DeepMind

Generating Images with Sparse Representations

Charlie Nash, Jacob Menick, Sander Dieleman, Peter Battaglia

Autoregressive Generative Image Models

Autoregressive generative image models generate images one pixel at a time.

The best models demonstrate exceptional

- Coherence
- Diversity

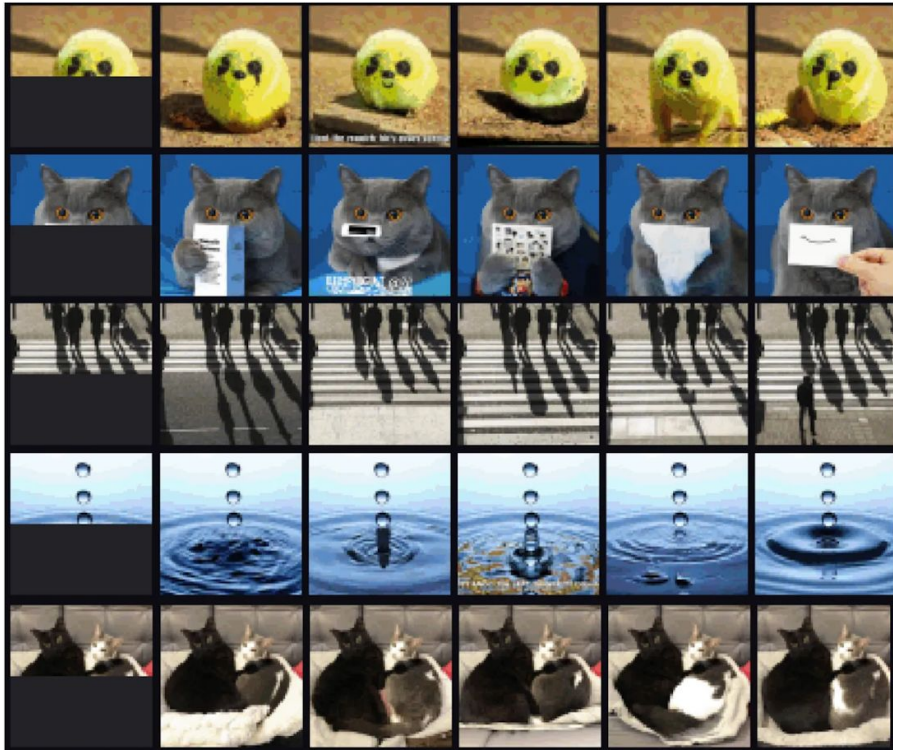


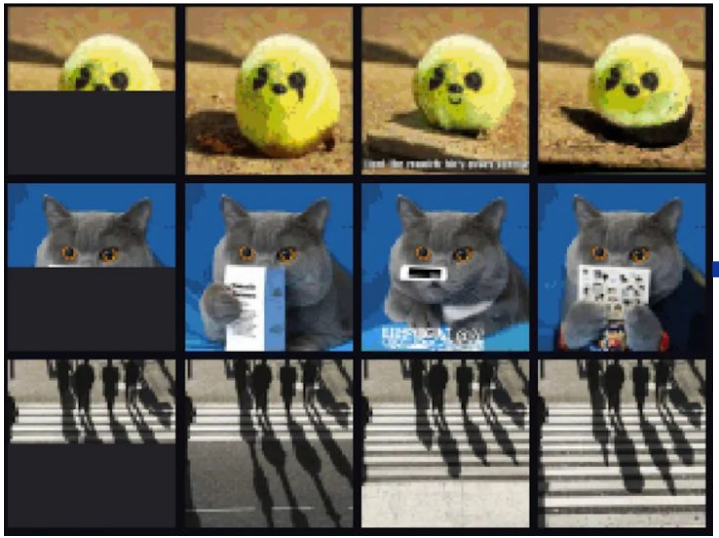
Image GPT

Generative Pretraining from Pixels,

Chen et al, 2020

Modelling high-dimensional images

- How to scale to **high resolution** images and video?
- Retain iGPTs extraordinary **coherence** and **diversity**
- Challenging to scale **Autoregressive Transformers** to very long sequences



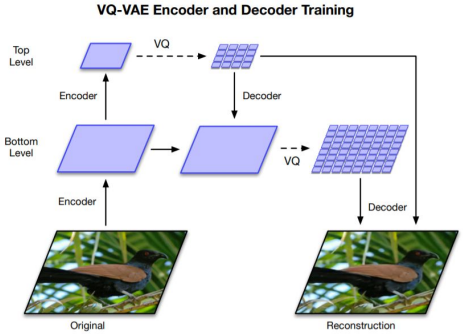
64x64 Images

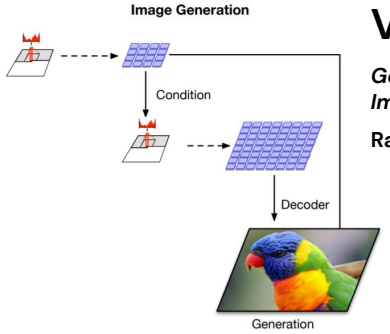
1024x1024 ++ Images

Existing approach: Learning and modelling compressed codes

VQ-VAE style models use:

- Learned neural encoder / decoder
- **Vector quantization** to obtain discrete codes
- **Autoregressive models** trained post-hoc

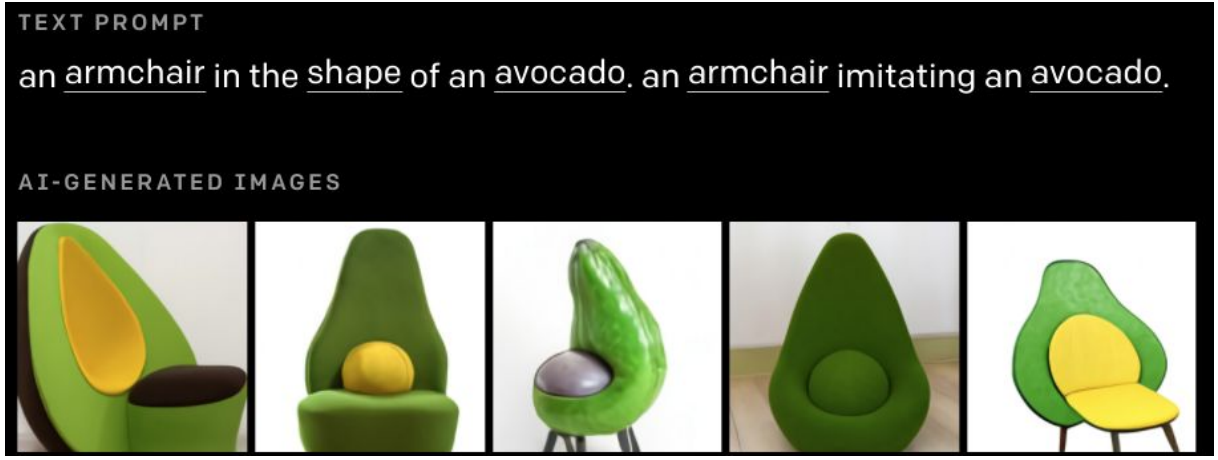




VQ-VAE-2

Generating Diverse High-Fidelity Images with VQ-VAE-2,

Razavi et al, 2020



DALL-E

Zero-Shot Text-to-Image Generation

Ramesh et al, 2021

Alternative: JPEG-style quantized DCT representations

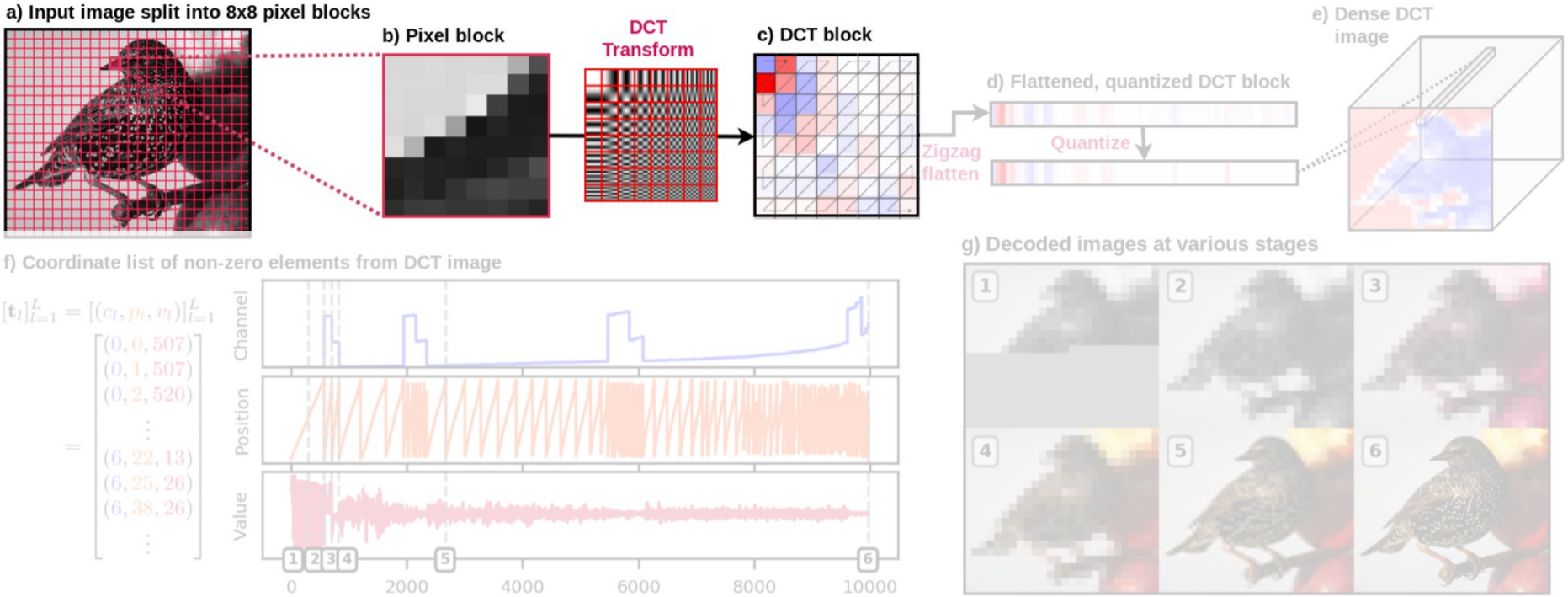
Do we need to do neural lossy compression?

Why not use **existing lossy codecs** as a way to represent high-dimensional data

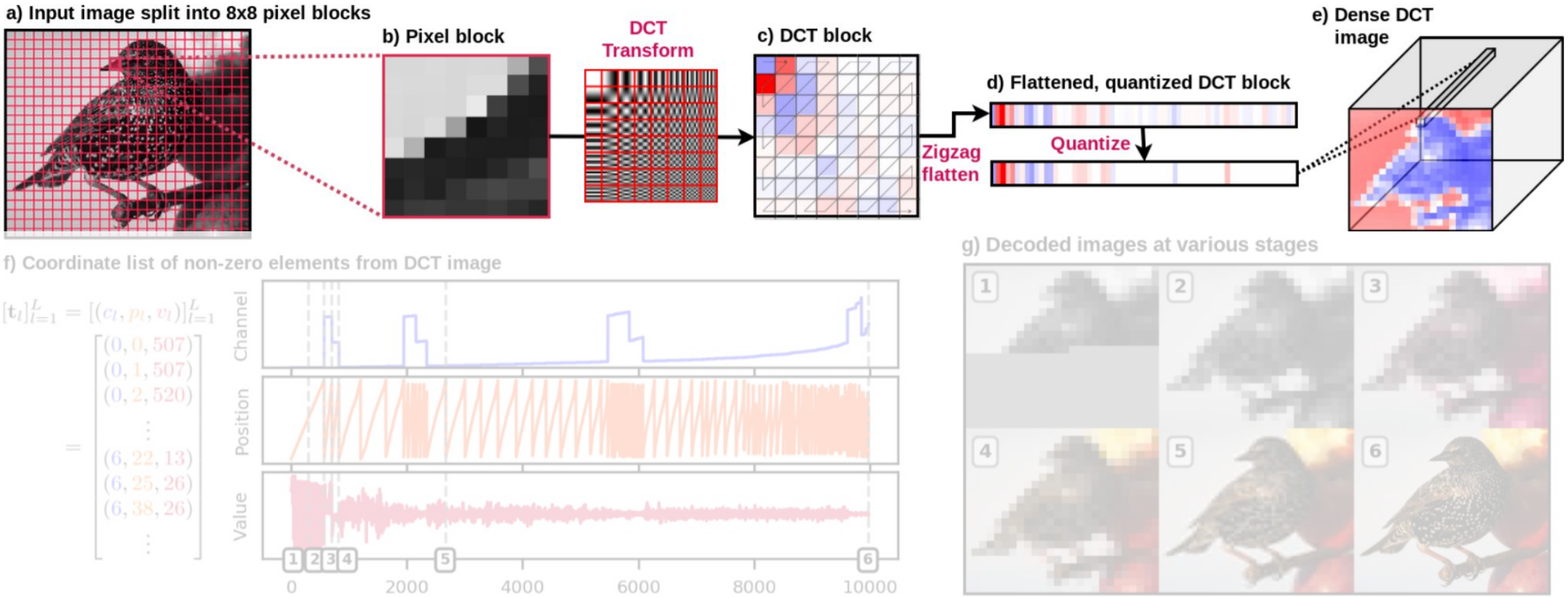
JPEG is a simple and ubiquitous codec based on the **discrete cosine transform (DCT)**

***Felis_silvestris_silvestris.jpg*:**
Michael Gäbler

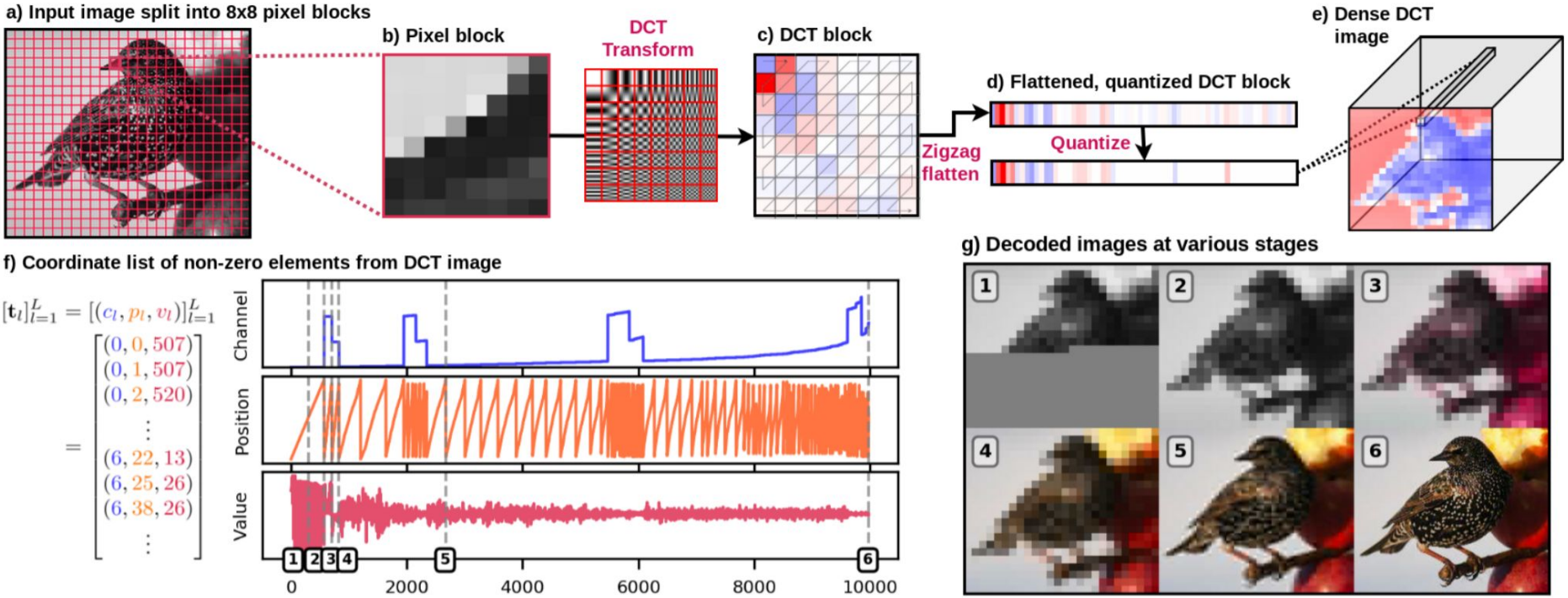
DCT-based sparse representations



DCT-based sparse representations



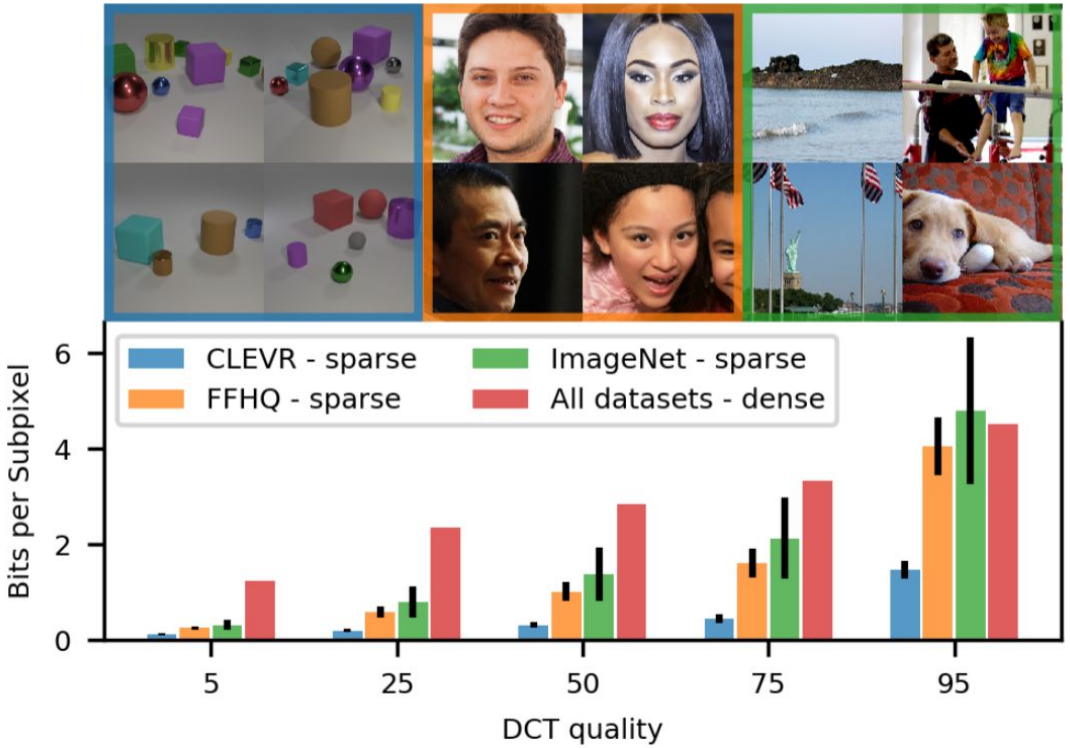
Alternative to VQ: DCT-based sparse representations



Sparse structure

Images often have **sparse structure**, with salient content **distributed unevenly** across locations

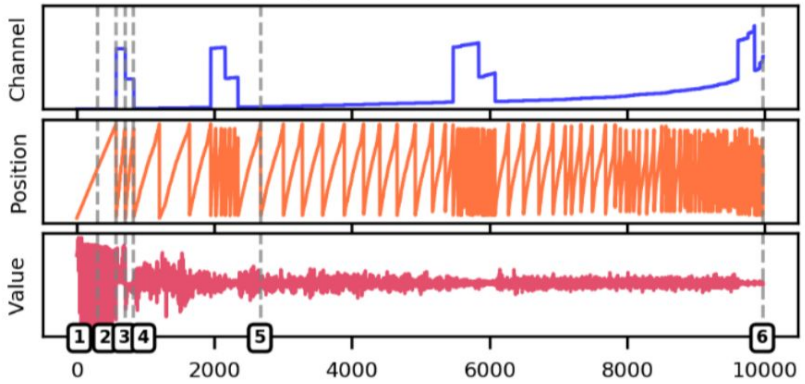
Sparse representations can encode the same data more compactly



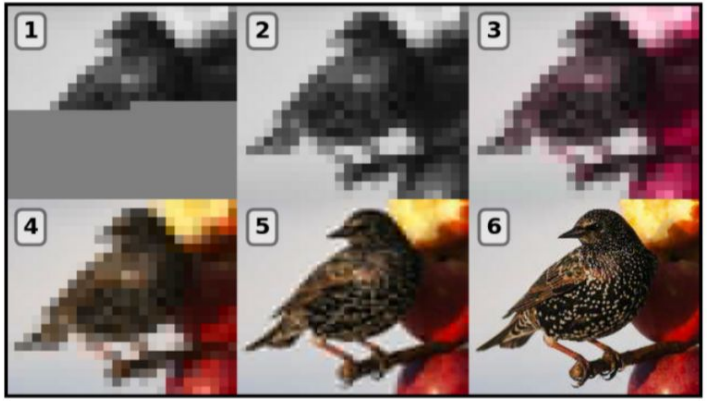
Colorization and Upsampling

f) Coordinate list of non-zero elements from DCT image

$$[\mathbf{t}_l]_{l=1}^{L} = [(c_l, p_l, v_l)]_{l=1}^{L} = \begin{bmatrix} (0,0,507) \\ (0,1,507) \\ (0,2,520) \\ \vdots \\ (6,22,13) \\ (6,25,26) \\ (6,38,26) \\ \vdots \end{bmatrix}$$



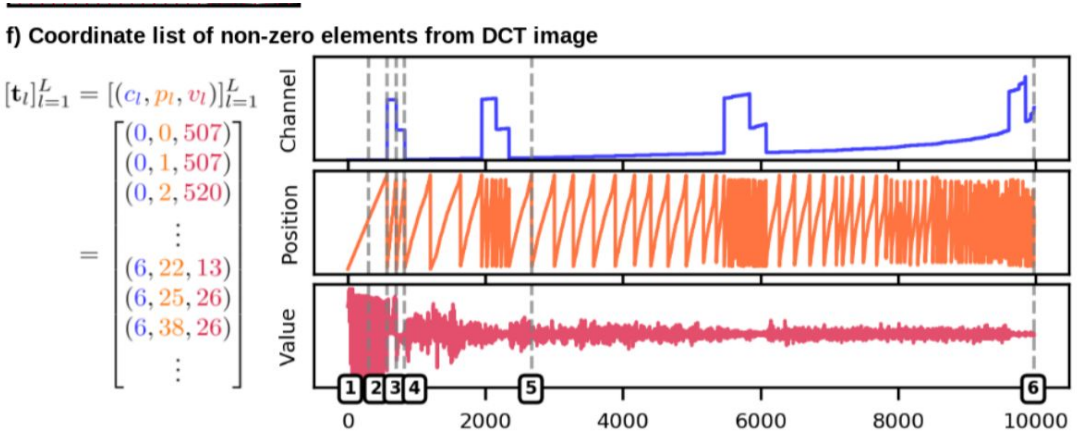
g) Decoded images at various stages



Perform **conditional generation** by treating **LHS** of the sequence as **input**

- Conditioning on the **first DCT component** is akin to **upsampling** by a factor of the block size (8x in most of our experiments)
- Can re-order sparse DCT sequence to place **luma components before chroma**. Conditioning on the luma components yields a **colorization** model.

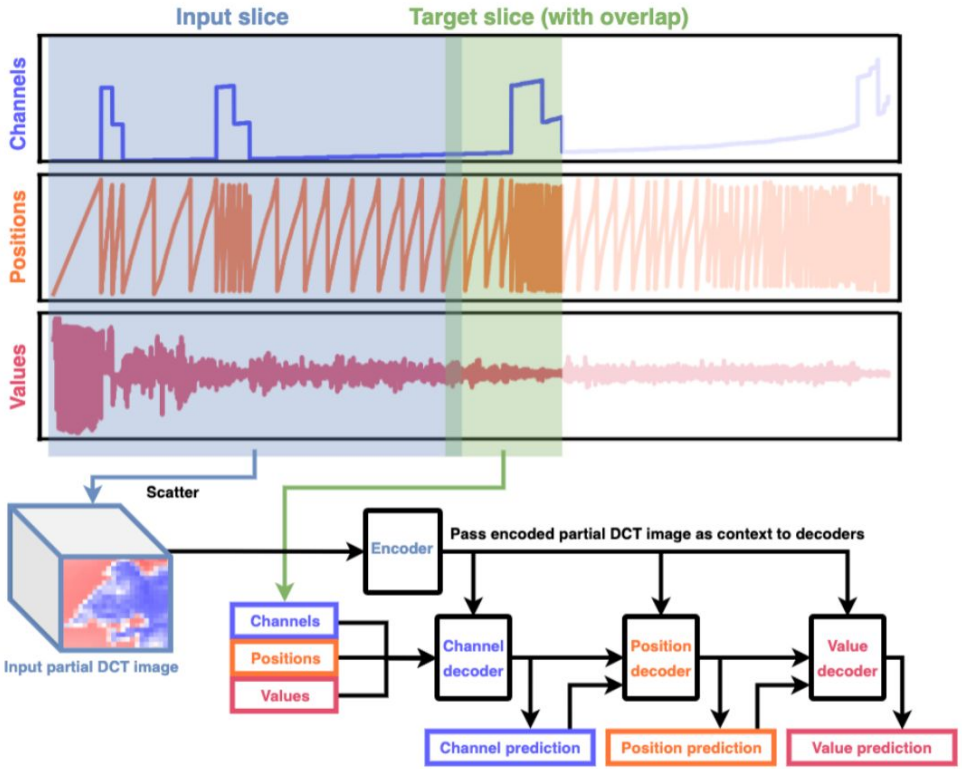
DCTransformer



- We model the sparse DCT sequence **autoregressively**
- Given previous elements, predict the next **DCT channel**, **spatial location**, and **DCT value** using categorical (softmax) distributions

$$\prod_{l=1}^{L} p(c_l|\mathbf{t}_{<l};\theta)p(p_l|c_l,\mathbf{t}_{<l};\theta)p(v_l|c_l,p_l,\mathbf{t}_{<l};\theta)$$

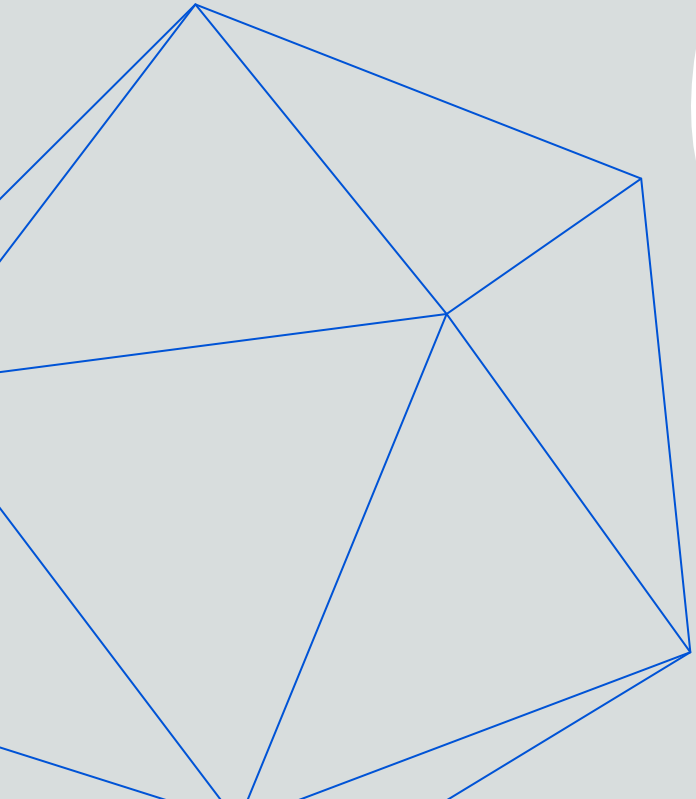
DCTransformer: Chunk based training



- A **target chunk** is selected during training, and previous elements are collected in an **input slice**.
- The **input slice** is **gathered** into a 3D **DCT image Tensor**
- The DCT-image is **encoded using a Transformer encoder**, and the target slice is passed through a series of Transformer decoders to predict **channels**, **positions** and **values**.

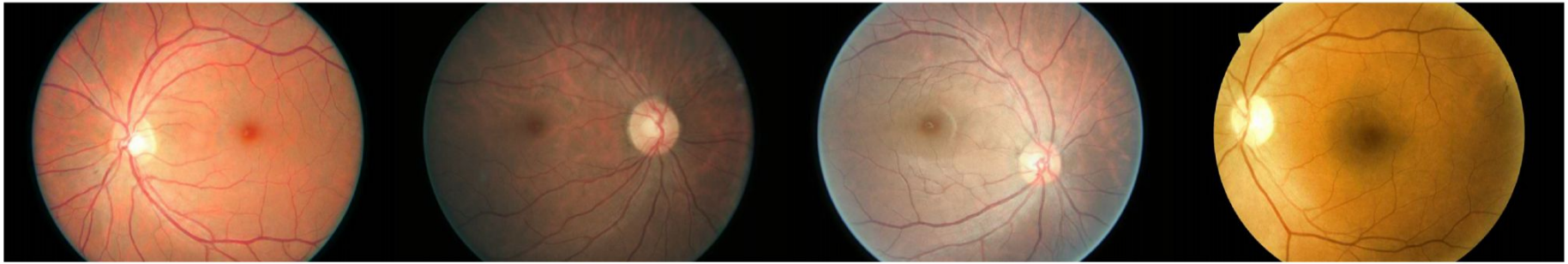
DeepMind

Results



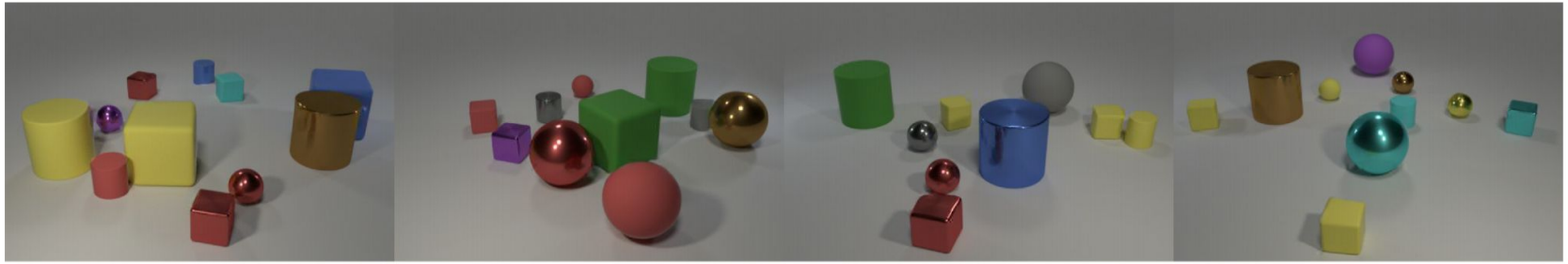
Samples (cherrypicked)

(a) **Plant leaves** Long-side resolution 2048, block-size 32 and DCT quality 50. Available at https://www.tensorflow.org/datasets/catalog/plant_leaves



(b) **Diabetic retinopathy** Long-side resolution 1024, block-size 16 and DCT quality 75. Available at https://www.tensorflow.org/datasets/catalog/diabetic_retinopathy_detection

Samples (cherrypicked)



(c) **CLEVR** Long-side resolution 480, block-size 8 and DCT quality 90. Available at https://www.tensorflow.org/datasets/catalog/clevr

(d) **FFHQ** Long-side resolution 1024, block-size 16 and DCT quality 35 Available at https://github.com/NVlabs/ffhq-dataset

Class-conditional ImageNet samples (cherrypicked)

Sampling sparsity

DCTransformer predicts *where* to place content, as well as *what* content to add

Heatmaps (right) show the distribution of sampled image locations

Model samples

Position samples heatmap

ImageNet samples comparison: Cheeseburger

Data

DCTransformer

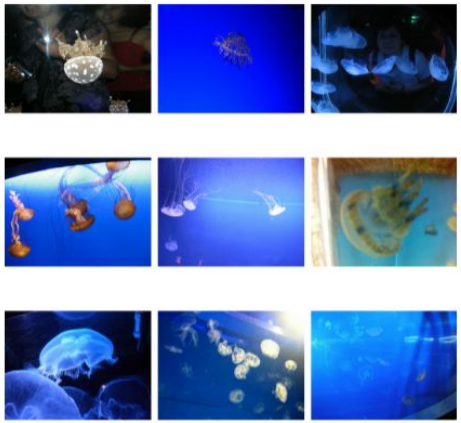
BigGAN

VQ-VAE2

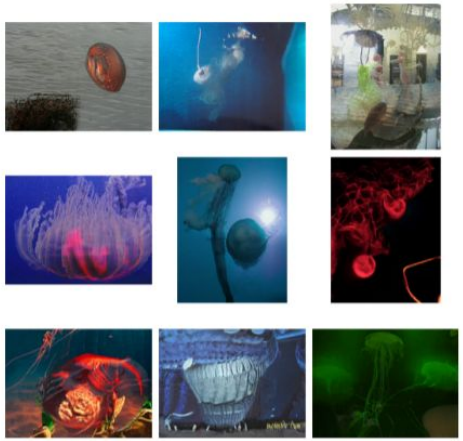
Qualitatively we find **DCTransformer** samples are:

- Detailed and diverse
- Less reliably coherent than **BigGAN**
- Similar to **VQ-VAE2** but sharper due to reduced lossy compression

ImageNet samples comparison: Jellyfish



Data

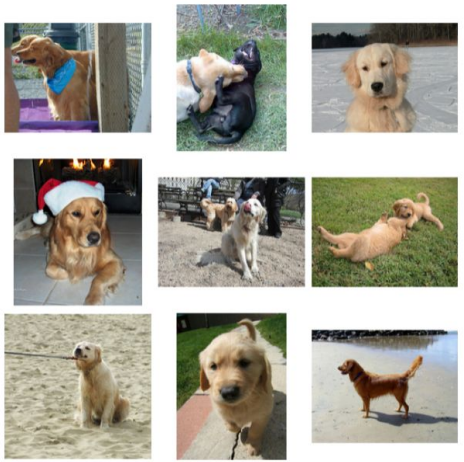


DCTransformer

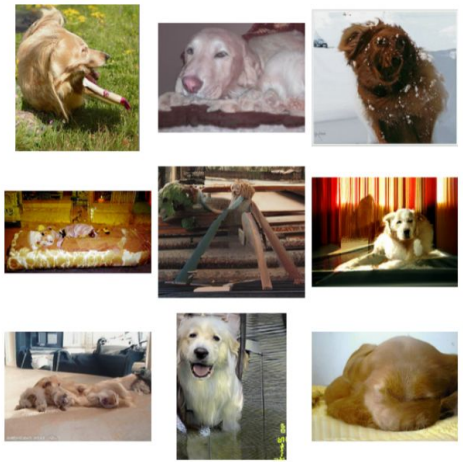
BigGAN

VQ-VAE2

ImageNet samples comparison: Retriever



Data



DCTransformer

BigGAN

VQ-VAE2

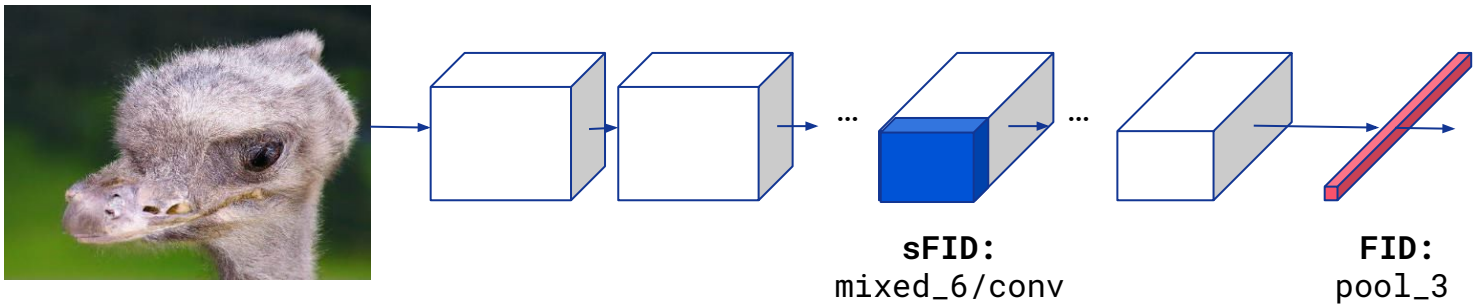
Sample metrics: FID and sFID

- FID is a **sample-based metric** for evaluating generative models
- Fit Gaussian distribution to **Inception image features** on real and fake images
- Compare distributions using **Frechet distance:**

$$\mathrm{FID} = |\mu - \mu_w|^2 + \mathrm{tr}(\Sigma + \Sigma_w - 2(\Sigma\Sigma_w)^{1/2})$$

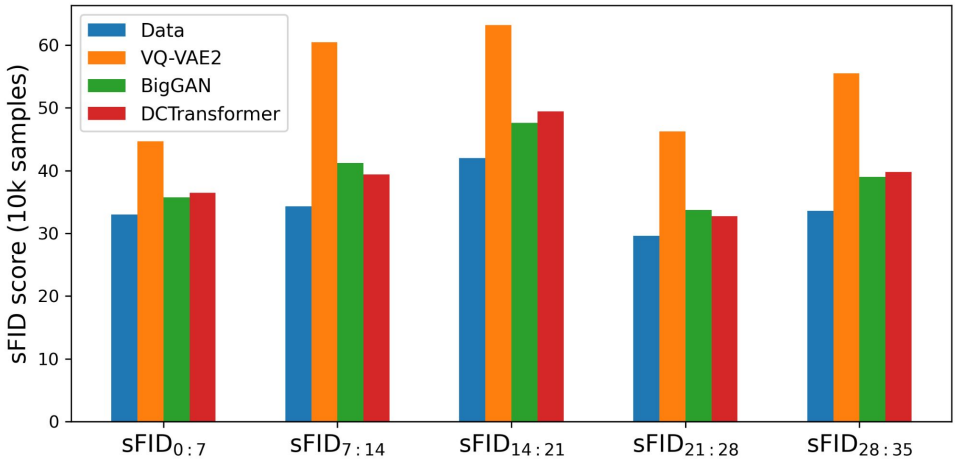
- **sFID** uses mid-level Inception features, which are more sensitive to **spatial variability**

Inception V3 (Szegedy et al., 2016)



Sample metrics: FID and sFID

- Overall the picture is **mixed**
- **GANs** achieving the **strongest precision** and **FID** scores
- **DCTransformer** typically achieving the best **recall** scores,
- The **FID gap is closed** to a large extent when using the **spatial sFID**



Sample metric results

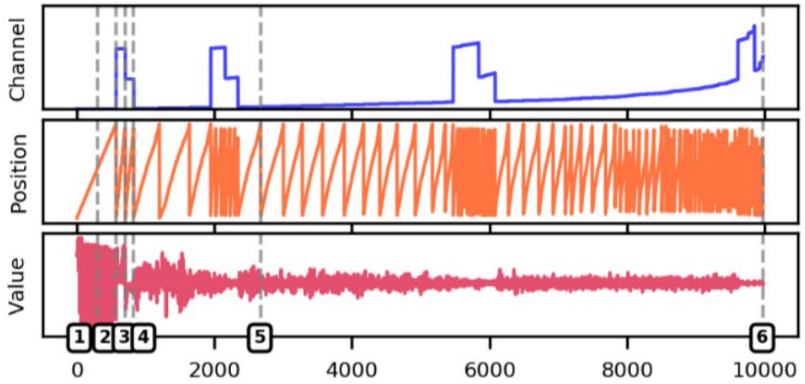
- Overall the picture is **mixed**
- **GANs** achieve the **strongest precision** and **FID** scores
- **DCTransformer** typically achieves the best **recall** scores
- The **FID gap is closed** to a large extent when using the **spatial sFID**

Model	Precision	Recall	FID	sFID
LSUN Bedrooms				
StyleGAN	**0.55**	0.48	**2.45**	6.68
ProGAN	0.43	0.40	8.35	9.46
DCTransformer	0.44	**0.56**	6.40	**6.66**
LSUN Towers				
ProGAN	0.51	0.33	10.24	10.02
DCTransformer	**0.54**	**0.54**	**8.78**	**7.70**
LSUN Churches				
StyleGAN2	**0.60**	0.43	**4.01**	**9.28**
ProGAN	0.61	0.38	6.42	10.47
DCTransformer	0.60	**0.48**	7.56	10.71
FFHQ				
StyleGAN	**0.71**	**0.41**	**4.39**	11.57
DCTransformer	0.51	0.40	13.06	**9.44**
ImageNet (class conditional)				
BigGAN-deep	**0.78**	0.35	**6.59**	**6.66**
VQ-VAE2	0.36	0.57	31.11	17.38
DCTransformer	0.36	**0.67**	36.51	8.24

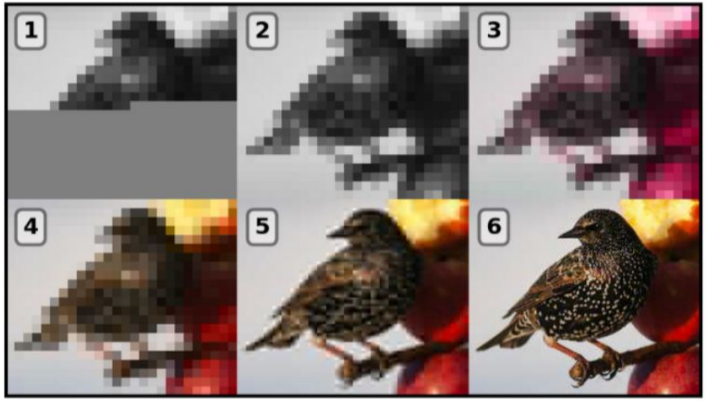
Colorization and Upsampling

f) Coordinate list of non-zero elements from DCT image

$$[\mathbf{t}_l]_{l=1}^{L} = [(c_l, p_l, v_l)]_{l=1}^{L} = \begin{bmatrix} (0,0,507) \\ (0,1,507) \\ (0,2,520) \\ \vdots \\ (6,22,13) \\ (6,25,26) \\ (6,38,26) \\ \vdots \end{bmatrix}$$



g) Decoded images at various stages



Perform **conditional generation** by treating **LHS** of the sequence as **input**

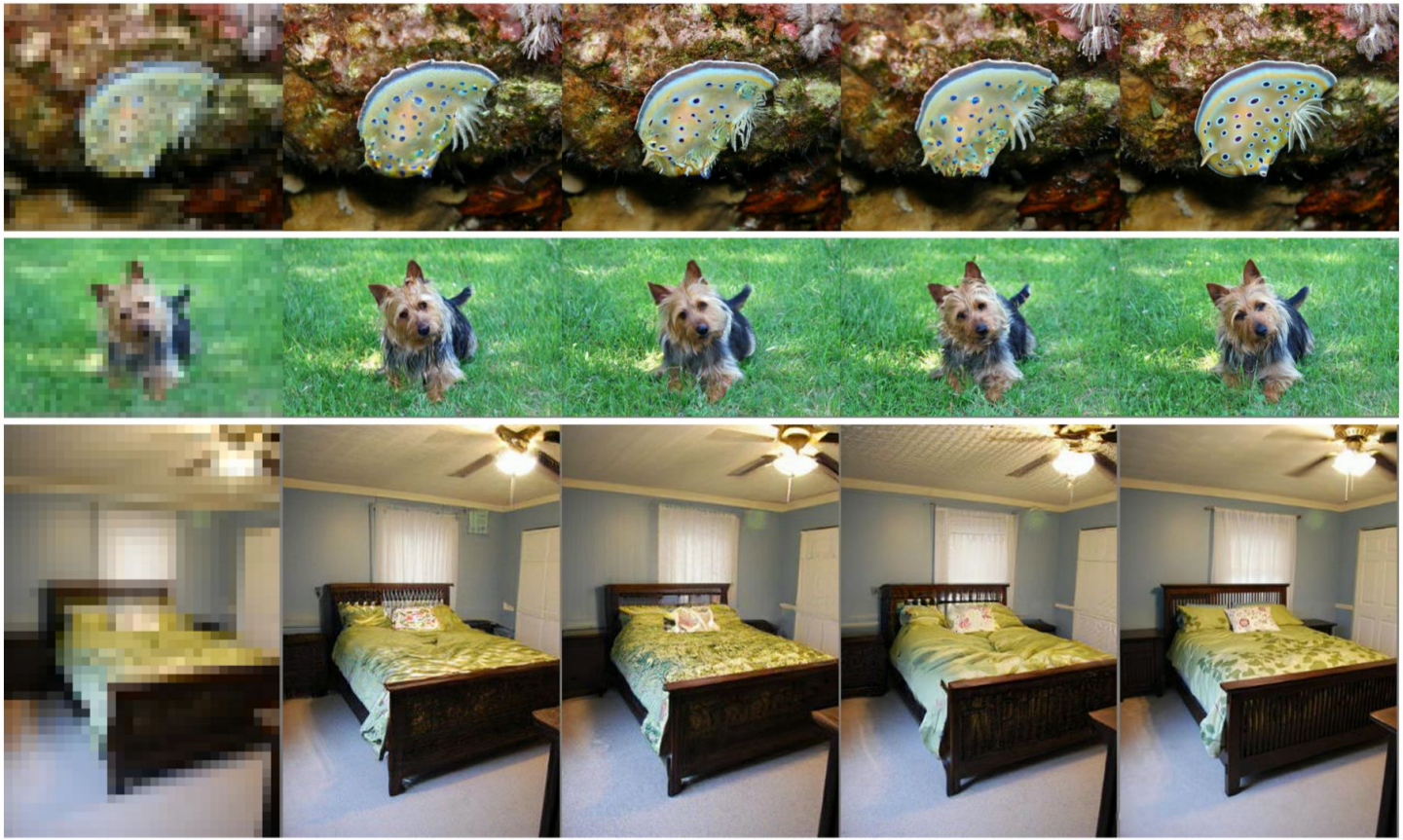
- Conditioning on the **first DCT component** is akin to **upsampling** by a factor of the block size (8x in most of our experiments)
- Can re-order sparse DCT sequence to **place luma components before chroma**. Conditioning on the luma components yields a **colorization** model.

Colorization and Upsampling (cherrypicked)

Figure 7. Image colorization. (left) grayscale image, (middle) three samples generated by DCTransformer, (right) original image.

Figure 8. 8x image upsampling. (left) downsampled image, (middle) three samples generated by DCTransformer, (right) original image.

More upsamping results (not cherrypicked)



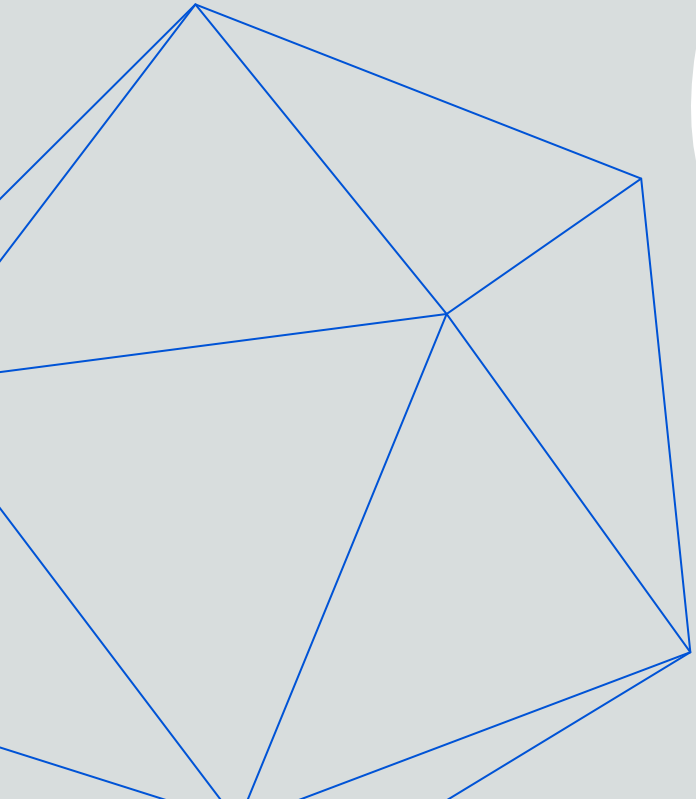
More colorization results (not cherrypicked)

Colorization and Upsampling: Sample metric results

DCTransformer upsampled / colorized images are **distributionally similar** to the training set based on FID / sFID scores

Model	Precision	Recall	FID	sFID
ImageNet (class conditional 8x upsampling)				
Val. set (low res.)	0.02	0.09	163.13	314.27
Val. set	**0.69**	0.59	**5.72**	14.06
DCTransformer	0.61	**0.61**	9.98	**11.06**
OpenImagesV4 colorization				
Val. set (greyscale)	0.61	0.28	34.19	38.09
Val. set	**0.75**	0.40	26.95	27.29
DCTransformer	0.72	**0.41**	**22.52**	**22.83**

DeepMind



Thank You

charlienash@google.com