Qualcomm AI research

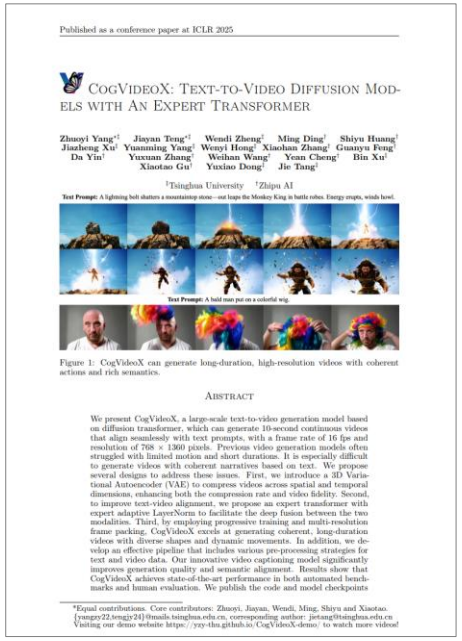
ICLR

Neodragon

Mobile Video Generation using Diffusion Transformer

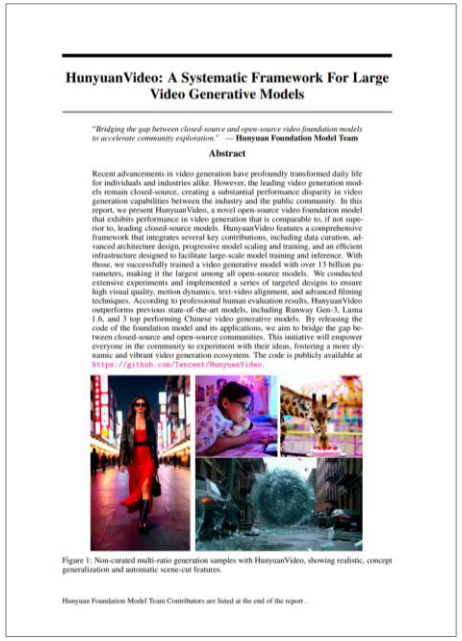
Animesh Karnewar, Denis Korzhenkov, Ioannis Lelekas,
Noor Fathima, Adil Karjauv, Mohsen Ghafoorian, Amirhossein Habibian

Frontier open-source VDMs (Video Diffusion Models)



CogVideoX: Text-to-Video Diffusion Models with An Expert Transformer

Cogvideo-X
Model-size: 2.0B
Vbench: **81.55**
Yang et al. 2025



HunyuanVideo: A Systematic Framework For Large Video Generative Models

HunyuanVideo
Model-size: 13.0B
Vbench: **85.09**
Kong et al. 2025

Pyramidal Flow Matching for Efficient Video Generative Modeling

Pyramidal-Flow
Model-size: 2.0B
Vbench: **81.56**
Jin et al. 2025

Wan: Open and Advanced Large-Scale Video Generative Models

Wan
Model-size: 1.3B
Vbench: **84.26**
Wan et al. 2025

LTX-Video: Realtime Video Latent Diffusion

LTX Video
Model-size: 2.0B
Vbench: **82.30**
HaCohen et al. 2024

Neodragon: Derivation of compute savings

Bidirectional Attention: $C_{\text{bi}} = (hwt)^2$ (1)

Causal Attention:
$$C_{\text{causal}} = \sum_{k=1}^{t} \underbrace{(h \cdot w)}_{\text{tokens in frame } k} \times \underbrace{(h \cdot w \cdot k)}_{\text{tokens in frames } 1..k} \tag{2}$$

$$= \sum_{k=1}^{t} (hw)^2 \cdot k \tag{3}$$

$$= (hw)^2 \sum_{k=1}^{t} k \tag{4}$$

$$= (hw)^2 \cdot \frac{t(t+1)}{2} \tag{5}$$

$$\textbf{Speedup}_{\textbf{causal}} = \frac{C_{\text{bi}}}{C_{\text{causal}}} = \frac{(hw)^2 t^2}{(hw)^2 \cdot \frac{t(t+1)}{2}} = \frac{2t}{t+1} \approx 2\times \text{ as } t \to \infty \tag{6}$$

since each 2× reduction per spatial dimension reduces the token count by a factor of 4. We refer to stage 0 as the highest (full-resolution) stage and stage $S-1$ as the lowest stage. For a query at frame k and a history frame $j \le k$, let the temporal distance be $d := k - j$. The number of tokens contributed by this particular history frame are:

$$T(d) = \begin{cases} M, & d = 0 \quad \text{(self)}, \\ \frac{M}{4^{d-1}}, & 1 \le d \le S-1, \\ \frac{M}{4^{S-1}}, & d \ge S. \end{cases}$$

Each query frame has M query tokens, so the dot-product cost contributed by a (k, j) pair is $M \cdot T(d)$. Summing over all ordered pairs (k, j) with $1 \le j \le k \le t$ is equivalent to summing over distances d and counting how many pairs have that distance: for a fixed d, there are exactly $(t-d)$ pairs (k, j) with $k - j = d$.

Total complexity (general S). Let $r = \frac{1}{4}$ be the token downsampling factor, and define the finite sums

$$A(S) := \sum_{m=0}^{S-2} r^m = \frac{1 - r^{S-1}}{1 - r} = \frac{4}{3}\left(1 - 4^{-(S-1)}\right), \qquad D(S) := \sum_{d=1}^{S-1} d\, r^{d-1} = \frac{1 - (S)r^{S-1} + (S-1)r^S}{(1-r)^2}.$$

With $u := t - S$ (and $t > S$ so $u \ge 1$), we obtain

$$C_{\text{pyr}}(t, S) = \sum_{k=1}^{t} \sum_{j=1}^{k} M \cdot T(k-j) = M^2 \left[\underbrace{t}_{\text{self } (d=0)} + \underbrace{\sum_{d=1}^{S-1} (t-d)\, r^{d-1}}_{\text{geometric ramp } (d=1..S-1)} + \underbrace{r^{S-1} \sum_{d=S}^{t-1} (t-d)}_{\text{bulk at lowest stage } (d \ge S)} \right]$$

$$= M^2 \left[t + t\,A(S) - D(S) + \frac{u(u+1)}{2 \cdot 4^{S-1}} \right]. \tag{7}$$

Asymptotically as $t \to \infty$ (with fixed S),

$$C_{\text{pyr}}(t, S) = \frac{M^2}{2 \cdot 4^{S-1}} t^2 + O(t), \qquad \Longrightarrow \qquad \text{Speedup}_{\text{pyr}}(S) = \frac{C_{\text{bi}}}{C_{\text{pyr}}(t, S)} \xrightarrow[t \to \infty]{} 2 \cdot 4^{S-1}. \tag{8}$$

Specialisation to $S = 3$ (matches Fig. 2). For $S = 3$, equation 8 becomes:

$$\textbf{Speedup}_{\textbf{temporal}} = \text{Speedup}_{\text{pyr}}(S = 3) = 2 \cdot 4^{(3-1)} = 32\times. \tag{9}$$

The 32× compute saving from the *Temporally Pyramidal Causal* attention is already a major boost, but the Pyramidal-Flow model goes further by also denoising each frame in a *spatial pyramid* (coarse-to-fine) fashion. This spatial pyramidal structure is orthogonal to the temporal pyramid and provides an additional speedup. We now derive this spatial speedup and then combine it with the temporal speedup to obtain the total compute savings over full bidirectional attention (see Fig. 2).

Spatial pyramid setup. Assume that the denoising process allocates fractions p_i of the total denoising steps to each stage, with $\sum_{i=0}^{S-1} p_i = 1$.

Per-frame cost scaling. At stage i, both the query and the effective K/V token counts scale by $1/4^i$ relative to full resolution. Since attention cost is bilinear in queries and keys, the per-frame cost at stage i scales as

$$\text{Cost factor at stage } i \;\propto\; \frac{1}{4^i} \cdot \frac{1}{4^i} = \frac{1}{16^i}.$$

Thus, if $C_{\text{temp}}^{(k)}$ denotes the per-frame cost under the *Temporally Pyramidal Causal* setup (with queries at full resolution), then the spatially adjusted per-frame cost is

$$C_{\text{spatial-temp}}^{(k)} = \left(\sum_{i=0}^{S-1} \frac{p_i}{16^i} \right) C_{\text{temp}}^{(k)} = \beta_S(\mathbf{p})\, C_{\text{temp}}^{(k)}, \qquad \beta_S(\mathbf{p}) := \sum_{i=0}^{S-1} \frac{p_i}{16^i}.$$

Spatial speedup. The relative compute multiplier in the spatial dimension is $\beta_S(\mathbf{p}) < 1$, so the speedup is

$$\boxed{\text{Speedup}_{\text{spatial}}(\mathbf{p}) = \frac{1}{\beta_S(\mathbf{p})}}.$$

Uniform allocation across stages. If denoising steps are split uniformly across stages, $p_i = \frac{1}{S}$, then

$$\beta_S = \frac{1}{S} \sum_{i=0}^{S-1} \frac{1}{16^i} = \frac{1}{S} \cdot \frac{1 - 16^{-S}}{1 - \frac{1}{16}} = \frac{16}{15S}\left(1 - 16^{-S}\right), \qquad \text{Speedup}_{\text{spatial}} = \frac{15S}{16(1 - 16^{-S})}.$$

For large S, this approaches $\beta_S \approx \frac{16}{15S}$ and $\text{Speedup}_{\text{spatial}} \approx \frac{15}{16} S$.

Specialisation to $S = 3$. With three spatial stages and uniform allocation $p_i = \frac{1}{3}$,

$$\beta_3 = \frac{1}{3}\left(1 + \frac{1}{16} + \frac{1}{256}\right) = \frac{273}{768} \approx 0.3555, \qquad \textbf{Speedup}_{\textbf{spatial}} = \frac{1}{\beta_3} = \frac{768}{273} \approx 2.81\times.$$

Combined spatio-temporal speedup. The temporal pyramid (with $S = 3$ stages) yields an asymptotic speedup of

$$\textbf{Speedup}_{\textbf{temporal}} \approx 32\times$$

relative to full bidirectional attention. The spatial pyramid (with $S = 3$ stages and uniform allocation) yields

$$\textbf{Speedup}_{\textbf{spatial}} \approx 2.81\times$$

relative to the temporal-only baseline. Since these optimisations act on orthogonal dimensions (temporal vs spatial), the combined speedup is multiplicative:

$$\boxed{\textbf{Speedup}_{\textbf{combined}} \approx 32 \times 2.81 \approx 90\times.}$$

Thus, a *Spatio-temporally Pyramidal Causal* latent generation setup can reduce the dominant attention complexity by nearly two orders of magnitude (90×) compared to a full-resolution, fully bidirectional attention. These efficiency gains are not merely theoretical; they enable practical scaling of autoregressive video diffusion to longer sequences and higher resolutions without prohibitive compute costs. For this reason,

Compute savings: **90x**

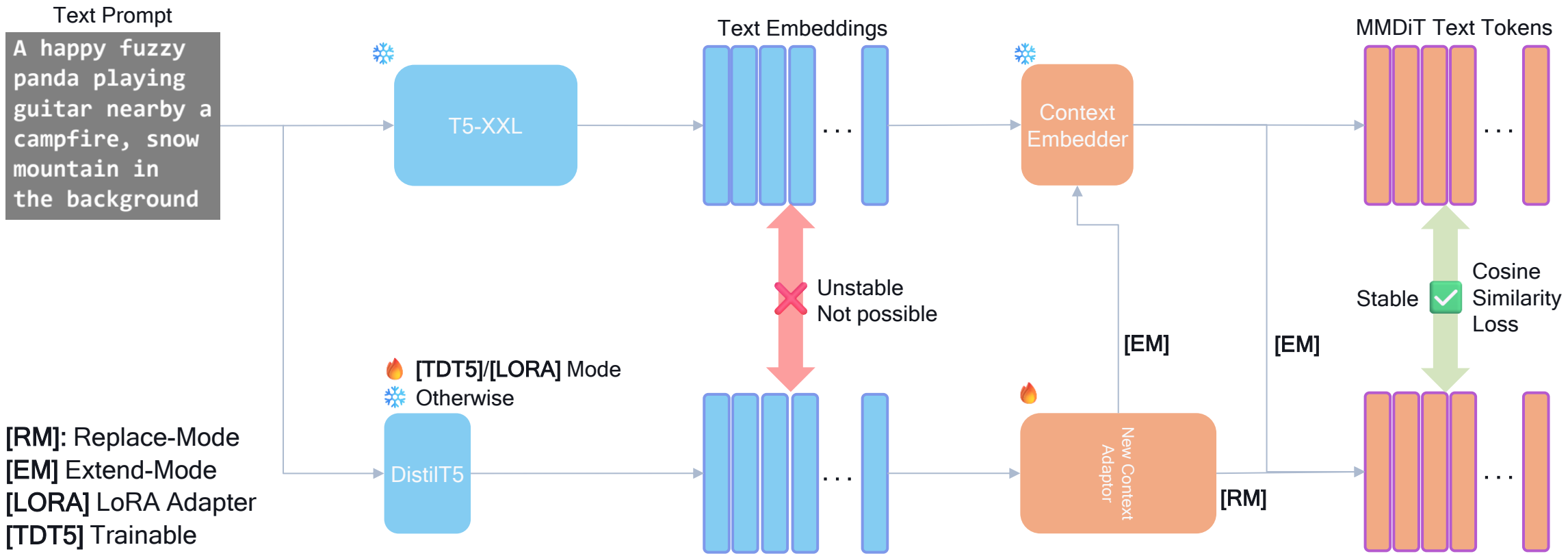
(Compared to vanilla Bidirectional DiT)

Neodragon

Optimisation 1:
Text-Encoder Distillation

Qualcomm
AI research

Neodragon: Text-Encoder Distillation framework



Qualitative Results

	Yoda playing guitar on the stage	A cat playing in park	A happy fuzzy panda playing guitar nearby a campfire, snow mountain in the background	A beautiful coastal beach in spring, waves lapping on sand, black and white
DT5 CA **[RM]** Replace				
DT5 CA **[EM]** Extend				
DT5 CA **[LORA]** LoRA				
[TDT5] Trainable *DT5*				

Text Encoder Distillation (Summary)

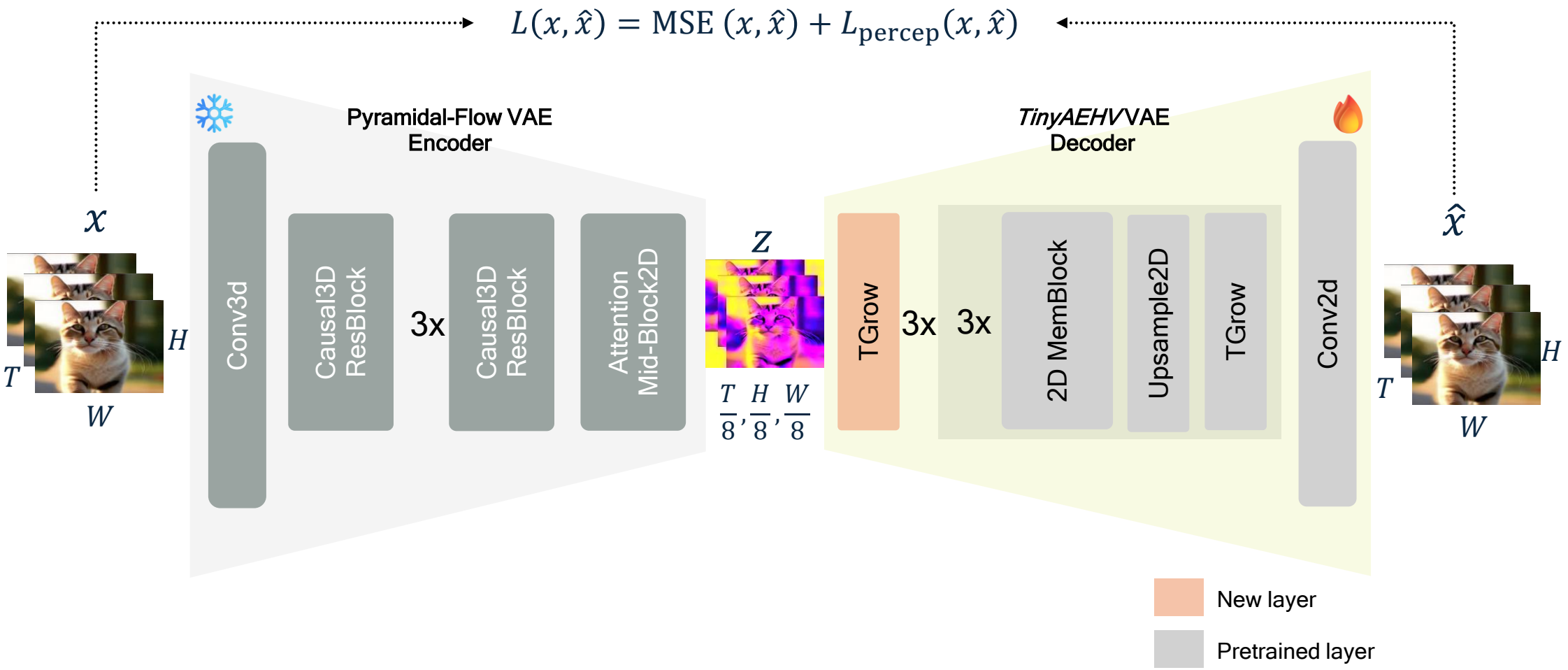
Method	#Params (↓) (TE+CA)	VBench		
		Total (↑)	Quality (↑)	Semantic (↑)
T5-XXL (Baseline)	4.732B	80.31	83.68	66.81
DT5 CA **[RM]** Replace	0.260B	79.64	83.71	63.39
DT5 CA **[EM]** Extend	0.266B	79.16	83.56	61.55
DT5 CA **[LORA]** LoRA	0.136B	64.74	74.94	24.08
[TDT5] Trainable *DT5*	0.136B	79.20	83.44	62.12

Neodragon

Optimisation 2:
Asymmetric Decoder Distillation

Qualcomm
AI research

Neodragon: Asymmetric Decoder Distillation

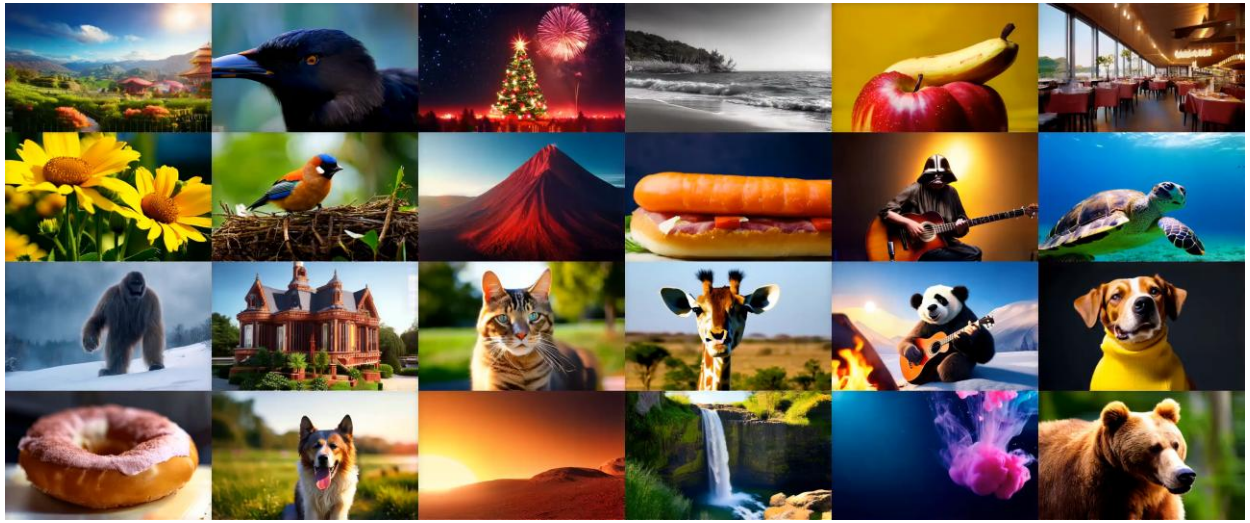


Asymmetric Decoder Distillation (Summary)

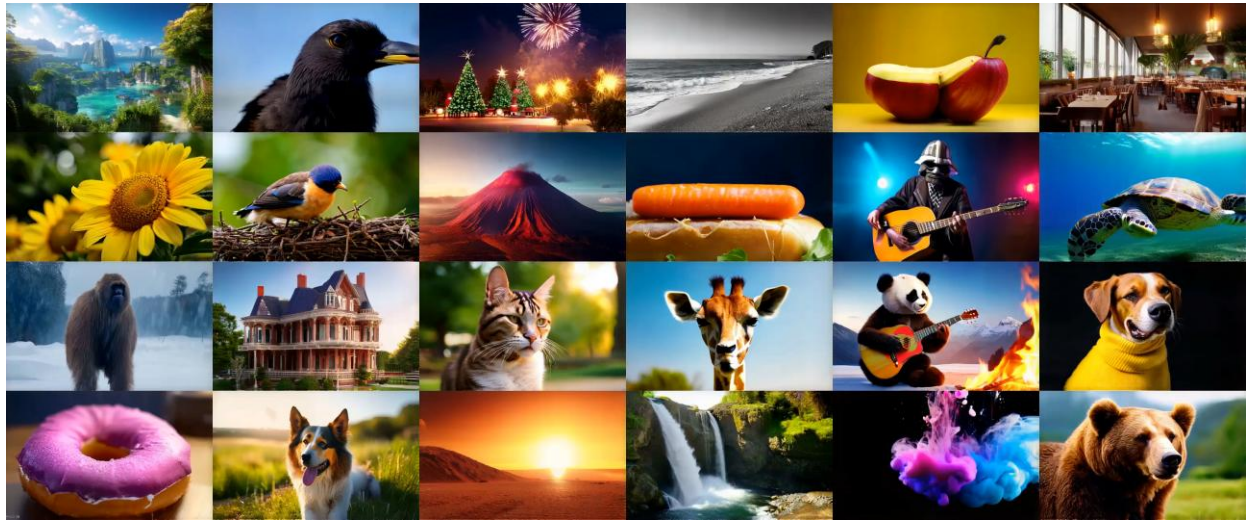
Method	#Params (↓) Decoder	GPU Latency (↓)	PSNR (↑) Orig. Enc.	PSNR (↑) Our Enc.	VBench		
					Total (↑)	Quality (↑)	Semantic (↑)
Pyramid-Flow Native Decoder	226M	2.496s	29.12	29.12	80.31	83.68	66.81
WAN modified	74M	1.666s	31.47	29.18	80.36	83.82	66.55
Cosmos CV [8x8x8]	63M	0.451s	29.34	29.45	79.96	83.37	66.35
LTX Video modified	237M	1.738s	28.34	29.46	80.34	83.75	66.68
(Our) TinyAEHV modified	10M	0.851s	27.71	28.40	80.25	83.51	67.19

Qualitative evaluation

Pyramidal-Flow



Neodragon



Neodragon

Optimisation 3:
MMDiT Block-Pruning

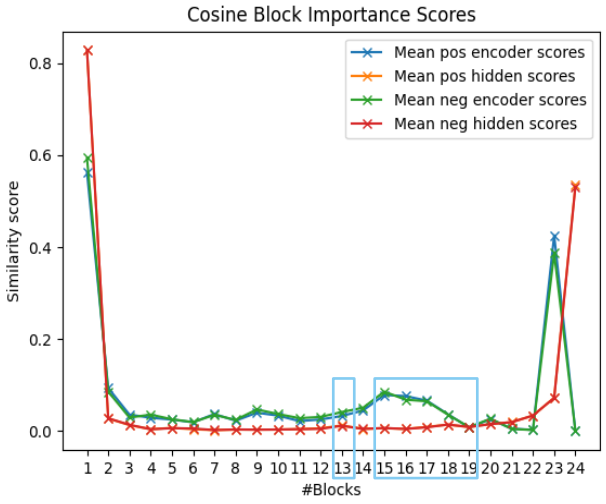
Qualcomm
AI research

Neodragon: MMDiT Block Pruning (Analysis)

Block Importance score

$$\mathrm{BI}_i = 1 - \mathbb{E}_{X,t} \frac{\mathbf{X}_{i,t}^T \mathbf{X}_{i+1,t}}{\|\mathbf{X}_{i,t}\|_2 \, \|\mathbf{X}_{i+1,t}\|_2}$$

Blockwise scores plot for 24 MMDiT blocks




No-prune


Block 1


Block 2


Block 3


Block 4


Block 5


Block 6


Block 7


Block 8


Block 9


Block 10


Block 11


Block 12


Block 13


Block 14


Block 15


Block 16


Block 17


Block 18


Block 19

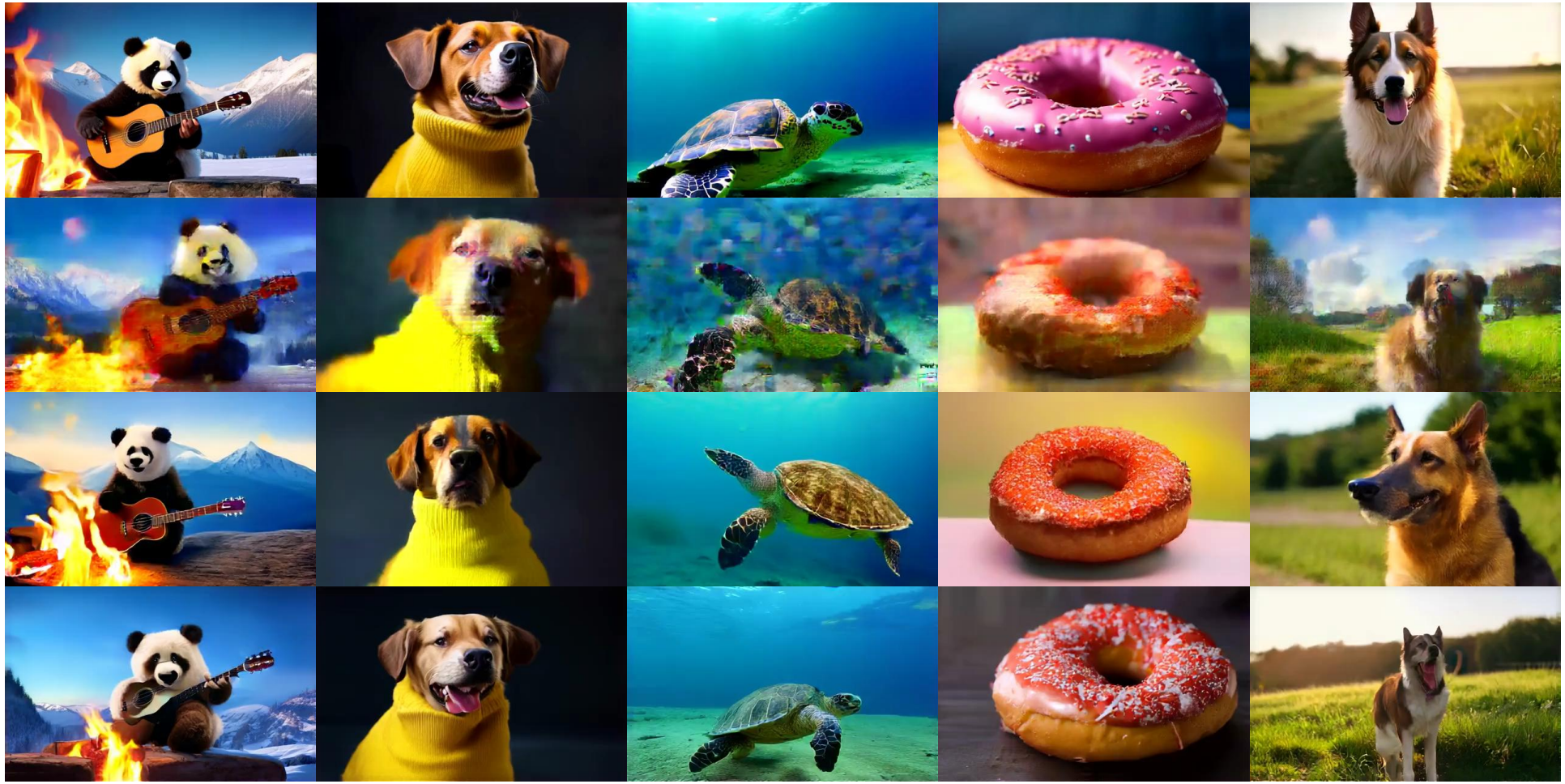

Block 20


Block 21


Block 22


Block 23

Neodragon: MMDiT Block Pruning (Finetuning)



24 blocks Model
[Baseline]
Vbench: 80.31

18 blocks model
No Finetuning
Vbench: N/A

18 blocks model
Stage-1 Fine tuning
Vbench: 78.39

18 blocks model
Stage-2 Fine tuning
Vbench: 80.21

Block Pruning (Summary)

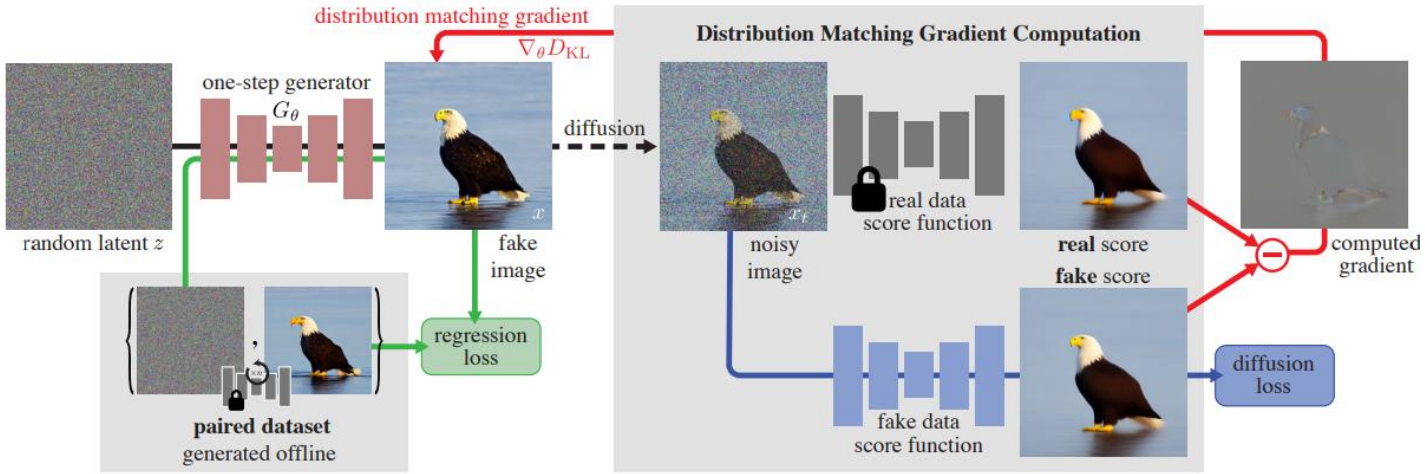
Method	#Params (↓)	GPU Latency (↓)	FLOPS /frame	VBench		
				Total (↑)	Quality (↑)	Semantic (↑)
Pyramid-Flow native 24 blocks model	2.028B	20.40s	55.99T	80.31	83.68	66.81
Stage-1						
18-blocks model	1.518B	17.53s	41.94T	78.39	81.58	65.63
16-blocks model	1.348B	15.85s	37.25T	74.59	78.74	57.99
Stage-2 ***(Shipped Models)***						
18-blocks model	1.518B	17.53s	41.94T	80.21	83.54	66.90
16-blocks model	1.348B	15.85s	37.25T	78.56	82.37	63.32

Neodragon

Optimisation 4:
Step Distillation

Qualcomm
AI research

Neodragon: Pyramidal DMD Step-Distillation



Distribution Matching Distillation (DMD)
Yin et al., 2024

Pyramidal-Flow decomposes the probability flow into S stages, where the i^{th} stage operates at $2^i\times$ smaller resolution than the original, where $i \in \{0, \dots, S-1\}$. Let $\mathrm{Down}(\cdot, s)$ and $\mathrm{Up}(\cdot, s)$ denote spatial downsampling and upsampling by a factor s, respectively. Each stage is parameterised by a pair of noise levels $(\sigma^i_{\text{start}}, \sigma^i_{\text{end}})$ with $1 > \sigma^i_{\text{start}} > \sigma^i_{\text{end}} > 0$, and operates on latents at resolution $\mathrm{Down}(\boldsymbol{z}, 2^i)$. The start and end distributions for stage i are defined as

$$\tilde{\boldsymbol{z}}_{\sigma^i_{\text{start}}} := (1-\sigma^i_{\text{start}})\ \mathrm{Up}\big(\mathrm{Down}(\boldsymbol{z}, 2^{i+1}), 2\big) + \sigma^i_{\text{start}}\,\epsilon, \tag{6}$$

$$\tilde{\boldsymbol{z}}_{\sigma^i_{\text{end}}} := (1-\sigma^i_{\text{end}})\ \mathrm{Down}(\boldsymbol{z}, 2^i) + \sigma^i_{\text{end}}\,\epsilon. \tag{7}$$

A different local noise-level $\sigma^i_{\text{local}} \sim \mathbb{U}(0,1)$ is used to learn the Flow-Matching model at the i^{th} stage, and the global noise-level σ relates to the local noise-level σ^i_{local} as $\sigma = (1-\sigma^i_{\text{local}})\sigma^i_{\text{end}} + \sigma^i_{\text{local}}\sigma^i_{\text{start}}$. Thus, the overall Pyramidal Flow Matching objective is an aggregate over the stagewise objectives: $\mathcal{L}_{\text{pyr-FM}} := \sum_{i=0}^{S-1} \mathcal{L}^i_{\text{FM}}$. Appendix section D provides more details. For the *Pyramidal*-DMD, at i^{th} stage input $\tilde{\boldsymbol{z}}_\sigma = (1-\sigma^i_{\text{local}})\tilde{\boldsymbol{z}}_{\sigma^i_{\text{end}}} + \sigma^i_{\text{local}}\tilde{\boldsymbol{z}}_{\sigma^i_{\text{start}}}$, the to-be-learned Student model $\mathcal{D}_\theta$ aims to predict the clean latent, parametrized as a single-step Euler solver $\tilde{\boldsymbol{z}}_\theta := \tilde{\boldsymbol{z}}_\sigma - (\sigma/(\sigma^i_{\text{start}} - \sigma^i_{\text{end}}))\mathcal{D}_\theta(\tilde{\boldsymbol{z}}_\sigma, \sigma)$, since the teacher Score-Model $\mathcal{D}$ had been trained to approximate the flow defined as a derivative w.r.t. σ^i_{local}. The so-called Fake-Score-Model $\mathcal{D}_\varphi$ is trained with pyramidal Flow Matching objective $\mathcal{L}_{\text{pyr-FM}}$ but on the distribution of student-predicted clean latents instead of ground-truth video latents. Having the fake model, the student network is updated with DMD loss defined through its gradient $\nabla_\theta L^i_{\text{DMD}} \propto (\mathcal{D}(\tilde{\boldsymbol{z}}_\tau, \tau) - \mathcal{D}_\varphi(\tilde{\boldsymbol{z}}_\tau, \tau)) \cdot \nabla_\theta \tilde{\boldsymbol{z}}_\theta$. The input of teacher and fake model $\tilde{\boldsymbol{z}}_\tau$ is defined as a stage-wise noisy version of student-predicted clean latent, similar to eq. 6 and eq. 7,

Figure 5: Qualitative evaluation of step distillation. We visualise randomly selected frames from the generated `[49×320×512]` videos, across different step distillation application on the block-pruned model for 4-4-4 configuration.

$$\tilde{y}_{\sigma^i_{\text{start}}} := (1-\sigma^i_{\text{start}})\ \mathrm{Up}\big(\mathrm{Down}(\tilde{\boldsymbol{z}}_\theta, 2), 2\big) + \sigma^i_{\text{start}}\,\varepsilon, \tag{8}$$

$$\tilde{y}_{\sigma^i_{\text{end}}} := (1-\sigma^i_{\text{end}})\,\tilde{\boldsymbol{z}}_\theta + \sigma^i_{\text{end}}\,\varepsilon, \tag{9}$$

$$\tilde{\boldsymbol{z}}_\tau := (1-\tau^i_{\text{local}})\tilde{y}_{\sigma^i_{\text{end}}} + \tau^i_{\text{local}}\tilde{y}_{\sigma^i_{\text{start}}}, \tag{10}$$

where $\varepsilon \sim \mathcal{N}(0, \mathbb{I})$ and $\tau = (1-\tau^i_{\text{local}})\sigma^i_{\text{end}} + \tau^i_{\text{local}}\sigma^i_{\text{start}}$. We follow Yin et al. (2024) and define the sample-specific weight of DMD loss as $\left\|\mathcal{D}(\tilde{\boldsymbol{z}}_\tau, \tau) - \left(\tilde{y}_{\sigma^i_{\text{start}}} - \tilde{y}_{\sigma^i_{\text{end}}}\right)\right\|_1^{-1}$. Therefore, the sample gets higher weight, if teacher model is capable to estimate its conditional flow with a smaller error. In addition we found the supervised Cauchy loss $\mathcal{L}_{\text{teacher}} = \log\left(1 + \left\|\tilde{\boldsymbol{z}}_\theta - \mathrm{Down}(\boldsymbol{z}, 2^i)\right\|_2^2\right)$ useful for visual quality and used it with weight 0.5. During training, we update the student and fake model in alternate manner: one update of θ per two updates of φ. For student's updates we limit the set of local noise levels τ^i_{local} to four evenly selected values, and for fake model it is sampled from $\mathbb{U}(0,1)$. To obtain teacher's prediction we employ classifier-free guidance with the same hyperparameters as those recommended for Pyramidal-Flow.

Pyramidal Step-Distillation Results

Optimised Pyramidal DMD paired with SSD-1B first frame generator

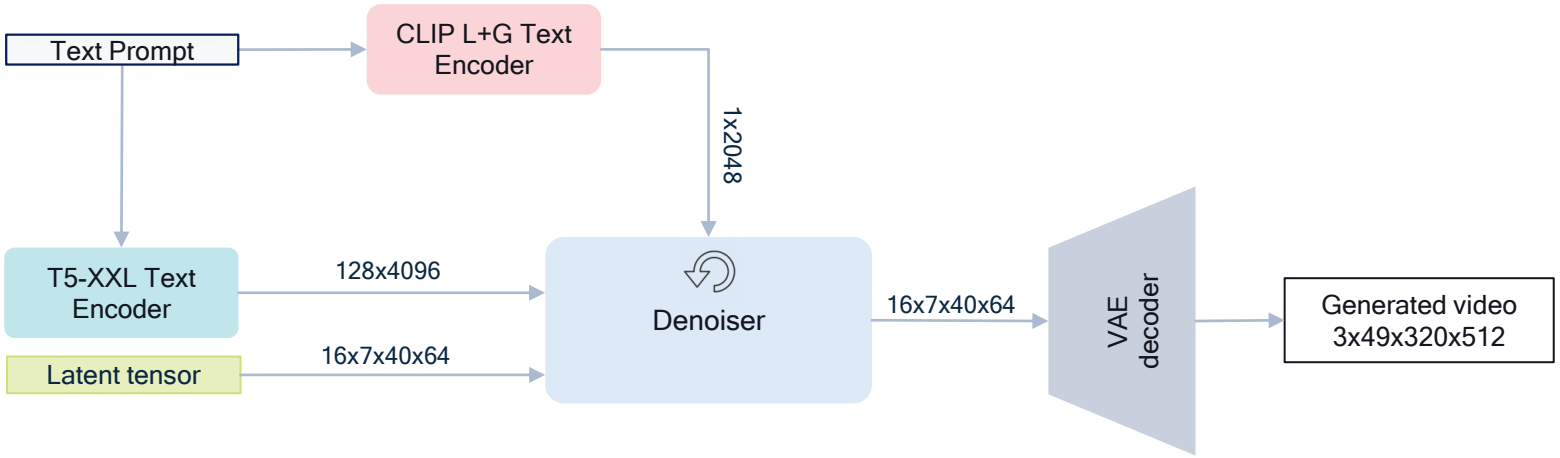
Vbench: 81.61

Neodragon

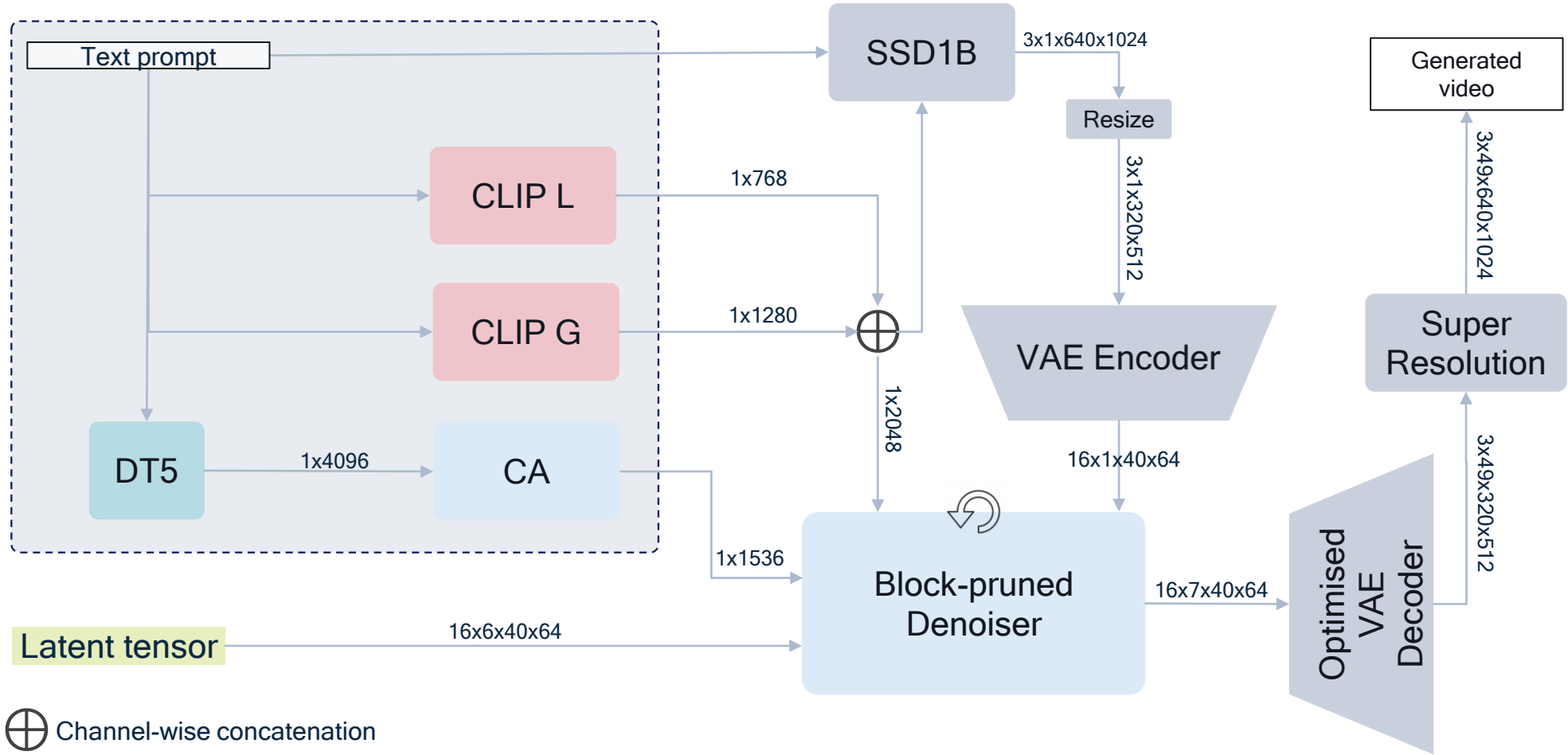
E2E Integration

Qualcomm AI research

Pyramidal-Flow Inference Pipeline

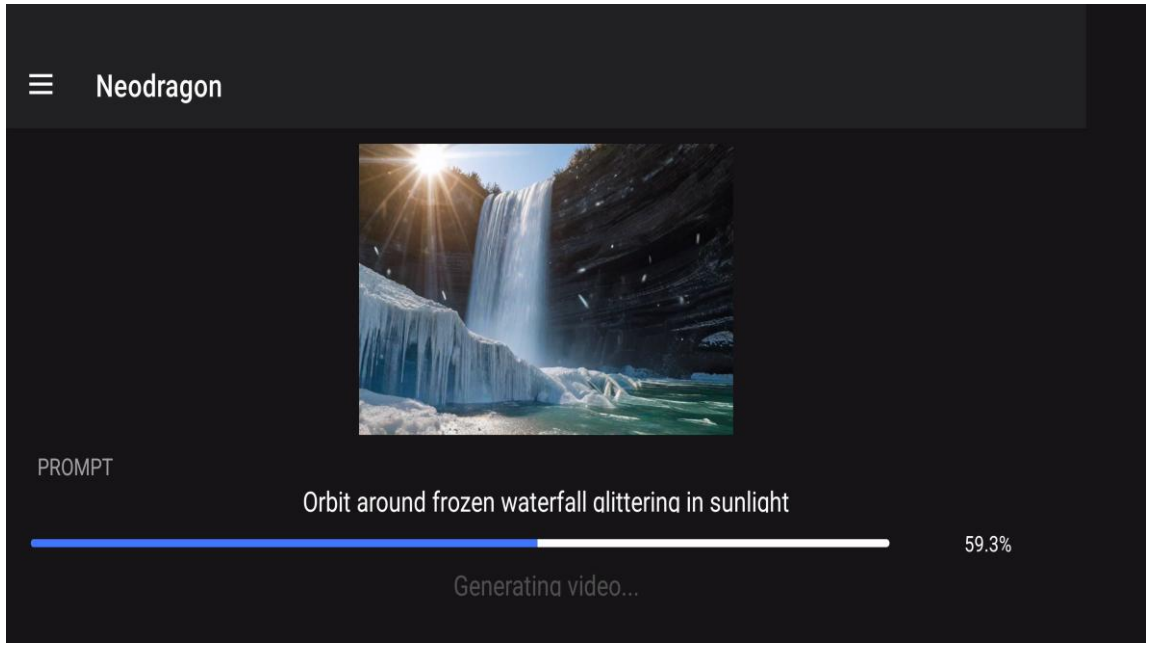


Neodragon: Full E2E Pipeline

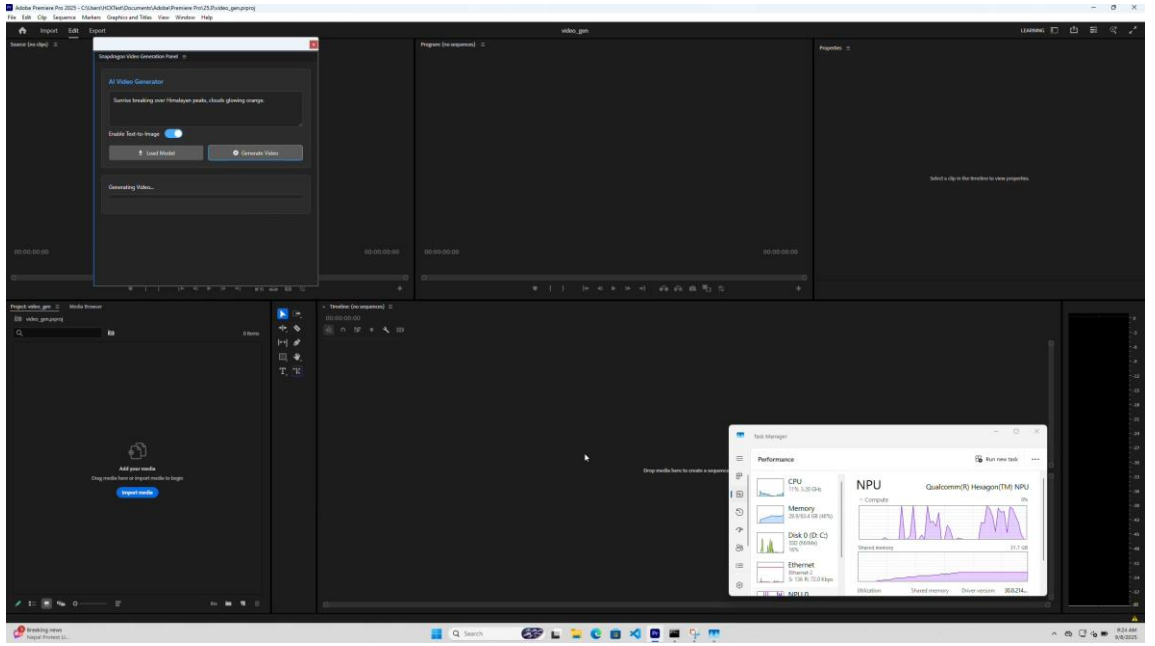


Neodragon: Demos

Snapdragon 8 Elite
Android mobile app



Snapdragon X Elite
Adobe PremierePro
Plugin



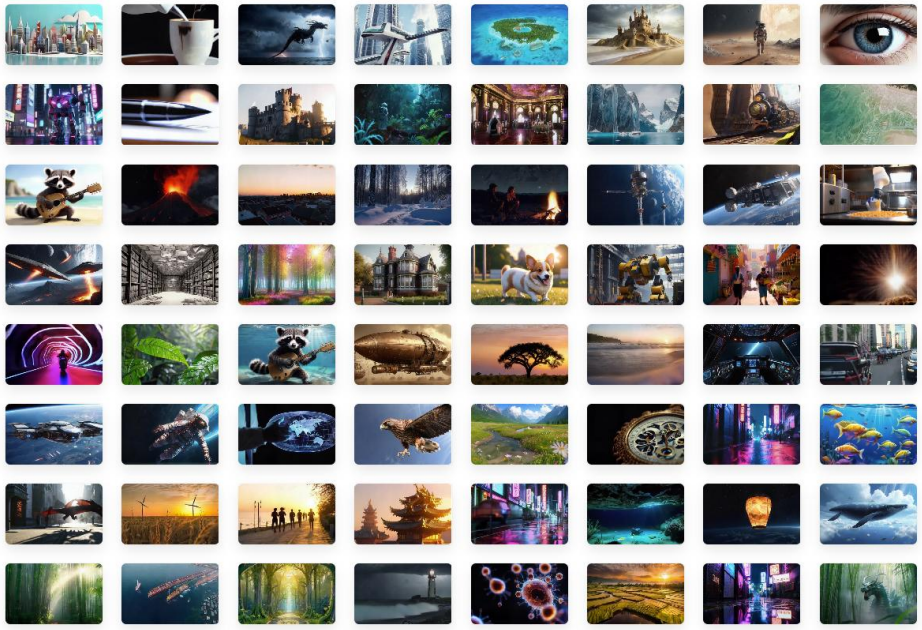
THANK YOU!

https://qualcomm-ai-research.github.io/neodragon

Neodragon: Mobile Video Generation using Diffusion Transformer

Animesh Karnewar, Denis Korzhenkov, Ioannis Lelekas, Noor Fathima Adil Karjauv, Hanwen Xiong, Vancheeswaran Vaidyanathan, Will Zeng, Rafael Esteves, Tushar Singhal, Fatih Porikli, Mohsen Ghafoorian, Amirhossein Habibian
Qualcomm AI Research

arXiv | Code (coming soon) | Checkpoint (coming soon)



Animesh Karnewar PhD
@AnimeshKarnewar

AI video generation is poised to be the next revolution, but its heavy computational demands limit real-world deployment. Excited to share Neodragon, my first project after the PhD — a significant step toward efficient, on-device video generation. Webpage: qualcomm-ai-research.github.io/neodragon

9:09 PM · Nov 11, 2025 · **37.7K** Views

View post engagements

3 | 17 | 137 | 120

Quotes | Reposts | Likes

Justin Johnson @jcjohnss
Cofounder @theworldlabs. Building Spatial Intelligence.
Following

Minjune Song @minjunesh · Nov 15
Wow quality looks great
2

Arrogant Bill @ai_codedream · Nov 16
Neodragon is a breakthrough in private, on-device AI video creation. Love how this opens up high-quality content for everyone-can't wait to see creators and users run with it.
1 | 116

Julien Blanchon @JulienBlanchon · Nov 14
Omg I was waiting for exactly that ! We need more on device generation research and projects
2 | 158

Cris Lenta @crislenta · Nov 14
whaaaat? no way this is too good to be true, right?!
1

Matt @Matthewagi · Nov 14
going the way of photos

Yash Bhalgat @ysbhalgat · Nov 14
Nice work, Animesh! Didn't know you joined @amir_habibian's team :)
1 | 1 | 1.4K

Animesh Karnewar PhD @AnimeshKarnewar · Nov 14
Thanks Yash! Haha, yeah I have been away from Social Media after my PhD, throughout the project.