.3. In contrast, under SP, the optimal global learning rate shifts with scale and several high-rate runs become unstable (Figs. 7c and 7d), leading to suboptimal training and less reliable measurements for fitting the scaling law (discussed later in Sec. 5.6).

5.3 Compute-Optimal Scaling Laws

We estimate the compute-optimal allocation between parameters and training tokens with all three of their estimators: the training-curve envelope, IsoFLOP profiles, and a direct parametric fit of the loss surface (Eq. 29). In all cases, we start by training the model family in Table 4 with different numbers of tokens, and ensure all models are trained in the infinite-data regime. All fits use activated parameters for N , since this is the quantity that determines the per-token compute of our sparse MoE backbone, and use the diffusion training loss defined in Sec. 3.1. The warmup-then-constant base-learning-rate schedule from Sec. 5.1 lets us treat intermediate checkpoints from a long run as comparable shorter-horizon runs.

Specifically, the envelope estimate follows the lower envelope of loss–FLOPs trajectories and records the checkpoint that

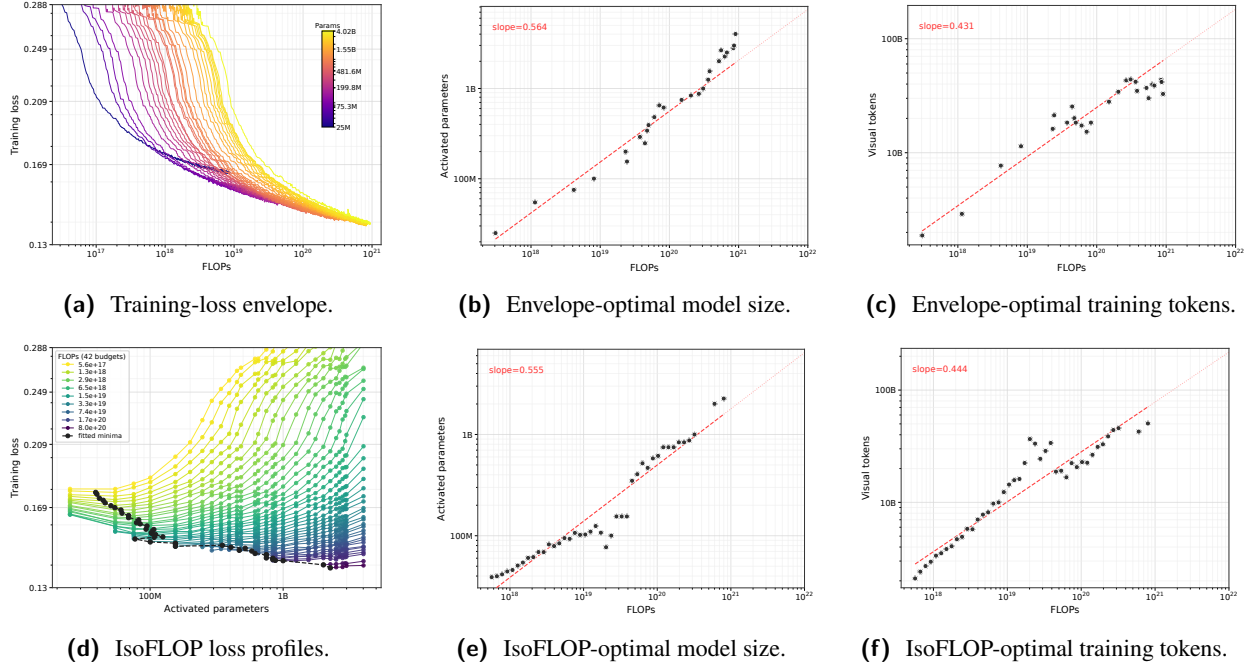


Figure 9 Compute-optimal scaling of *Chimera* for 180p video pre-training. **Top row, training-loss envelope:** (a) Diffusion training loss versus cumulative training FLOPs; the lower envelope identifies the best checkpoint at each compute budget. (b) Compute-optimal activated model size $N_{\text{opt}}(C)$ from the envelope-winning checkpoints. (c) The corresponding compute-optimal training tokens $D_{\text{opt}}(C)$, obtained using each model’s logged per-step FLOPs. **Bottom row, IsoFLOP profiles:** (d) Diffusion loss versus activated model size at fixed compute budgets; quadratic-profile minima, or observed minima for boundary slices, identify the optimal model at each budget. (e) Compute-optimal activated model size inferred from these minima. (f) The corresponding compute-optimal training tokens. Dashed lines show the fitted power laws, with their exponents annotated in the scaling panels.

attains the lowest loss at each compute budget. Although cumulative FLOPs grow with D for a fixed model, the same FLOP budget corresponds to different visual-token counts across model sizes because the per-token cost varies with N . Each envelope-winning checkpoint therefore jointly determines $N_{\text{opt}}(C)$ and $D_{\text{opt}}(C)$. The latter is the optimal training volume associated with the selected model size, rather than an independent compute axis. We use the logged per-step FLOPs to map each checkpoint to C . The IsoFLOP estimate instead slices the trajectories at fixed FLOPs, fits a local quadratic profile of loss against $\log N$, and uses the profile minimum to infer $N_{\text{opt}}(C)$ and $D_{\text{opt}}(C)$. We then fit power laws to these optima. This mirrors the first two empirical estimators in Hoffmann et al. [36], while adapting them to visual diffusion loss and activated-parameter accounting.

Scaling Laws on Image generation. For 256^2 image pre-training, the current fit uses 36 usable runs covering 29 activated model sizes and 1.2×10^5 logged loss points. The envelope spans 8.6×10^{16} to 2.2×10^{21} FLOPs, while the IsoFLOP profiles use interior budgets from 6.2×10^{17} to 1.55×10^{20} FLOPs. Fig. 8 (a-c) shows that the training curves are cleanly ordered and that the lower envelope moves from smaller to larger models as the budget increases. The resulting frontier is close to balanced: the envelope fit gives $N_{\text{opt}}(C) \propto C^{0.505}$ and $D_{\text{opt}}(C) \propto C^{0.484}$. The exponents sum to $0.989 \approx 1$, consistent with the approximate compute relation $C \propto ND$. The compute is divided nearly evenly between activated model size and visual-token count.

Fig. 8 (d-f) gives an independent view of the same law. Each fixed-compute slice forms a smooth, approximately convex profile in model size, with a minimum that shifts rightward as compute increases. Fitting these minima yields $N_{\text{opt}}(C) \propto C^{0.481}$ and $D_{\text{opt}}(C) \propto C^{0.511}$. The agreement between the envelope and IsoFLOP estimates indicates that additional image-training compute should be allocated nearly evenly between activated parameters and visual tokens, rather than spent mostly on either larger models or longer training alone.

Scaling Laws on Video generation. We repeat the study on 180p videos using 30 independent training-from-scratch runs on the same 29-model activated-parameter grid. Video curves cover 1.8×10^{17} to 9.5×10^{20} FLOPs on the envelope, and the IsoFLOP analysis uses budgets from 5.6×10^{17} to 3.2×10^{20} FLOPs, with the high-compute frontier

anchored by the 6×10^{20} and 8×10^{20} FLOP slices.

The video frontier is again close to balanced, but the largest budgets now available tilt mildly toward larger activated models. The full envelope fit in Fig. 9 (a-c) gives $N_{\text{opt}}(C) \propto C^{0.564}$ and $D_{\text{opt}}(C) \propto C^{0.431}$. The IsoFLOP profiles in Fig. 9 (d-f) give $N_{\text{opt}}(C) \propto C^{0.555}$ and $D_{\text{opt}}(C) \propto C^{0.444}$ when the high-compute slices are included.

Compared with image generation, video generation is consistently more model-heavy across both estimators: the envelope/IsoFLOP model-size exponents increase from 0.505/0.481 for images to 0.564/0.555 for videos, while the corresponding token exponents decrease from 0.484/0.511 to 0.431/0.444. Thus, as the compute budget grows, compute-optimal video training allocates a modestly larger share of additional compute to model capacity rather than training tokens. One plausible explanation for this shift is that video generation places a broader representational burden on the backbone. In addition to the appearance and semantic structure required for images, a video model must capture frame correspondence, including object dynamics, temporal correspondences, and camera motion. This may reduce the marginal return from scaling D relative to scaling N , consistent with the observed shift of the video frontier.

Parametric loss fit. The envelope and IsoFLOP estimators use only the per-budget optima. As a third, independent route, we fit the full loss surface of Eq. 29 to all measurements at once, following the third approach of Hoffmann et al. [36]: the five parameters $\{E, A, B, a, b\}$ are obtained by minimizing a Huber loss ($\delta = 10^{-3}$) on the residuals between predicted and observed log-losses, using L-BFGS-B from a grid of 4,500 initializations (doubled for video with a second grid centered on the observed loss scale). For images, the fit uses the de-spiked training trajectories of the same 36 runs, sampling 24 geometrically spaced checkpoints per model (696 points spanning 3.2×10^{17} to 2.2×10^{21} FLOPs); for video, it uses the final checkpoint of each of the 30 runs. The fitted surfaces are

$$\hat{L}_{\text{image}}(N, D) = 0.126 + 5.28 N^{-0.315} + 33.8 D^{-0.336}, \quad (30)$$

$$\hat{L}_{\text{video}}(N, D) = 0.124 + 8.07 N^{-0.330} + 145.2 D^{-0.394}. \quad (31)$$

Both fits are accurate: the image surface reproduces the observed log-losses with $R^2 = 0.993$ and a median absolute relative error of 0.19% (0.22% under leave-one-model-out validation), and the video surface predicts the held-out terminal losses with a median error of 0.49%.

Minimizing the fitted surfaces under the constraint $C \approx 6ND$ yields $N_{\text{opt}}(C) \propto C^{0.516}$, $D_{\text{opt}}(C) \propto C^{0.484}$ for images and $N_{\text{opt}}(C) \propto C^{0.544}$, $D_{\text{opt}}(C) \propto C^{0.456}$ for video. These closed-form exponents agree closely with the two non-parametric estimators above: all three routes place the image frontier within $[0.48, 0.52]$ and the video frontier within $[0.53, 0.56]$ for the model-size exponent. A run-level bootstrap (100 resamples over 80% of the model settings) gives an 80% interval of $[0.503, 0.546]$ for the image exponent; the corresponding video interval is wider ($[0.47, 0.80]$), since the terminal checkpoints span only 0.7 decades in D , so we read the video fit as confirming the direction of the allocation rather than sharpening its exponent. Notably, the fitted irreducible losses are nearly identical across the two modalities ($E = 0.126$ versus 0.124). The near equality of E is consistent with using the same models, including the denoiser and VAE, for both modalities, as architecture changes tend primarily to shift the offset of a scaling curve [31]. Overall, the three estimators consistently indicate that image pre-training compute should be split almost evenly between activated parameters and visual tokens, while video pre-training currently favors allocating a modestly larger share to model size.

5.4 Optimal Image-Video Data Ratio

We extend the compute-optimal analysis along the mixture axis introduced in Sec. 4.2 and estimate how the optimal image–video data ratio evolves with the training budget.

Setup. Unlike the single-modality studies above, these runs train on image and video data jointly. We train models spanning roughly 50M to 2B activated parameters under several image:video mixture ratios between 1:1 and 5:1, following the recipe of Sec. 5.1, and record the image and video diffusion losses separately. Note that this ratio searching range is decided by the actual amount of image and video data, considering two facts: i) Video data is usually less than image data; ii) The selected ratio should be ideally maintain the infinite-data regime (less video data reuse). In detail, we fix the total batch size during training and balance the number of image and video samples. We follow the previous 256² and 180p settings for image and video data, respectively. The analysis covers eight compute budgets that span 10^{18} to 10^{19} FLOPs. Slicing all training trajectories in a given budget yields 39–45 measurements per budget, each

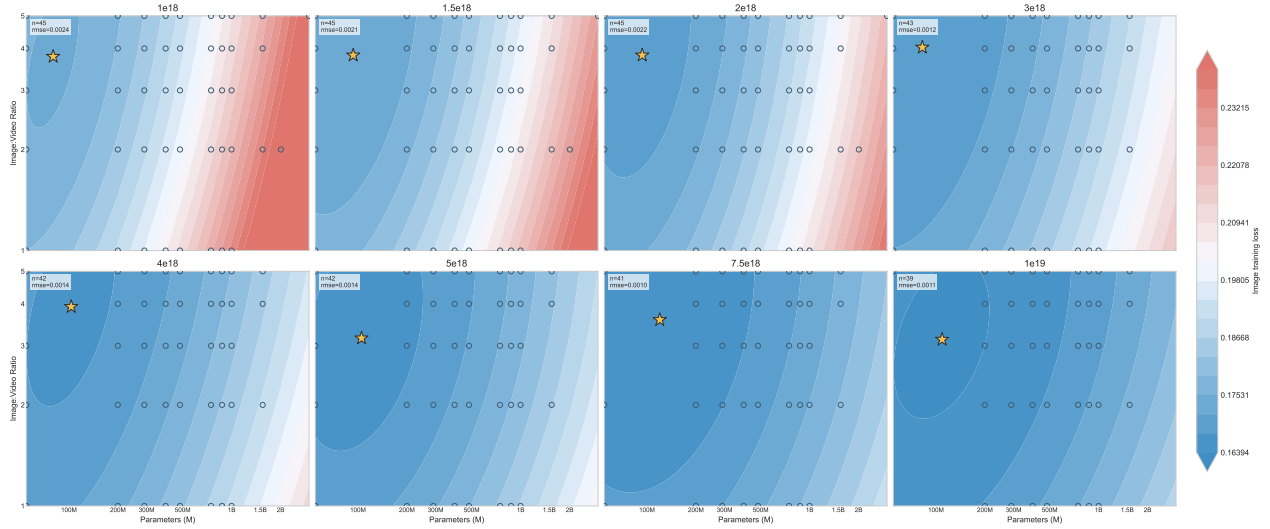


Figure 10 IsoFLOP surfaces of the image diffusion loss over model size and image:video data sample ratio. For each of eight compute budgets from 10^{18} to 10^{19} FLOPs, we fit a quadratic surface to the losses of runs spanning both model sizes and mixture ratios. Each circle marks an experimental data point, and the star marks the fitted optimum under that budget.

corresponding to one model-size and ratio configuration. Following the extended IsoFLOP methodology of Sec. 4.2, we fit a quadratic surface of each loss over model size and mixture ratio at every budget, and take the surface optimum as the compute-optimal configuration for that budget.

IsoFLOP loss surfaces. Fig. 10 shows the fitted image-loss surfaces. The quadratic surfaces reproduce the measurements closely at all budgets, with root-mean-square residuals of at most 2.4×10^{-3} , about 1% of the measured loss. The surfaces are smooth and approximately convex, and their optima (stars) lie in the interior of the ratio range. We notice two consistent patterns. First, the optimal model size shifts toward larger models as the budget grows, in line with the compute-optimal frontier of Sec. 5.3. Second, near the optimum the loss varies far more strongly along the model-size axis than along the ratio axis. Choosing the model size correctly therefore matters most, while moderately off-optimal mixtures cost little.

Compute-optimal data ratio. Fig. 11 tracks the optimal image-to-video ratio at each compute budget for both image and video losses, together with power-law fits. The two losses lead to clearly different optima. For image loss, the optimum is consistently image-heavy across all examined budgets, lying between 3:1 and 4:1. It shifts only gradually toward video as compute increases, moving from roughly 4:1 at 10^{18} FLOPs to roughly 3:1 at 10^{19} FLOPs. The fitted exponent is -0.08 , meaning that the optimal ratio decreases by about 17% per decade of compute. For video loss, the fitted optimum remains at 1:1, the video-heavy boundary of our search range, for all budgets, with a fitted exponent of 0.00. Because this optimum lies on the boundary, we interpret it as a one-sided result: within the mixtures explored here, video loss improves monotonically as the video share increases.

These results suggest a simple principle for data mixing: the compute-optimal mixture is image-heavy, but shifts slowly toward video as the training budget grows. This behavior is intuitive given the different roles and costs of image and video data. Images provide appearance and semantic supervision at a much lower token cost per sample, making image-heavy mixtures more compute-efficient in the low-compute regime. As compute increases, the optimal

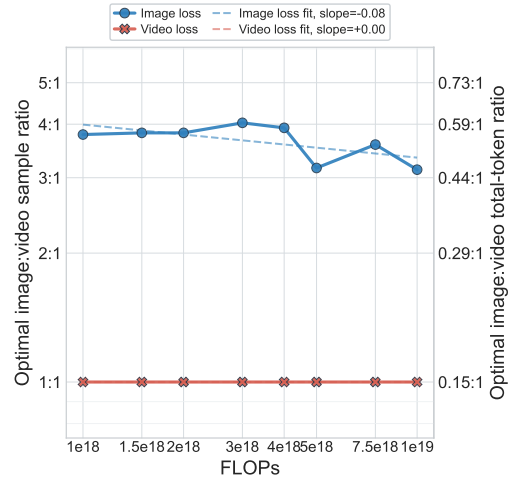


Figure 11 Compute-optimal image:video ratio versus training compute. The image-loss optimum drifts slowly from about 4:1 toward 3:1 (exponent -0.08), while the video-loss optimum stays at 1:1, the most video-heavy mixture in the fitted range.

mixture allocates a modestly larger share to video, suggesting that additional budget can be used to learn temporally rich and long-tail dynamics after the model has acquired sufficient appearance priors from images. The small negative slope further indicates that this transition is gradual rather than abrupt: video becomes increasingly valuable with scale, but does not eliminate the token-efficiency advantage of image data.

We also convert the sample ratio into a total-token ratio, including text tokens. Under this view, the same optima correspond to a much more video-heavy allocation in terms of training tokens, reflecting the substantially higher token cost of each video sample. Together with the size–data frontier in Sec. 5.3, these fits specify the compute-optimal model size, token budget, and data mixture for our largest runs.

5.5 Generation Quality and Comparisons

Loss comparisons under matched compute. We construct 2B instantiations of the Wan2.1 [82] and Z-Image [8] architectures and train them alongside two 2B **Chimera** variants on the same data, with the same diffusion objective, and to a shared compute budget of 5×10^{20} FLOPs. The first **Chimera** variant, Chimera-dense, uses dense FFNs and standard parameterization, providing a matched comparison without sparse capacity or HeteroP. The complete variant adds MoE, iHC, and HeteroP.

Fig. 12 shows a clear convergence advantage for **Chimera**. At the common training loss of 0.149, Wan requires 4.29×10^{20} FLOPs and Z-Image requires 3.75×10^{20} FLOPs. Chimera-dense reaches the same loss with 2.55×10^{20} FLOPs, corresponding to $1.7\times$ higher compute efficiency than Wan despite using neither MoE nor HeteroP. The complete **Chimera** configuration reaches the target with only 6.27×10^{19} FLOPs, a $6.8\times$ compute-efficiency gain over Wan. Its curve remains below all dense baselines throughout the shared compute range. These results separate two sources of improvement: the dense **Chimera** architecture already converges more efficiently than matched full-attention baselines, while sparse capacity, iHC, and HeteroP compound this advantage.

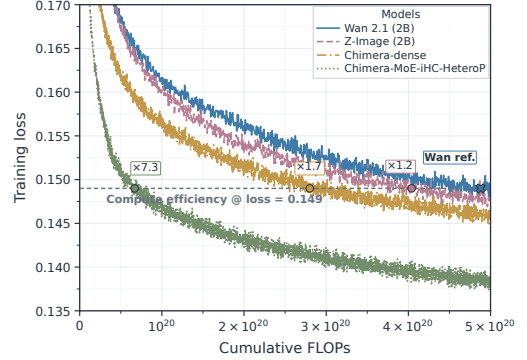


Figure 12 Compute efficiency under matched training conditions. We calculate the compute efficiency across 2B instantiations of Wan2.1, Z-Image, Chimera-dense, and the complete Chimera configuration trained with the deterministic dataloader and a compute budget of 5×10^{20} FLOPs.

Qualitative results. Fig. 13 presents text-to-image samples generated by **Chimera** at a resolution of 1344×864 , using prompts ranging from short phrases to detailed descriptions. The samples cover diverse visual styles, including photorealistic macro and portrait photography, studio-style food photography, watercolor and ink illustration, and anime-inspired artwork. Across these examples, **Chimera** captures fine-grained structures such as petal and stamen geometry, skin texture, and reflections on liquid and glass surfaces, while maintaining coherent lighting, depth of field, object boundaries, and artistic style. The last row further examines typography-aware generation, with each example combining a mythical chimera with the word “Chimera.” The word is rendered using materials specified by the prompts, including glowing bubbles, drizzled sauce, and marker strokes, while remaining legible and visually consistent with the geometry and illumination of each scene. Overall, these examples suggest that **Chimera** can produce high-resolution images with strong local detail, global coherence, and text–image alignment despite relying predominantly on linear attention. Despite being trained exclusively on 1K images, **Chimera** can directly generate coherent 2K and 4K images in a zero-shot manner, without resolution-specific finetuning.

Fig. 15 presents qualitative results for text-to-video generation, showing five uniformly sampled frames from each 81-frame (5-second) video. Across diverse scenarios involving object emergence, natural and articulated motion, and camera movement, **Chimera** produces temporally consistent content while preserving object identity, scene composition, and lighting. The examples containing the words “Relax” and “Chimera” further show that rendered text remains legible and geometrically consistent across frames. These results suggest that **Chimera** maintains strong semantic and visual consistency over time.

Zero-shot length extrapolation. To test whether KDA and our modality-aware causal convolution provide a natural mechanism for length extrapolation, we evaluate a **Chimera** checkpoint trained with 5-second video clips at inference

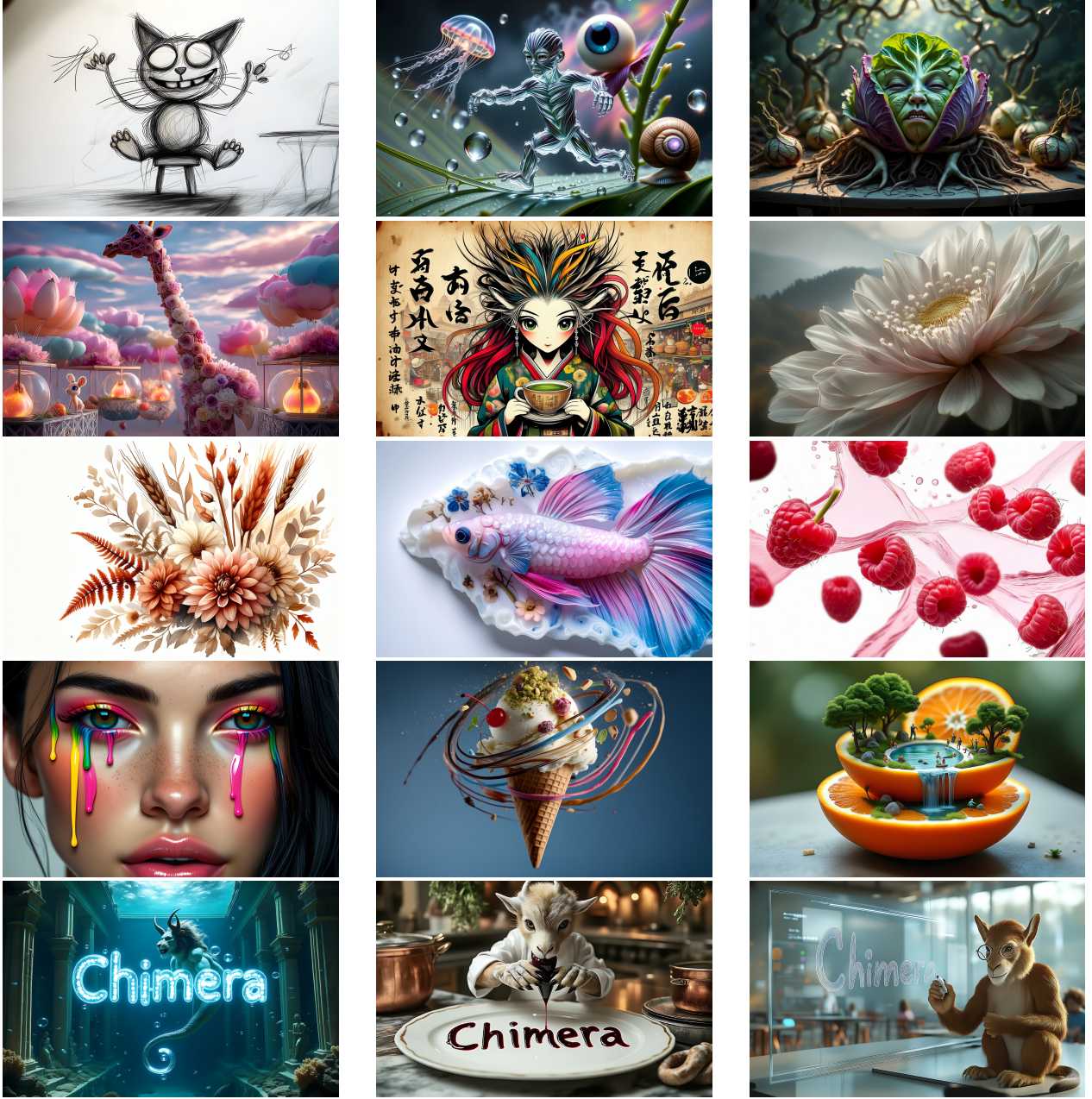


Figure 13 Qualitative text-to-image samples from **Chimera.** All images are generated at 1344×864 resolution with classifier-free guidance scale 4, using prompts that range from short phrases to long, detailed descriptions.

lengths of 5, 10, 15, 20, 25, and 30 seconds, without length-specific finetuning or positional interpolation. We compare against Wan2.1-T2V-1.3B [82] and HunyuanVideo-1.5 [80]. For every model and duration, we generate 512 videos and evaluate them against 512 reference videos sampled from the validation set. To keep the evaluation window fixed while measuring quality progressively farther beyond the training horizon, we compute both metrics only on the final 5 seconds of each video. This yields 512×81 generated frames for FID and 512 generated tail clips for FVD. We use the same prompts, seeds, resolution, frame rate, and duration grid across all models, although the shared 448×288 and 16-fps setting lies outside the baselines' native configurations.

Fig. 16 shows that **Chimera** preserves quality substantially better as the generation horizon increases. From 5 to 30 seconds, its FID increases only from 77.1 to 82.1, a relative degradation of 6.5%, compared with 50.5% for Wan2.1 and 53.6% for HunyuanVideo-1.5. Its FVD increases from 685.8 to 829.5, corresponding to 20.9% degradation, whereas Wan2.1 and HunyuanVideo-1.5 degrade by 33.9% and 33.7%, respectively. At 30 seconds, **Chimera** also attains the lowest measured absolute FID and FVD among the three models. These results are consistent with the architectural motivation of our NoPE design: KDA repeatedly applies the same content-adaptive recurrent update beyond the training horizon, while the temporally causal convolution supplies local structure through fixed-support kernels that introduce no



Figure 14 Zero-shot high-resolution generation. Our model generate images at 4K (left panel) and 2K (right panel) resolutions in a zero-shot manner.

length-dependent positional phases. The resulting backbone thus exhibits natural zero-shot temporal extrapolation.

The qualitative comparisons in Fig. 17 reveal the same trend: Wan2.1 becomes blurry and nearly static during long-horizon inference, while HunyuanVideo-1.5 produces sharper frames but likewise exhibits little motion. In contrast, **Chimera** remains sharp and continues to evolve throughout the 30-second sequence, avoiding both failure modes.

Benchmark evaluation. To quantitatively assess the image-generation capability of **Chimera**, we evaluate it on two widely used image benchmarks, GenEval and DPG-Bench. Together, these benchmarks measure compositional accuracy across attributes such as object count, color, and spatial relations, as well as overall text-image alignment. Table 2 reports the results.

Chimera achieves an overall GenEval score of 0.82 and a DPG-Bench score of 85.12. It matches FLUX.1-dev and Z-Image-Turbo on GenEval while outperforming both on DPG-Bench, and it surpasses every other evaluated baseline except Seedream 3.0 on both aggregate metrics. Although our limited training budget prevents **Chimera** from matching the strongest large-scale image-generation model, its consistent advantage over the remaining baselines demonstrates strong compositional understanding and prompt alignment for a unified model trained with much less compute.

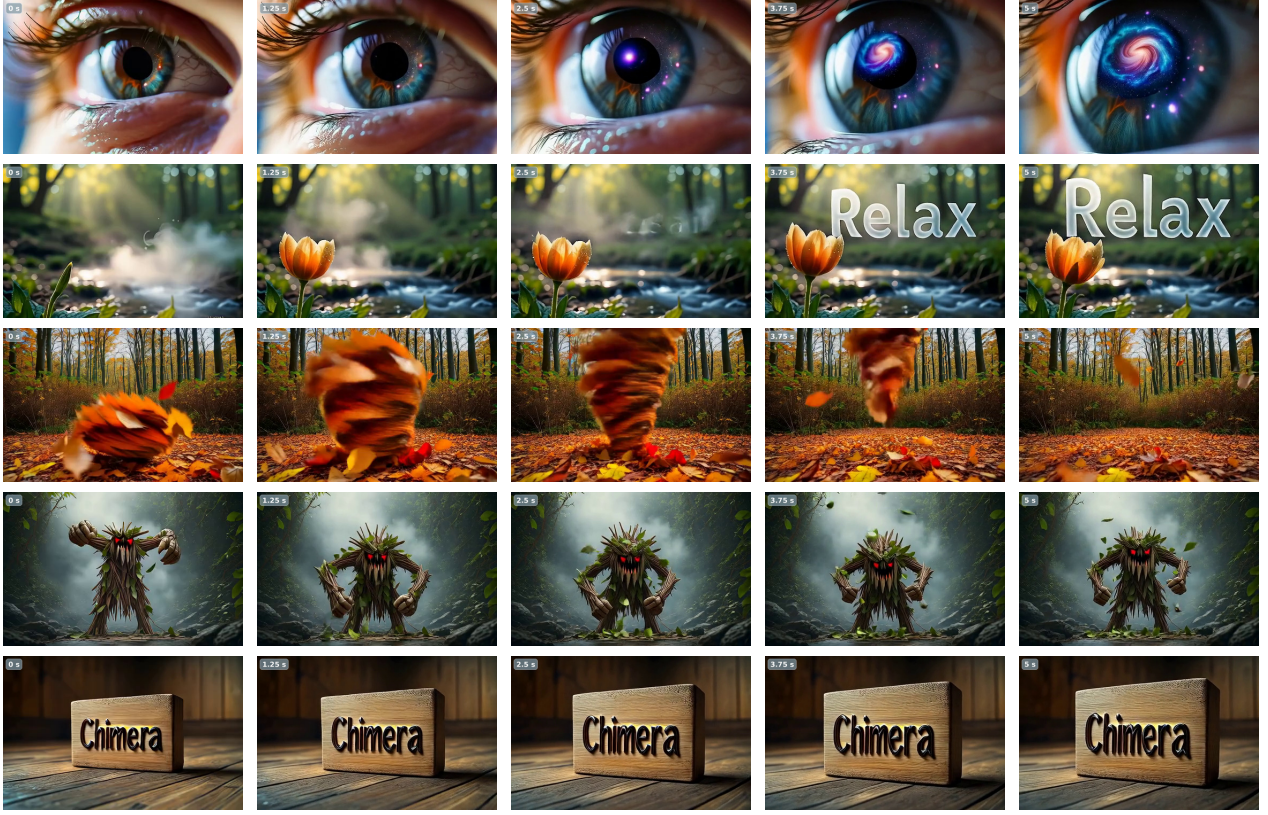


Figure 15 Qualitative text-to-video samples from *Chimera*. Each row shows five evenly spaced frames of one 81-frame video generated at 448×288 resolution with classifier-free guidance scale 8, with time proceeding from left to right.

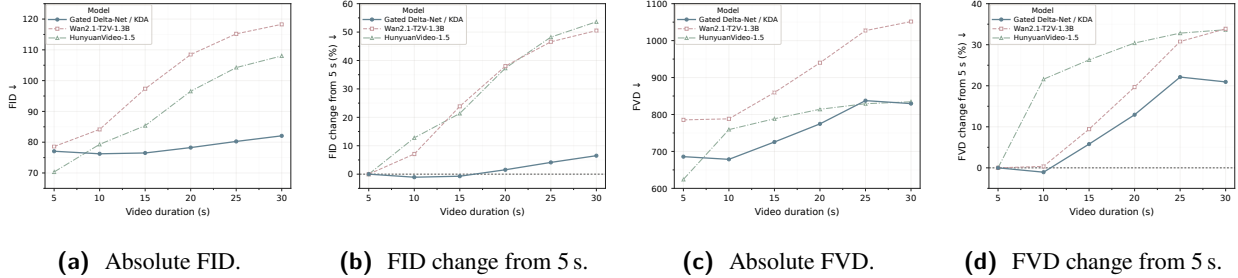


Figure 16 Zero-shot video-length extrapolation beyond the 5-second training horizon. We generate videos of 5–30 seconds without length-specific finetuning and evaluate only the final 5 seconds of every video. KDA denotes our *Chimera* model. Panels (a) and (c) report absolute FID and FVD, while panels (b) and (d) report the percentage change relative to each model’s own 5-second result. Evaluation uses 512 videos per model and duration, with 512×81 tail frames for FID and 512 tail clips for FVD. Lower is better, and a positive relative change denotes degradation.

5.6 Ablation Studies

Hybrid KDA–MLA attention improves memory efficiency and latency. To isolate the system-level benefit of hybrid attention, we compare our KDA/MLA backbone with an otherwise matched MHA/MLA counterpart, both with approximately 2B activated parameters. Both use the same 3:1 schedule, but every KDA layer in the counterpart is replaced by conventional multi-head self-attention, while the periodic MLA layers and all other components remain unchanged. We use batch size one and BF16 activations on an NVIDIA A100-SXM4-80GB, with 512 text tokens and 18×28 visual tokens per temporal slice, and vary the total sequence length. The memory experiment measures peak allocated CUDA memory: we allocate, touch, and retain all cache tensors on device during one complete denoiser forward. The 24 MHA layers store sequence-

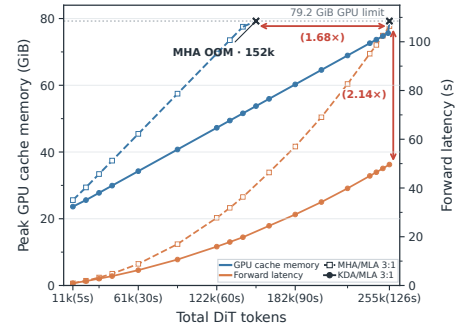
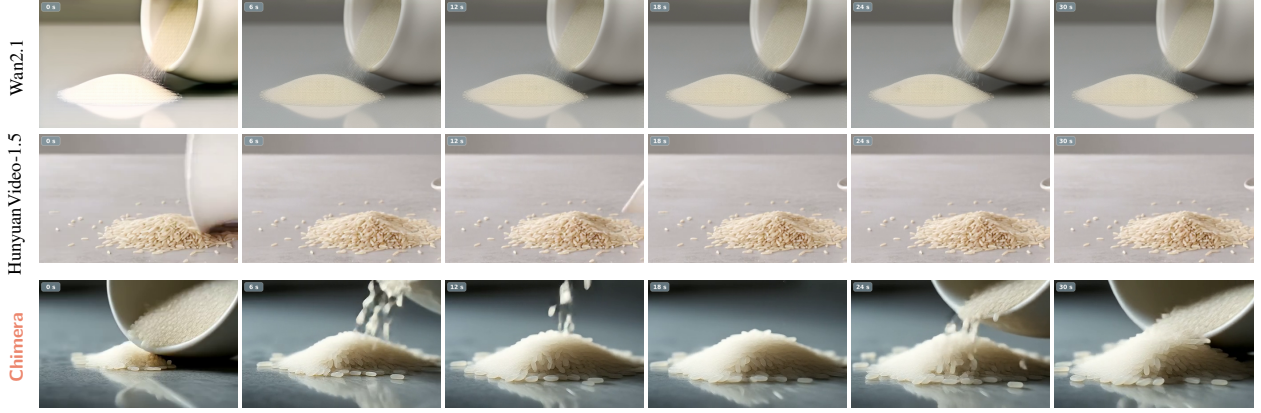


Figure 18 Memory and latency growth with tokens. We compare 3:1 KDA/MLA and MHA/MLA backbones with 2B activated parameters.



(a) The bowl slowly tips from upright to its side and remains at rest while rice grains pour out in a steady stream, cascading downward and spreading forward across the counter, some grains bouncing slightly before settling into a small mound; the countertop stays stationary.



(b) The rose’s petals gradually unfurl outward from the center in a slow, natural opening motion while the background stays fixed; no other movement occurs.

Figure 17 Qualitative zero-shot video-length extrapolation to 30 seconds. For each prompt, we show six frames sampled at 6-second intervals from videos generated by Wan2.1, HunyuanVideo-1.5, and **Chimera**, with time proceeding from left to right. All models use the same prompt, seed, resolution, frame rate, and duration within each comparison.

dependent BF16 key–value caches of size $\mathcal{O}(NHd_h) = \mathcal{O}(ND)$, whereas the 24 KDA layers retain fixed-size FP32 recurrent states of shape $H \times d_h \times d_h$, with memory $\mathcal{O}(Hd_h^2)$ independent of N . Both backbones retain the same eight BF16 MLA latent key–value caches. MHA is implemented via FlashAttention. For latency, we time only the denoiser forward, exclude cache construction, synchronize CUDA, and report the median of three post-warm-up runs.

Fig. 18 shows that the MHA/MLA backbone reaches the GPU memory limit and first runs out of memory at approximately 152k tokens, whereas the KDA/MLA backbone completes the 255k-token test on the same GPU, supporting more than $1.68\times$ the sequence length under the roll-out-cache setting. Although FlashAttention removes the quadratic attention workspace, MHA still incurs quadratic attention arithmetic and its latency therefore grows more rapidly with sequence length. At 255k tokens, KDA/MLA is $2.14\times$ faster than MHA/MLA. Thus, replacing most MHA layers with KDA substantially reduces persistent cache growth and long-sequence latency, while the shared periodic MLA layers preserve direct global token interactions.

MoE, iHC, and HeteroP improve pre-training compute efficiency. We ablate the three components cumulatively from a 2B Chimera-dense baseline. We first replace its dense FFNs with sparsely activated MoE layers while matching the activated parameter count. We then add iHC to provide adaptive read and write mappings over multiple residual streams. Finally, we apply the HeteroP (Sec. 4.1), which transfers the near-optimal base learning rate and the associated module-wise optimizer settings.

Fig. 19 shows that every addition improves the loss–compute trajectory. At a common training loss of 0.149, Chimera-

Table 2 Text-to-image generation results on GenEval and DPG-Bench. Higher values are better. Note that our **Chimera** is trained with a budget of only 600 H100 days.

Model	GenEval							DPG
	Single Obj.	Two Obj.	Counting	Colors	Position	Color Attri.	Overall↑	Overall↑
<i>Generation Only</i>								
PixArt- α [12]	0.98	0.50	0.44	0.80	0.08	0.07	0.48	71.11
SD v2.1 [67]	0.98	0.51	0.44	0.85	0.07	0.17	0.50	68.09
DALL-E 2 [66]	0.94	0.66	0.49	0.77	0.10	0.19	0.52	–
SDXL	0.98	0.74	0.39	0.85	0.15	0.23	0.55	74.65
DALL-E 3	0.96	0.87	0.47	0.83	0.43	0.45	0.67	83.50
SD3-Medium [24]	0.99	0.94	0.72	0.89	0.33	0.60	0.74	84.08
FLUX.1-dev [†]	0.98	0.93	0.75	0.93	0.68	0.65	0.82	84.00
Seedream 3.0	0.99	0.96	0.91	0.93	0.47	0.80	0.84	88.27
Z-Image-Turbo [8]	1.00	0.95	0.77	0.89	0.65	0.68	0.82	84.86
Chimera	0.98	0.88	0.85	0.88	0.62	0.68	0.82	85.12

dense requires 2.55×10^{20} FLOPs. Introducing MoE reduces this requirement to 1.76×10^{20} FLOPs, an approximately $1.5\times$ compute-efficiency gain. Adding iHC further reduces it to 1.50×10^{20} FLOPs, yielding a $1.7\times$ gain over the dense baseline. The complete configuration with HeteroP reaches the same loss with 6.27×10^{19} FLOPs, corresponding to a $4.1\times$ gain. The progression indicates that sparse capacity and multi-stream residual routing provide complementary gains, while HeteroP contributes the largest additional improvement. Consistent with Fig. 7, the HeteroP parameterization allows the larger base learning rate to remain stable and near-optimal, accelerating convergence without sacrificing training stability.

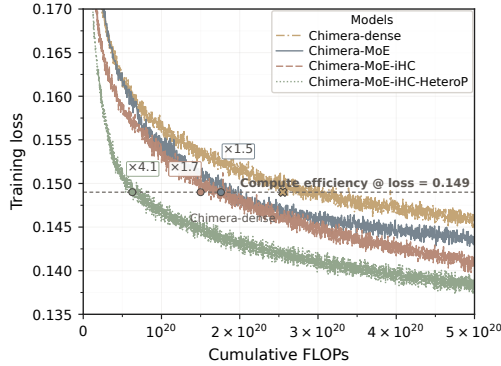


Figure 19 Component-wise pre-training compute efficiency.

Under HeteroP recipe, every curve keeps a wide and flat basin around $\eta_{\text{base}} = 10^{-3}$ and degrades gracefully up to $\eta_{\text{base}} = 3 \times 10^{-3}$. Under SP, the loss instead rises abruptly once the global learning rate exceeds each model’s optimum by only a small factor, and several runs blow up outright. The smallest width model leaves the plotted loss range already near 4×10^{-4} , mid-sized widths explode between 10^{-3} and 3×10^{-3} , and the deepest models spike sharply at 2×10^{-3} . These explosions also break the clean loss ordering by scale that HeteroP maintains at every swept base learning rate. SP therefore fails in two compounding ways, drifting optima and high-learning-rate divergence, which makes proxy-based hyperparameter tuning unreliable.

Scaling-law bias without HeteroP. The mis-tuning above is not only a per-run nuisance: it biases the scaling law fitted on such runs. To quantify this, we repeat the 256² image study of Sec. 5.3 on a matched model family trained under SP, where every model reuses the proxy-tuned global learning rate and therefore carries a scale-dependent mis-tuning penalty. The effect is directly visible in Fig. 20a: the loss trajectories are noisier than their HeteroP counterparts in Fig. 8, and several large models exhibit loss spikes late in training.

Learning-rate transfer fails without HeteroP. We first ablate the transfer recipe at the level of individual runs. The bottom row of Fig. 7 repeats the width and depth sweeps of Sec. 5.2 under standard parameterization (SP), using the same numerical grid for the unscaled global learning rate but without our transfer rules. In contrast to the aligned minima in the top row, the optimal learning rate under SP is no longer consistent across scales. The fitted optima in Figs. 7c and 7d spread over a $6\times$ range along the width family, from 10^{-4} for the smallest model to about 6×10^{-4} for the larger ones, and shift toward smaller values as depth grows. No single global learning rate is simultaneously near-optimal for the whole family, so a global rate tuned on the proxy is necessarily mis-tuned at most other scales.

The sweeps further reveal a stability failure at high learning rates.

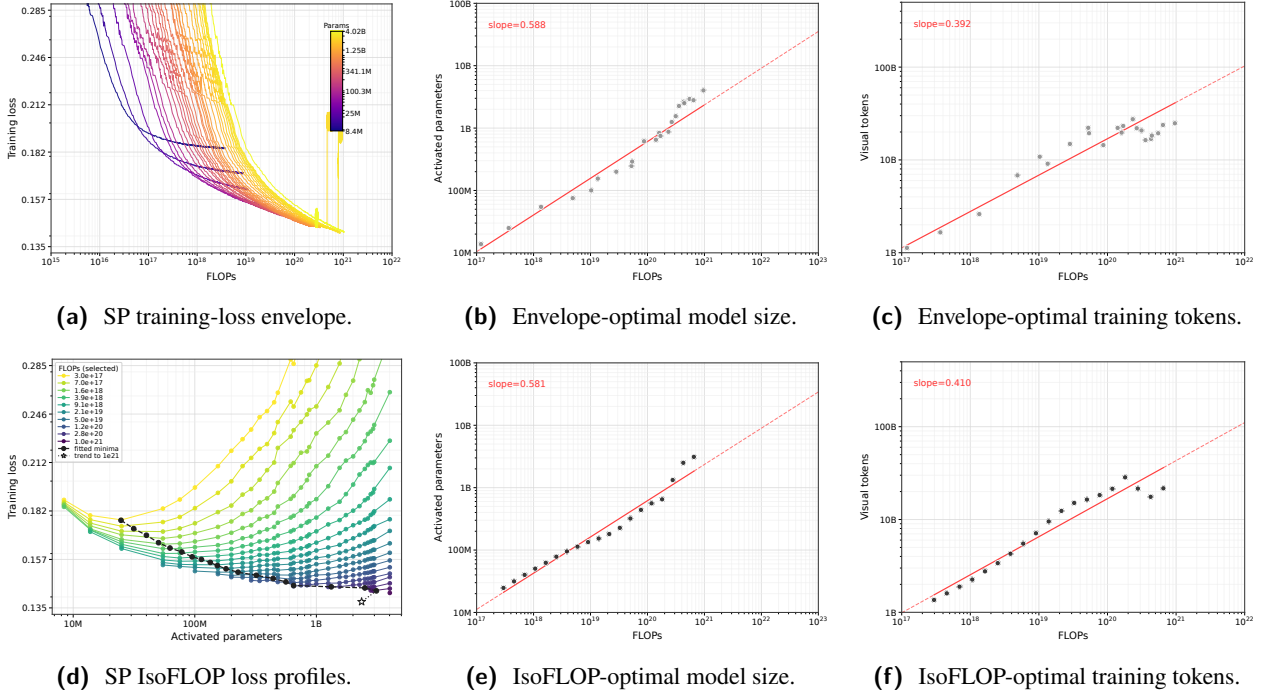


Figure 20 Compute-optimal scaling without HeteroP (256^2 image pre-training), leading to conservative scaling recipes compared with Fig. 8. All runs use standard parameterization (SP) with the proxy-tuned global learning rate. **Top row, training-loss envelope:** (a) Diffusion training loss versus cumulative training FLOPs for the same model family as Fig. 8b; the trajectories are noisier and several large models exhibit late-training loss spikes. (b) Envelope-optimal model size $N_{\text{opt}}(C)$, whose fitted exponent increases to 0.588 from 0.505 under HeteroP. (c) The corresponding optimal training tokens $D_{\text{opt}}(C)$, whose exponent decreases to 0.392 from 0.484. **Bottom row, IsoFLOP profiles:** (d) Fixed-compute loss profiles, with the fitted minimum of each profile highlighted. (e) IsoFLOP-optimal model size, with exponent 0.581 instead of the transferred value 0.481 in Fig. 8e. (f) The corresponding optimal training tokens, with exponent 0.410 instead of 0.511. Both estimators shift the allocation toward oversized and undertrained models, demonstrating that scale-dependent learning-rate mis-tuning biases the fitted scaling law.

The distortion propagates into the fitted law. The envelope estimate on the SP runs gives $N_{\text{opt}}(C) \propto C^{0.588}$ and $D_{\text{opt}}(C) \propto C^{0.392}$ (Fig. 20 (a-c)), and the IsoFLOP estimate gives $C^{0.581}$ and $C^{0.410}$ (Fig. 20 (d-f)). Both estimators inflate the model-size exponent by about 0.08–0.10 relative to the HeteroP fits (0.505 and 0.481) and deflate the token exponent by a similar margin. Because the mis-tuning penalty varies systematically with scale, the per-budget optima are shifted rather than merely noisier, and the near-balanced allocation of Sec. 5.3 is replaced by a spurious preference for parameters over data. Taken at face value, the SP law prescribes a compute-optimal model that grows about $1.2\times$ larger per decade of compute relative to the HeteroP law, reaching roughly $2\times$ oversized, and correspondingly undertrained, when extrapolated three decades beyond the fitted range. HeteroP is therefore not an optional refinement but a prerequisite for reliable compute-optimal scaling laws.

5.7 Discussions

We discuss several techniques, including Muon optimizer, MoE, and details of time embedding.

Muon optimizer. Following language model research, we experiment with using the Muon optimizer [42] for hidden layers. Although we investigated sufficient efforts in tuning hyper-parameters and splitting the correct parameter groups, we report that the loss under Muon optimizer is systematically worse than using AdamW. The only case where Muon outperforms is the beginning of training under large learning rate. This conclusion holds across different model architectures, scales, and whether HeteroP and MoE are used. We hypothesize that Muon underperforms Adam in diffusion visual generation because it lacks Adam’s implicit low-rank bias. Diffusion training combines denoising objectives across diverse timesteps and noise levels, so the useful optimization signal may concentrate in a small set of stable directions, while many other directions are noisy or inconsistent. To test this, we compare the spectra of parameter updates and trained weights from Adam and Muon models. We find that Adam concentrates most of its update and

weight energy in a much smaller number of singular directions, whereas Muon produces substantially more full-rank and spectrally uniform updates. This supports the hypothesis that Adam implicitly filters for persistent low-dimensional denoising directions, while Muon spreads update energy more evenly across the matrix. Consequently, Muon can improve early training under large learning rates, but later makes slower progress because its update energy is diluted away from the stable low-rank subspace important for diffusion optimization.

MoE. We observe that MoE scaling provides much weaker compute efficiency in diffusion-based visual generation than what is commonly observed in language modeling. In our experiments, increasing sparsity yields only around a $1.5\times$ improvement in compute efficiency, substantially below the gain predicted by the common $\sqrt{\text{sparsity}}$ heuristic. This suggests that simply increasing the number of inactive experts does not translate into proportional effective capacity for diffusion models. We hypothesize that this gap arises because visual denoising offers less exploitable expert specialization: visual tokens are continuous, spatially redundant, and highly correlated, while diffusion timesteps share a large common denoising component that must be modeled consistently. Consistent with this hypothesis, we find that expert routing exhibits weak specialization and often becomes more uniform across tokens or timesteps. These results suggest that MoE sparsity in diffusion models is constrained by the intrinsic shared structure of visual denoising, rather than only by routing or implementation details.

Time embedding. We also observe that the timestep embedding network is a model-size-sensitive component in diffusion training. In our default design, sinusoidal timestep features are mapped to the model dimension through an MLP. However, using an overly wide hidden dimension for this MLP can destabilize training; for example, a 4096-dimensional hidden layer is unstable for a 1024-width backbone. This suggests that the timestep MLP should not be treated as a fixed auxiliary module independent of model scale. We hypothesize that the reason is that timestep embeddings act as a global conditioning signal, often controlling layer-wise modulation such as scale, shift, or gating. If the timestep MLP is too wide relative to the backbone, it can produce an overly strong or sharp timestep-dependent modulation, amplifying gradient conflicts across noise levels and disrupting the shared denoising representation. We therefore scale the hidden dimension of the timestep MLP with the backbone width, which improves training stability.

6 Related Works

Diffusion Models for Visual Generation. Diffusion models have been repeatedly adapted to the structure, scale, and controllability requirements of visual data for improving visual generation quality. Early work adopt convolutional U-Net backbones [33, 73], improvements in noise schedules [60], and guidance [19] to improve generation fidelity. Later systems adopted hierarchical [66, 68] and latent-space designs [67]. Combined with classifier-free guidance [32] and strong language encoders [59], these models enabled high-quality controllable generation. Following work has revisited both the denoising backbone and the latent representation. DiT replace convolutional U-Nets with Transformer architectures, enabling stronger scaling [12, 24, 63]. At the same time, the VAE latent space used by latent diffusion has been identified as a bottleneck for visual quality, motivating representation autoencoders [10, 102] and renewed interest in pixel-space or hybrid pixel-latent generation [49, 96].

Beyond image generation, Diffusion models for video generation have evolved through several stages. Early work primarily adapted image diffusion models to videos by adding temporal modeling modules, reusing strong image priors while learning motion [5, 6, 34, 35, 71]. More recently, the focus has shifted from generating short clips to long-horizon video. One line of work reformulates video diffusion as autoregressive or causal rollout [11, 41, 72]. Another line of work, which is more relevant to **Chimera**, is adopting efficient Transformer variants, by leveraging spatial-temporal attention patterns [21, 88, 89], adopting linear attention [13, 83], and sparse attention [76, 98, 100]. In this paper, we develop a novel hybrid attention architecture with minimal inductive bias, enabling native joint training of image and video generation. We showcase efficient, stable training and a scaling law that attains the Chinchilla-optimal allocation.

Efficient Attention Mechanisms. The efficient attention mechanisms broadly categorized by what they reduce: the representation being attended to, the attention operator itself, or the set of token interactions. The first line compresses the key-value representation while preserving the standard softmax attention formulation [1, 52, 69]. The second line removes the softmax operation, providing linear-time or constant-state inference while preserving parallelizable training [29, 45, 75, 94]. The third line sparsifies attention by computing only a subset of token interactions, by designing attention patterns [14, 105] or making it learnable [54, 97, 103]. In this work, we adapt the architecture of Kimi Linear [78] to visual generation while injecting minimal inductive bias of spatial-temporal modeling.

Model Scaling. Model scaling research studies how model quality changes as we allocate more parameters, data, and compute. Early scaling laws showed that language-model loss follows predictable power-law trends with model size, dataset size, and training compute [44]. Compute-optimal scaling later systematically studies the optimal allocation of data and model size given a fixed compute budget [36]. These scaling principles have also been extended beyond language modeling to autoregressive image, video, and multimodal generation [30, 50]. To reduce the cost of hyperparameter search, a complementary line studies parameterization and hyperparameter transfer [18, 58, 92]. We propose HeteroP that builds on these principles but targets a different structural issue: heterogeneous backbones require module-specific width ratios rather than one global model-width ratio. Besides, model scaling is not limited to dense parameter growth. Sparse activation and mixture-of-experts architectures scale total capacity while keeping activated compute relatively low [25, 26, 48]. In this work, we incorporate these techniques for a systemically study of scaling laws of visual Diffusion model in the contemporary era.

7 Future Exploration and Conclusion

Conclusion. We introduced **Chimera**, a hybrid, heterogeneous, single-stream diffusion backbone co-designed with a principled scaling recipe for token-extensive visual generation. Architecturally, **Chimera** combines KDA for efficient long-context state tracking, periodic MLA for global interaction, and modality-aware short convolutions for local spatio-temporal structure, enabling a unified text–image–video sequence without positional embeddings. Sparse MoE, iHC, and sandwich normalization further improve capacity and training stability. To scale this heterogeneous architecture, HeteroP transfers proxy-tuned hyperparameters and produces a consistently tuned model family from which we derive compute-optimal laws over activated model size, visual training tokens, and image–video composition. We hope that the architectural principles embodied by **Chimera** provide useful guidance for designing future long-context visual generators. More broadly, we hope HeteroP and the compute-optimal framework offer a reusable way to turn inexpensive proxy experiments into reliable model, data, and compute-allocation decisions for multimodal generators.

Future Exploration. There are some extensions for **Chimera**’s future explorations. First, our study deliberately isolates width and depth transfer. Although HeteroP parameterizes the heterogeneous modules according to their functional fan-in, the scaling family still holds several structural axes fixed, including the iHC residual-stream count, the MoE expert and top- K routing configuration (and hence its sparsity), and the MLA KV-compression ratio as detailed in Sec. 4.1. A more complete and flexible parameterization should transfer from an inexpensive dense proxy to sparse target models and make residual-stream count, expert count, routing sparsity, and related structural choices scale-dependent rather than preset. Second, while **Chimera** already processes text, image, and video tokens in one stream, the present study evaluates only text-to-image and text-to-video generation. Incorporating the successful design choices in **Chimera** to single-stream hybrid-linear-global architectures (e.g., Qwen3.5 [79], K3 [77]) is a promising path toward a single model for both long-context multimodal understanding and generation. Third, **Chimera** still relies on a diffusion-specific AdaLN modulation. Our preliminary experiments suggest that this can be absorbed into iHC by conditioning its stream read and write mappings directly on the diffusion timestep. Removing the separate AdaLN modulation would further align hybrid language-model and hybrid diffusion architectures toward a shared backbone.

Acknowledgments. We thank Hongwu Peng for insightful discussions of the Muon optimizer and μ P; Bi Sai for constructive feedback; and Tong Sun, Rajiv Jain, Kalyan Sunkavalli, and John Yang for supporting this project.

References

- [1] Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, 2023.
- [2] Zeyuan Allen-Zhu. Physics of language models: Part 4.1, architecture design and the magic of canon layers. *Advances in Neural Information Processing Systems*, 38:42349–42369, 2026.
- [3] Federico Barbero, Alex Vitvitskyi, Christos Perivolaropoulos, Razvan Pascanu, and Petar Veličković. Round and round we go! what makes rotary positional encodings useful? *arXiv preprint arXiv:2410.06205*, 2024.
- [4] Black Forest Labs. FLUX.2: Frontier Visual Intelligence. <https://bfl.ai/blog/flux-2>, 2025.
- [5] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.
- [6] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22563–22575, 2023.
- [7] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators, 2024. <https://openai.com/research/video-generation-models-as-world-simulators>.
- [8] Huanqia Cai, Sihan Cao, Ruoyi Du, Peng Gao, Steven Hoi, Zhaohui Hou, Shijie Huang, Dengyang Jiang, Xin Jin, Liangchen Li, et al. Z-image: An efficient image generation foundation model with single-stream diffusion transformer. *arXiv preprint arXiv:2511.22699*, 2025.
- [9] Matias D Cattaneo and Boris Shigida. The effect of mini-batch noise on the implicit bias of adam. *arXiv preprint arXiv:2602.01642*, 2026.
- [10] Bowei Chen, Sai Bi, Hao Tan, He Zhang, Tianyuan Zhang, Zhengqi Li, Yuanjun Xiong, Jianming Zhang, and Kai Zhang. Aligning visual foundation encoders to tokenizers for diffusion models. In *The Fourteenth International Conference on Learning Representations*, 2025.
- [11] Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.
- [12] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-alpha: Fast training of diffusion transformer for photorealistic text-to-image synthesis. In *International conference on learning representations*, volume 2024, pages 57611–57640, 2024.
- [13] Junsong Chen, Yuyang Zhao, Jincheng Yu, Ruihang Chu, Junyu Chen, Shuai Yang, Xianbang Wang, Yicheng Pan, Daquan Zhou, Huan Ling, et al. Sana-video: Efficient video generation with block linear diffusion transformer. *arXiv preprint arXiv:2509.24695*, 2025.
- [14] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.
- [15] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, and Chunhua Shen. Conditional positional encodings for vision transformers. In *International Conference on Learning Representations*, 2023.
- [16] Hyung Won Chung, Noah Constant, Xavier Garcia, Adam Roberts, Yi Tay, Sharan Narang, and Orhan Firat. Unimax: Fairer and more effective language sampling for large-scale multilingual pretraining. *arXiv preprint arXiv:2304.09151*, 2023.
- [17] Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1280–1297, 2024.
- [18] Nolan Dey, Bin Zhang, Lorenzo Noci, Mufan Li, Blake Bordelon, Shane Bergsma, Cengiz Pehlevan, Boris Hanin, and Joel Hestness. Don’t be lazy: Completex enables compute-efficient deep transformers. *Advances in Neural Information Processing Systems*, 38:137707–137739, 2026.

- [19] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.
- [20] Sander Dieleman. Diffusion is spectral autoregression. <https://sander.ai/2024/09/02/spectral-autoregression.html>, September 2024. Blog post.
- [21] Hangliang Ding, Dacheng Li, Runlong Su, Peiyuan Zhang, Zhijie Deng, Ion Stoica, and Hao Zhang. Efficient-vdit: Efficient video diffusion transformers with attention tile. *arXiv preprint arXiv:2502.06155*, 2025.
- [22] Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, and Jie Tang. Cogview: Mastering text-to-image generation via transformers. In *Advances in Neural Information Processing Systems*, volume 34, pages 19822–19835, 2021.
- [23] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12, 2021.
- [24] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.
- [25] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.
- [26] Zhengcong Fei, Mingyuan Fan, Changqian Yu, Debang Li, and Junshi Huang. Scaling diffusion transformers to 16 billion parameters, 2024. URL <https://arxiv.org/abs/2407.116>, 33, 2024.
- [27] Alberto Fernández-Hernández, Cristian Pérez-Corral, Jose I Mestre, Manuel F Dolz, and Enrique S Quintana-Ortí. Why adam works better with $\beta_1=\beta_2$: The missing gradient scale invariance principle. *arXiv preprint arXiv:2601.21739*, 2026.
- [28] Dhruva Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152, 2023.
- [29] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.
- [30] Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020.
- [31] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.
- [32] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- [33] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.
- [34] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.
- [35] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in neural information processing systems*, 35:8633–8646, 2022.
- [36] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, DDL Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 10, 2022.
- [37] Yicong Hong, Long Mai, Yuan Yao, and Feng Liu. Pushing the boundaries of state space models for image and video generation, 2025. <https://arxiv.org/abs/2502.00972>.
- [38] Yicong Hong, Yiqun Mei, Chongjian Ge, Yiran Xu, Yang Zhou, Sai Bi, Yannick Hold-Geoffroy, Mike Roberts, Matthew Fisher, Eli Shechtman, et al. Relic: Interactive video world model with long-horizon memory. *arXiv preprint arXiv:2512.04040*, 2025.
- [39] Vincent Tao Hu, Stefan Andreas Baumann, Ming Gui, Olga Grebenkova, Pingchuan Ma, Johannes Fischer, and Björn Ommer. Zigma: A dit-style zigzag mamba diffusion model. In *European conference on computer vision*, pages 148–166. Springer, 2024.
- [40] Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. Ella: Equip diffusion models with llm for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024.

- [41] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *Advances in Neural Information Processing Systems*, 38:167283–167308, 2026.
- [42] Keller Jordan, Yuchen Jin, Vlado Boza, You Jiacheng, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. URL <https://kellerjordan.github.io/posts/muon>, 6(3):4, 2024.
- [43] Xuan Ju, Tianyu Wang, Yuqian Zhou, He Zhang, Qing Liu, Nanxuan Zhao, Zhifei Zhang, Yijun Li, Yuanhao Cai, Shaoteng Liu, Daniil Pakhomov, Zhe Lin, Soo Ye Kim, and Qiang Xu. Editverse: Unifying image and video editing and generation with in-context learning. *arXiv preprint arXiv:2509.20360*, 2025.
- [44] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
- [45] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and Francois Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the International Conference on Machine Learning*, 2020.
- [46] Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36:24892–24928, 2023.
- [47] Kimi Team. Kimi linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025.
- [48] Dmitry Lepikhin, Hyukjoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*, 2020.
- [49] Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 36115–36125, 2026.
- [50] Zhengyang Liang, Hao He, Ceyuan Yang, and Bo Dai. Scaling laws for diffusion transformers. *arXiv preprint arXiv:2410.08184*, 2024.
- [51] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.
- [52] Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024.
- [53] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.
- [54] Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.
- [55] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.
- [56] Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. Vmamba: Visual state space model. *Advances in neural information processing systems*, 37:103031–103063, 2024.
- [57] William Merrill, Yanhong Li, Tyler Romero, Anej Svete, Caia Costello, Pradeep Dasigi, Dirk Groeneveld, David Heineman, Bailey Kuehl, Nathan Lambert, et al. Olmo hybrid: From theory to practice and back. *arXiv preprint arXiv:2604.03444*, 2026.
- [58] Bruno Mlodozieniec, Pierre Ablin, Louis Béthune, Dan Busbridge, Michal Klein, Jason Ramapuram, and Marco Cuturi. Completed hyperparameter transfer across modules, width, depth, batch and duration. *arXiv preprint arXiv:2512.22382*, 2025.
- [59] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.
- [60] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.
- [61] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.
- [62] Antonio Orvieto and Robert Gower. In search of adam’s secret sauce. *Advances in Neural Information Processing Systems*, 38: 63404–63442, 2026.

- [63] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.
- [64] Ofir Press, Noah A. Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2022.
- [65] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21 (140):1–67, 2020.
- [66] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.
- [67] 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.
- [68] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.
- [69] Noam Shazeer. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*, 2019.
- [70] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.
- [71] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.
- [72] Kiwhan Song, Boyuan Chen, Max Simchowitz, Yilun Du, Russ Tedrake, and Vincent Sitzmann. History-guided video diffusion. *arXiv preprint arXiv:2502.06764*, 2025.
- [73] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.
- [74] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.
- [75] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.
- [76] Xin Tan, Yuetao Chen, Yimin Jiang, Xing Chen, Kun Yan, Nan Duan, Yibo Zhu, Daxin Jiang, and Hong Xu. Dsv: Exploiting dynamic sparsity to accelerate large-scale video dit training. *arXiv preprint arXiv:2502.07590*, 2025.
- [77] Kimi Team. Kimi k3: Open frontier intelligence, 2026. <https://arxiv.org/abs/2607.24653>.
- [78] Kimi Team, Yu Zhang, Zongyu Lin, Xingcheng Yao, Jiaxi Hu, Fanqing Meng, Chengyin Liu, Xin Men, Songlin Yang, Zhiyuan Li, et al. Kimi linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025.
- [79] Qwen Team. Qwen3. 5-omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.
- [80] Tencent Hunyuan Foundation Model Team. Hunyuanvideo 1.5 technical report, 2025. <https://arxiv.org/abs/2511.18870>.
- [81] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.
- [82] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Fei Wu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.
- [83] Hongjie Wang, Chih-Yao Ma, Yen-Cheng Liu, Ji Hou, Tao Xu, Jialiang Wang, Felix Juefei-Xu, Yaqiao Luo, Peizhao Zhang, Tingbo Hou, et al. Lingen: Towards high-resolution minute-length text-to-video generation with linear computational complexity. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2578–2588, 2025.
- [84] Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. *arXiv preprint arXiv:2408.15664*, 2024.
- [85] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

- [86] Lilian Weng. Scaling laws, carefully. *lilianweng.github.io*, June 2026. <https://lilianweng.github.io/posts/2026-06-24-scaling-laws/>.
- [87] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.
- [88] Jianzong Wu, Liang Hou, Haotian Yang, Xin Tao, Ye Tian, Pengfei Wan, Di Zhang, and Yunhai Tong. Vmoba: Mixture-of-block attention for video diffusion models. *arXiv preprint arXiv:2506.23858*, 2025.
- [89] Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, et al. Sparse videogen: Accelerating video diffusion transformers with spatial-temporal sparsity. *arXiv preprint arXiv:2502.01776*, 2025.
- [90] Zhenda Xie, Yixuan Wei, Huanqi Cao, Chenggang Zhao, Chengqi Deng, Jiashi Li, Damai Dai, Huazuo Gao, Jiang Chang, Kuai Yu, et al. mhc: Manifold-constrained hyper-connections. *arXiv preprint arXiv:2512.24880*, 2025.
- [91] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.
- [92] Ge Yang, Edward Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tuning large neural networks via zero-shot hyperparameter transfer. *Advances in Neural Information Processing Systems*, 34:17084–17097, 2021.
- [93] Greg Yang, Edward J. Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. *arXiv preprint arXiv:2203.03466*, 2022.
- [94] Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. *Advances in neural information processing systems*, 37:115491–115522, 2024.
- [95] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, volume 2025, pages 83048–83077, 2025.
- [96] Yongsheng Yu, Wei Xiong, Weili Nie, Yichen Sheng, Shiqiu Liu, and Jiebo Luo. Pixeldit: Pixel diffusion transformers for image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14273–14282, 2026.
- [97] Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Yuxing Wei, Lean Wang, Zhiping Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 23078–23097, 2025.
- [98] Chenlu Zhan, Wen Li, Chuyu Shen, Jun Zhang, Suhui Wu, and Hao Zhang. Bidirectional sparse attention for faster video diffusion training. *arXiv preprint arXiv:2509.01085*, 2025.
- [99] Kai Zhang, Peng Wang, Sai Bi, Jianming Zhang, and Yuanjun Xiong. Knapformer: An online load balancer for efficient diffusion transformers training. *arXiv preprint arXiv:2508.06001*, 2025.
- [100] Peiyuan Zhang, Yongqi Chen, Haofeng Huang, Will Lin, Zhengzhong Liu, Ion Stoica, Eric Xing, and Hao Zhang. Faster video diffusion with trainable sparse attention. *Advances in Neural Information Processing Systems*, 38:152509–152534, 2026.
- [101] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.
- [102] Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. *arXiv preprint arXiv:2510.11690*, 2025.
- [103] Chenyu Zheng, Xinyu Zhang, Rongzhen Wang, Wei Huang, Zhi Tian, Weilin Huang, Jun Zhu, and Chongxuan Li. Scaling diffusion transformers efficiently via mup. *Advances in Neural Information Processing Systems*, 38:19372–19413, 2026.
- [104] Defa Zhu, Hongzhi Huang, Zihao Huang, Yutao Zeng, Yunyao Mao, Banggu Wu, Qiyang Min, and Xun Zhou. Hyper-connections. In *International Conference on Learning Representations*, volume 2025, pages 97183–97219, 2025.
- [105] Yuchen Zhu, Jing Shi, Chongjian Ge, Hao Tan, Yiran Xu, Wanrong Zhu, Jason Kuen, Koustava Goswami, Rajiv Jain, Yongxin Chen, et al. Flare: Diffusion for hybrid language model. *arXiv preprint arXiv:2606.01774*, 2026.

A Notation

Table 3 summarizes the main notation used throughout the paper. Symbols are grouped by component, and the scope of section-local or reused symbols is stated explicitly in the table.

Table 3 Main notation used in this paper.

Symbol	Meaning	Symbol	Meaning
<i>Diffusion objective (Sec. 3.1)</i>		<i>MLA (Sec. 3.3)</i>	
z_1	clean visual tokens (patchified VAE latents)	n_h	number of attention heads
z_0	Gaussian noise	U	compressed key-value latent
τ, z_τ	diffusion timestep; noised visual tokens at τ	K^{direct}	direct-key branch projected from the MLA input and shared across heads
v_τ	rectified-flow velocity target $z_1 - z_0$	K^{lat}, V	latent keys and values decoded from U
c	text-token sequence from the frozen text encoder	K_i	head- i key $[K^{\text{direct}}, K_i^{\text{lat}}]$
θ	trainable parameters of the denoiser	<i>MoE (Sec. 3.5)</i>	
$\mathcal{L}$	diffusion training objective evaluated at visual-token positions	u	token input to the MoE FFN
<i>Sequence and layout (Secs. 3.2, 3.3)</i>		N, K	section-local expert count ($N = 56$); activated experts per token ($K = 8$)
x	packed input sequence; reused after iHC expansion for the sequence-level multi-stream state	E_i	the i -th routed expert
$P_{\text{text}}, P_{\text{vis}}$	modality-specific input projections	$r(\cdot), g_i(u)$	router; routing weight of expert i
e_τ	timestep embedding	$b \in \mathbb{R}^N$	non-gradient load-balancing bias
T, H, W	temporal, height, width sizes of the latent grid	$\mathcal{T}(u)$	top- K expert set selected for token u
m	flattened index of the temporal-major raster order	$\text{Load}_i, \overline{\text{Load}}$	per-expert load; mean expert load; maximal load
L	sequence length	MaxVio	violation (Eq. 22)
<i>Block computation (Secs. 3.2, 3.6)</i>		<i>RoPE analysis (Sec. 3.3, App. C)</i>	
d	shared hidden dimension	θ_j	rotation frequency of rotary pair j (rad/token)
δ_A, γ_A	AdaLN shift and scale of the attention input	δ	token offset, query position minus key position
g^A, g^F	timestep-conditioned residual gates	$q^{(i)}, k^{(i)}$	components of q, k in rotary channel pair i
M	number of parallel residual streams ($M = 4$)	$\ q^{(i)}\ \ k^{(i)}\ $	amplitude of pair i (window-averaged q, k)
R, R_m	per-token multi-stream state; its m -th stream (stream-local index)	E_i	per-pair energy $\mathbb{E}_\ell \ q_\ell^{(i)}\ \ k_\ell^{(i)}\ $ over token positions (distinct from expert E_i)
$h_{\text{pre}}(R), h_{\text{post}}(R)$	token-dependent iHC read and write vectors	local / far mass	attention fraction at Chebyshev distance $\leq 1 / \geq 5$
α_h	learnable scale in the iHC coefficient generator	<i>Scaling (Secs. 4, 5.3)</i>	
Δ	gated attention or FFN update written to the streams	$\mathcal{M}^{(0)}, \mathcal{M}$	proxy and target models
H_{res}	residual-stream transition matrix (identity in iHC)	$n_{\text{blk}}^{(0)}, n_{\text{blk}}$	proxy and target block counts
<i>KDA (Sec. 3.3)</i>		$W^{(0)}, W$	corresponding proxy and target parameter groups
t	scan index $1, \dots, L$ (distinct from diffusion timestep τ)	m_W, m_L	target-to-proxy fan-in ratio; target-to-proxy block-count ratio
q_t, k_t, v_t	query, key, and value of token t	$\bar{\mathbf{h}}$	tuple of proxy-tuned base hyperparameters
d_q, d_k, d_v	per-head query, key, and value dimensions	$\mathcal{P}_W, \rho(W)$	transferred settings for W ; its functional role
S_t	recurrent state ($d_k \times d_v$) after token t	$\mathcal{T}_{\rho(W)}$	role-dependent transfer map
α_t	channel-wise forget gate (state decay)	N	activated parameter count in the scaling law
β_t	head-wise erase/write strength	D	cumulative number of visual latent positions processed
mShortConv	modality-aware short conv. (Sec. 3.4)	C	training compute, approximated by $C \approx 6ND$
		$\hat{\mathcal{L}}$	fitted diffusion-loss surface
		$E; A, B; a, b$	irreducible loss; positive coefficients; scaling exponents

B Representative Model Configurations

Table 4 reports the exact configurations visualized in Fig. 6. The proxy is used for hyperparameter tuning, the scaling family supports the compute-optimal analyses, and the final model is selected for full training.

Table 4 Representative **Chimera** configurations used in our experiments. All models instantiate the architecture of Sec. 3; structural ratios (MoE experts, KDA-to-MLA ratio, residual streams, KV-compression ratio, patchification) are held fixed across scales (Sec. 4.1). We report both per-token *activated* parameters and total denoiser parameters, where total parameters count all routed MoE experts. The full sweep contains additional intermediate sizes.

Model	Act. Params	Total Params	Width	FFN Hidden	Depth
Proxy	22M	59M	512	2048	4
Scaling family	55M	253M	384	1536	16
	200M	893M	896	2432	16
	291M	1.35B	1024	4096	16
	393M	1.88B	1152	5120	16
	482M	2.46B	896	4096	32
	748M	3.58B	1280	4096	32
	873M	4.41B	1280	5120	32
	999M	5.25B	1280	6144	32
	1.55B	7.99B	1664	6656	32
	2.01B	9.93B	2048	6400	32
	2.26B	10.33B	2432	5248	32
	2.50B	11.92B	2432	6400	32
	2.65B	10.79B	2944	4736	32
	2.80B	14.26B	2304	9216	32
	2.93B	14.25B	2560	7296	32
	2.99B	15.09B	2432	9088	32
	4.02B	19.31B	3072	8448	32
Final Training Model	2.22B	11.28B	2048	8192	32

C Visual RoPE Analysis: Protocol and Results

This appendix details the FLUX.2 and Wan2.2 analysis summarized in Sec. 3.3. All statistics are computed from the models’ own sampling trajectories, so the probed activations match the regime in which the models operate. Fig. 21 visualizes the per-pair decompositions, the frequency-band ablations, and the division of labor across all heads.

Models and Sampling. We probe the public weights of Wan2.2-T2V-A14B and FLUX.2-dev. Wan2.2 has 40 layers with 40 self-attention heads of dimension 128; axial RoPE splits each head into 22 temporal, 21 height, and 21 width channel pairs with per-axis frequencies $\theta_j = 10000^{-2j/d_a}$, spanning 1 down to $\sim 1.5 \times 10^{-4}$ rad/token. Its two denoising experts (applied above and below $t = 875$) are probed separately. FLUX.2 has 8 double-stream and 48 single-stream blocks with 48 heads of dimension 128; its axial RoPE assigns 16 pairs to each of four axes (time, height, width, text index) with base 2000, spanning 1 down to $\sim 8 \times 10^{-4}$ rad/token. Image tokens carry coordinates only on the height and width axes, so the time- and text-axis channels, half of every head, are unrotated between image tokens; text tokens attend in the same sequence and use the text axis. We sample Wan2.2 at 480×832 with 33 frames (a $9 \times 30 \times 52$ token grid, 14,040 visual tokens) using UniPC with 40 steps and classifier-free guidance 4.0/3.0, and FLUX.2 at 1024^2 (a 64×64 grid) using its 50-step flow schedule with embedded guidance 4.0. Each model is probed on six shared prompts covering people, animals, nature, urban scenes, close-ups, and still-life motion, at three denoising timesteps ($t = \{964, 750, 347\}$ for Wan2.2, spanning both experts; noise levels $\sigma \approx \{0.99, 0.88, 0.46\}$ for FLUX.2), giving 18 settings per model.

Capture and Head Statistics. At each captured step we record, for every self-attention layer, the per-head queries and keys after query-key normalization and before rotation (for Wan2.2, on the conditional branch of classifier-free guidance). For a fixed random subset of 512 visual query tokens, shared across all layers and settings, we form the full attention rows and measure per head: the *local attention mass* on the immediate spatio-temporal neighborhood, *i.e.*, the fraction of a query’s attention falling on tokens within Chebyshev distance 1 on the token grid, self excluded (26 neighbors for video, 8 for images); the *far-field mass* at Chebyshev distance ≥ 5 ; the mean attended distance; and the

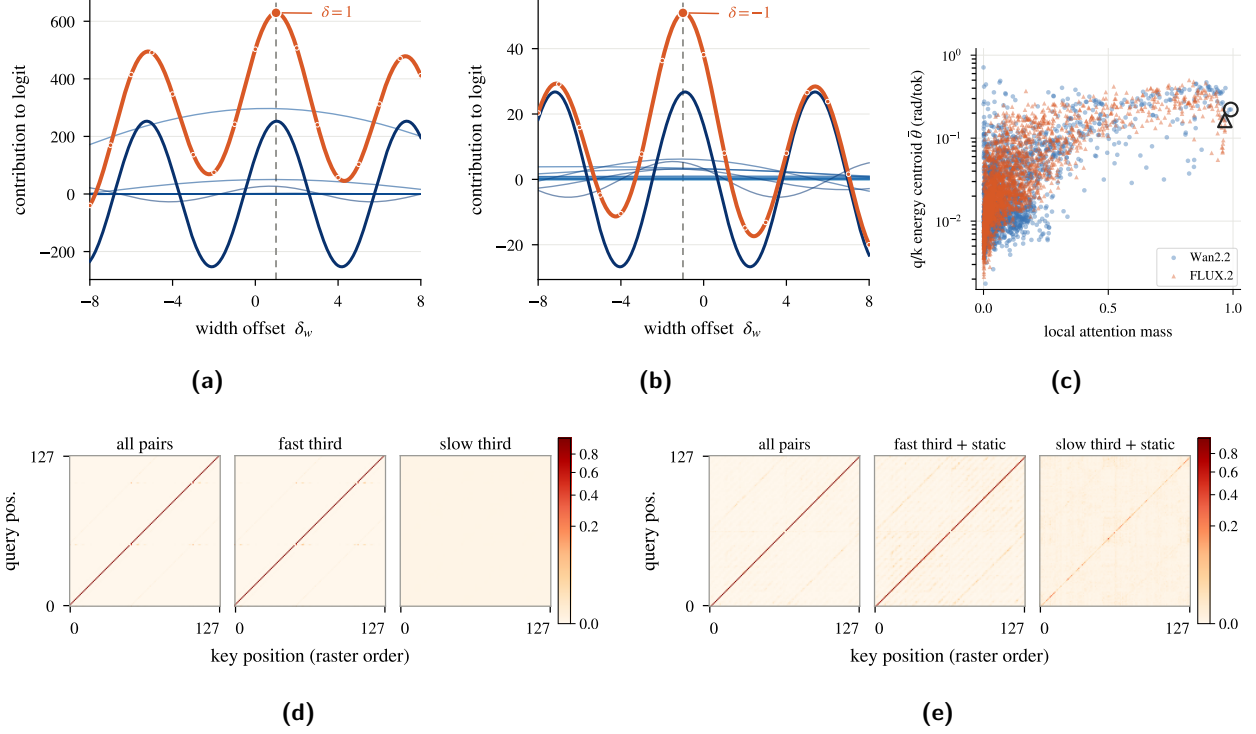


Figure 21 RoPE frequency usage in visual diffusion Transformers. (a) Per-pair logit decomposition of the strongest local head of Wan2.2 (layer 33, head 30) along the width axis: one curve per width-axis rotary pair (darker blue indicates faster rotation), whose sum (orange) peaks at offset +1, the raster-order previous token; pairs on other axes add offset-independent constants (not shown). (b) The same decomposition for the strongest local head of FLUX.2 (layer 46, head 33), peaking at the raster-order next token ($\delta_w = -1$). (c) Energy-weighted rotation speed of the query-key channels versus local attention mass, for every head of both models: local heads concentrate their energy at fast rotations, far-field heads at near-zero rotations (black outlines mark the heads from (a) and (b)). (d) Attention of the Wan2.2 local head on a 128-token raster crop, recomputed from frequency subsets: the fastest third of each axis’s pairs reproduces the local pattern, the slowest third erases it. (e) The same test for the FLUX.2 local head; here the fast pairs are combined with the axes that FLUX.2 leaves unrotated for image tokens, which carry the head’s content gating.

rate at which the strongest non-self key is a neighbor. For FLUX.2, attention rows span the concatenated text and image tokens, and we additionally record the mass on text keys.

Frequency Bands and Head Selection. Within each positional axis we sort the channel pairs by rotation speed and split them into fastest, middle, and slowest thirds; for FLUX.2 the unrotated time- and text-axis channels form an additional static group. Band-restricted attention is obtained by recomputing the logits from a channel subset, keeping the $1/\sqrt{d}$ scale, and re-applying the softmax. For each model we select the eight heads with the highest local attention mass averaged over all 18 settings; the ablation numbers below average over these heads and settings.

Local Heads and Position Selection. Both models contain sharp *local heads*, the visual analogue of previous-token heads. In Wan2.2, layer 33, head 30 — at the same position in both denoising experts, whose top-eight local-head lists share seven of eight entries — attends to the immediately preceding token of the raster scan: its local attention mass is 0.99, and 98.1% of queries place their argmax exactly there. This rate equals the fraction of queries that do not start a row, *i.e.*, selection is exact up to grid boundaries. In FLUX.2, head 33 of layer 46, a late single-stream block, mirrors this with the immediately following token (98.4% of queries, again saturating the boundary limit; local attention mass 0.93). Each remains the top-ranked local head of its layer in all 18 settings, indicating that locality is a stable head property rather than an artifact of a particular input. The per-pair decomposition traces this selection to the fast end of the spectrum, exactly as in Qwen3-4B: we average each head’s pre-rotation queries and keys over all visual tokens and express every channel pair by its amplitude $\|q^{(i)}\| \|k^{(i)}\|$, phase, and frequency; the summed width-axis contributions peak at offset one (Figs. 21a and 21b), pairs on other axes contribute offset-independent constants, and the pairwise sum matches the directly rotated dot product to within 3×10^{-12} (Wan2.2) and 3×10^{-14} (FLUX.2) in float64. The band

ablations confirm the attribution. In Wan2.2, retaining only the fastest third of each axis’s pairs preserves the selected heads’ local attention mass (0.944 versus 0.967 in the low-noise expert; 0.934 versus 0.957 in the high-noise expert), while retaining only the slowest third collapses it to 0.002 (Fig. 21d). In FLUX.2, the unrotated channels select which tokens to attend to by content while the fast pairs place the peak at the target offset: attention recomputed from the two groups together matches the full heads almost exactly (cosine similarity 0.994; local mass 0.776 versus 0.926), whereas either group alone fails (0.190 from the fast pairs alone, 0.051 without them; Fig. 21e).

Far-Field Heads and Semantic Matching. The slowly rotating channels serve the opposite role. Using full per-token dumps at one setting per model, we measure each head’s per-pair energy $E_i = \mathbb{E}_t \|q_t^{(i)}\| \|k_t^{(i)}\|$ and summarize its spectrum by the energy-weighted geometric mean of the rotating pairs’ frequencies. Heads whose attention lies far outside the local neighborhood (far-field mass > 0.8 ; mean attended distance of ~ 20 tokens in Wan2.2 and ~ 30 in FLUX.2) concentrate their energy at near-zero rotation speeds: their median centroid is 0.012 rad/token in Wan2.2 and 0.013 in FLUX.2, more than $20\times$ slower than that of local heads (local mass > 0.5 ; medians 0.26 and 0.25, respectively), so their dot products turn by only a few degrees over the attended range (Fig. 21c). FLUX.2 makes the division explicit: its far-field heads park half of their energy (median 51%, versus 13% for local heads) in the unrotated channels, and the slow channels alone (the slowest third plus the unrotated group) reproduce their attention almost exactly (cosine similarity 0.965).