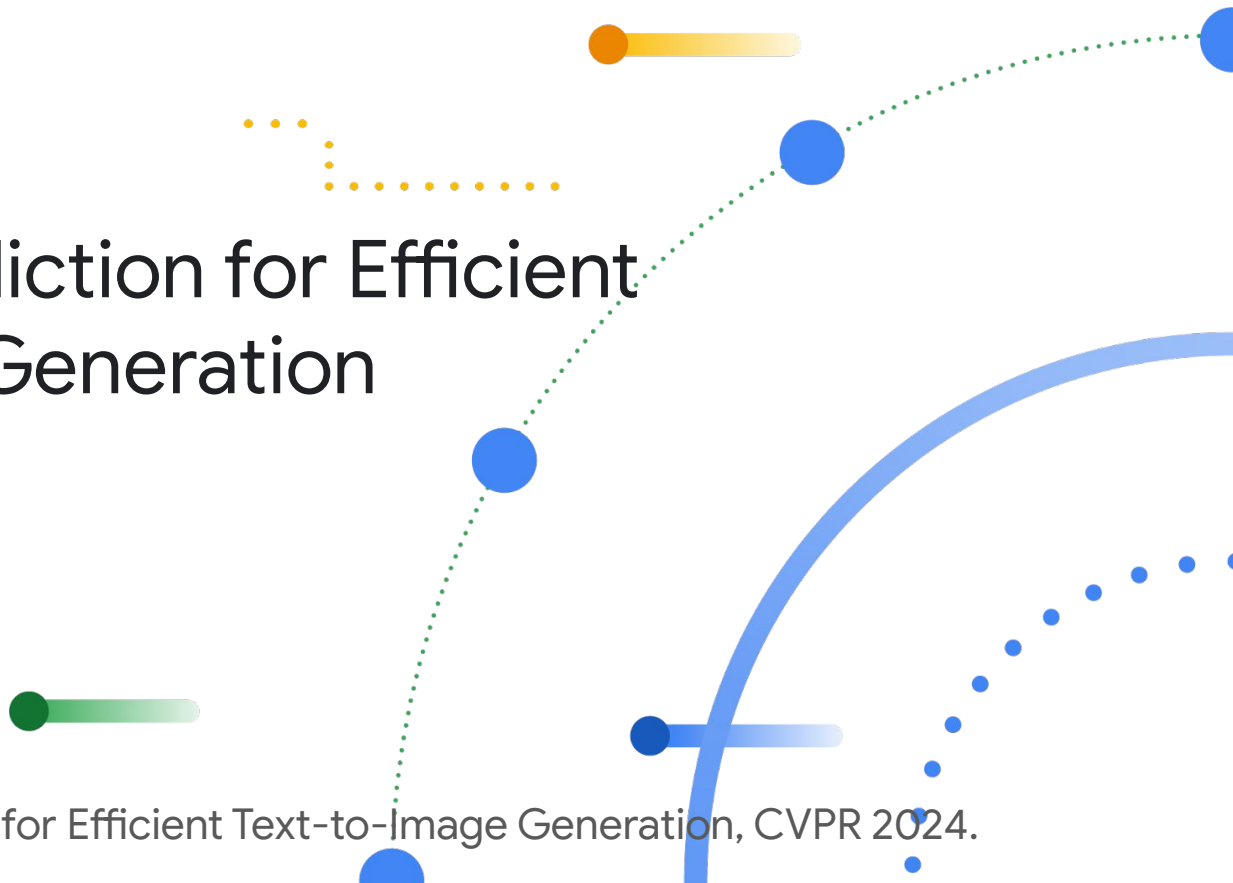
Google Research

Structured Prediction for Efficient Text-to-Image Generation



Based on:
MarkovGen: Structured Prediction for Efficient Text-to-Image Generation, CVPR 2024.

Image Generation Methods

- Diffusion models:
  - Stable Diffusion, Imagen, Dall-E 2
- Transformer-based models:
  - Parti, Muse

Image Tokenization

256 x 256 x 3

Tokenizer

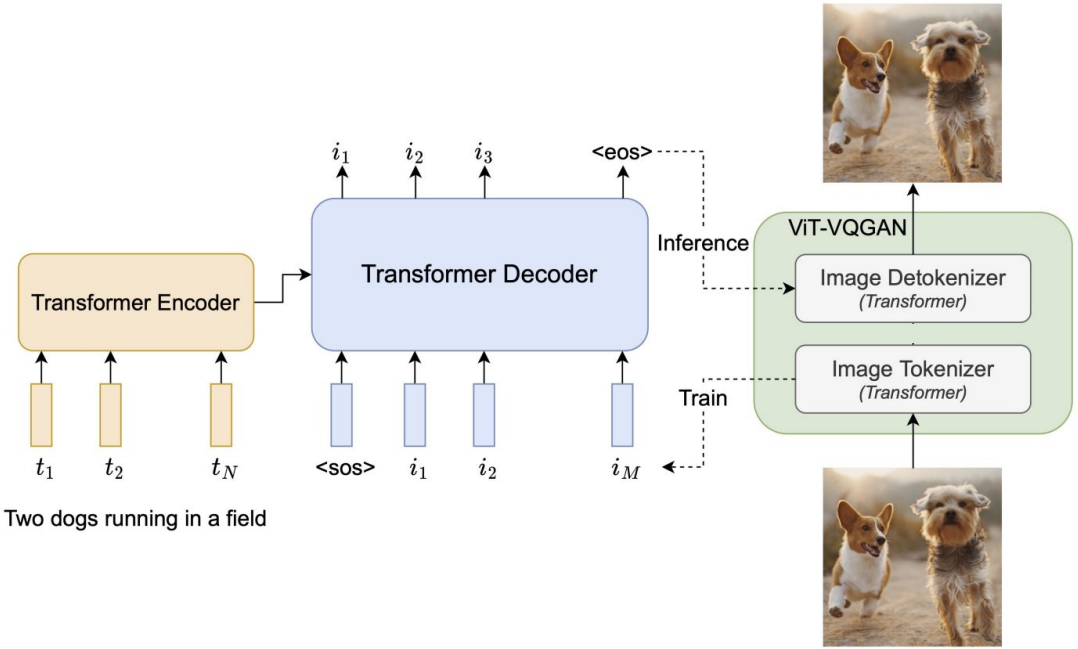
9	28	15	78
19	36	27	32
8	56	68	71
96	85	49	82

16 x 16

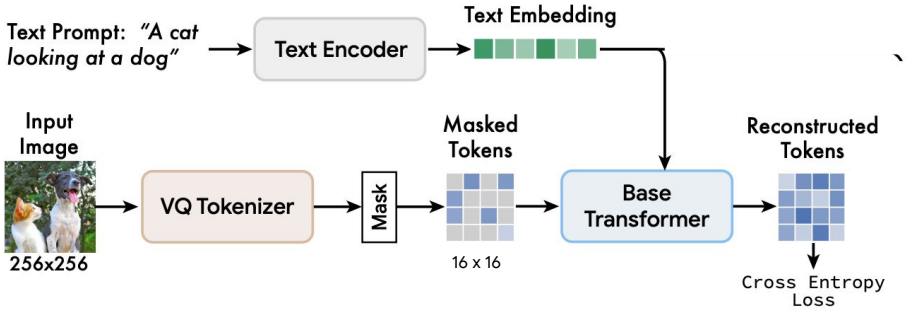
Detokenizer

256 x 256 x 3

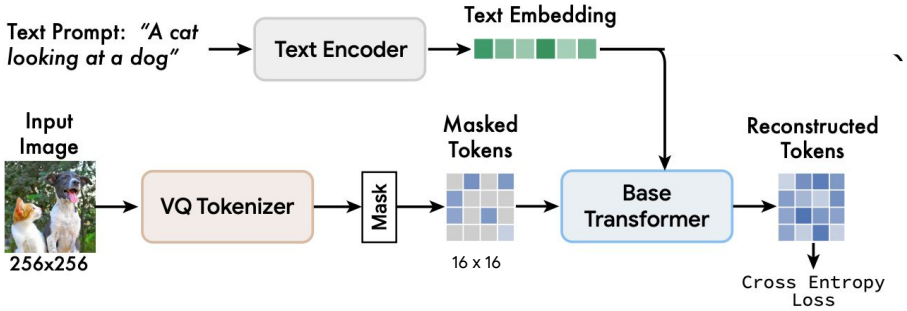
Parti (Autoregressive)



Muse (Parallel Decoding)

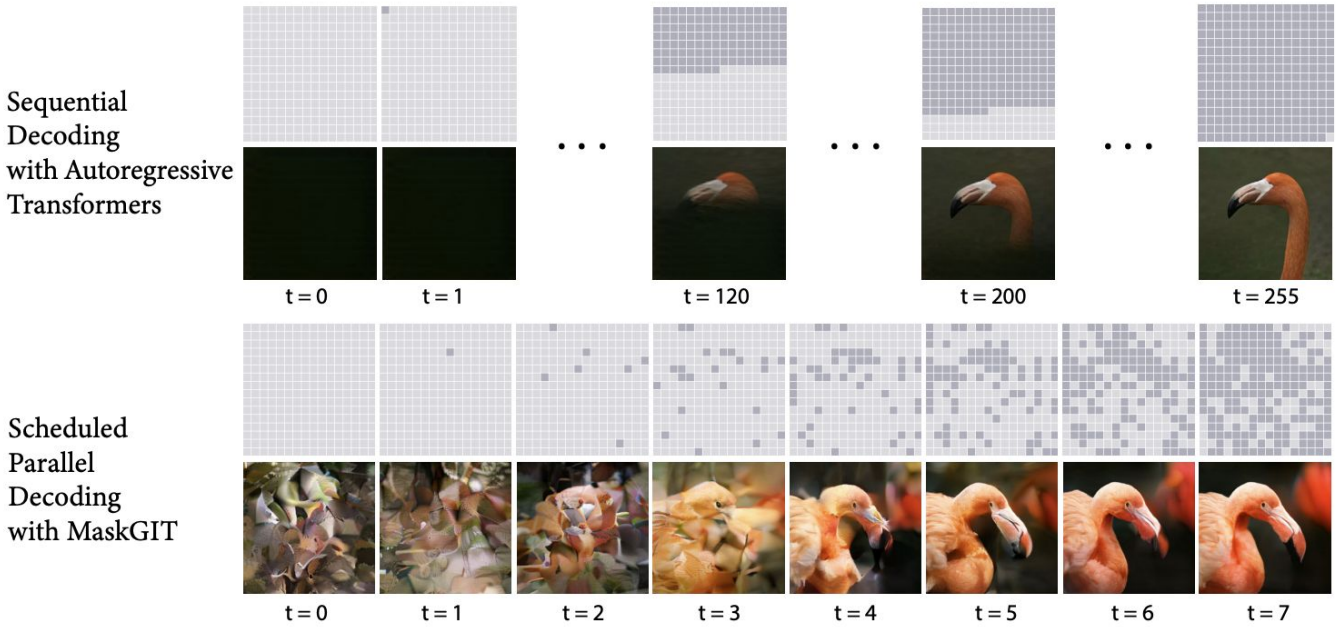


Muse (Parallel Decoding)

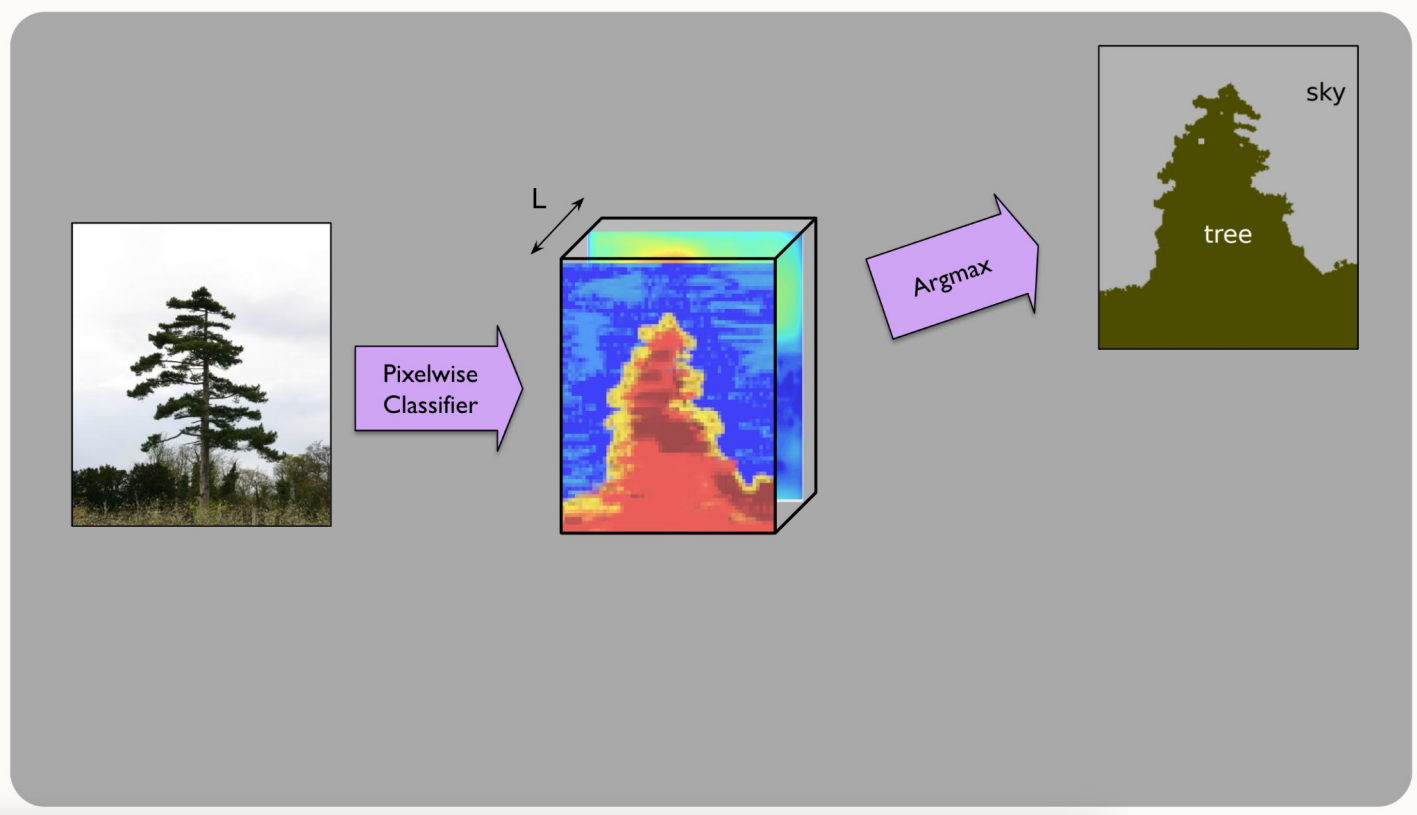


- The Muse 3B model is 10x faster than Parti/Imagen 3B on TPUv4.

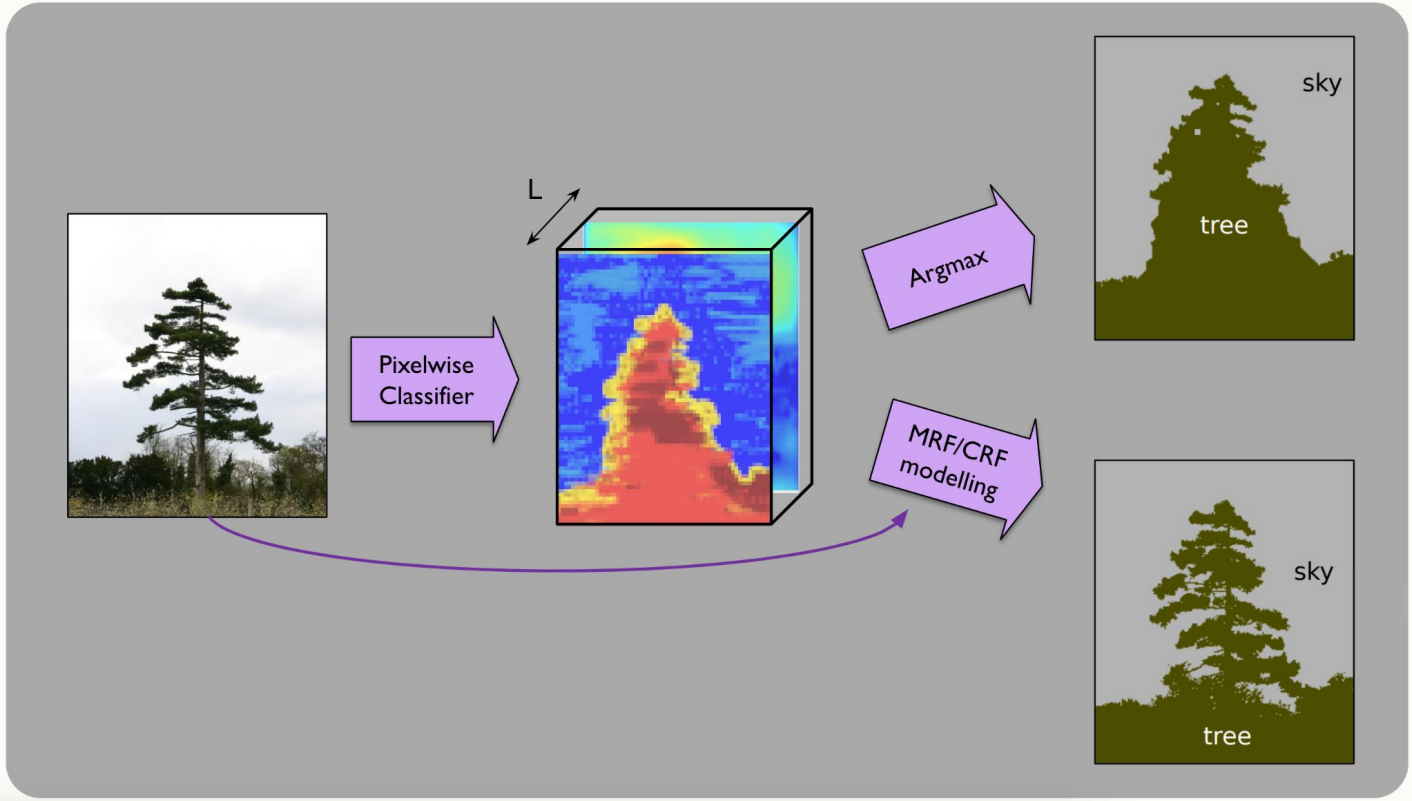
Muse (Parallel Decoding)



Markov Random Fields (MRFs)

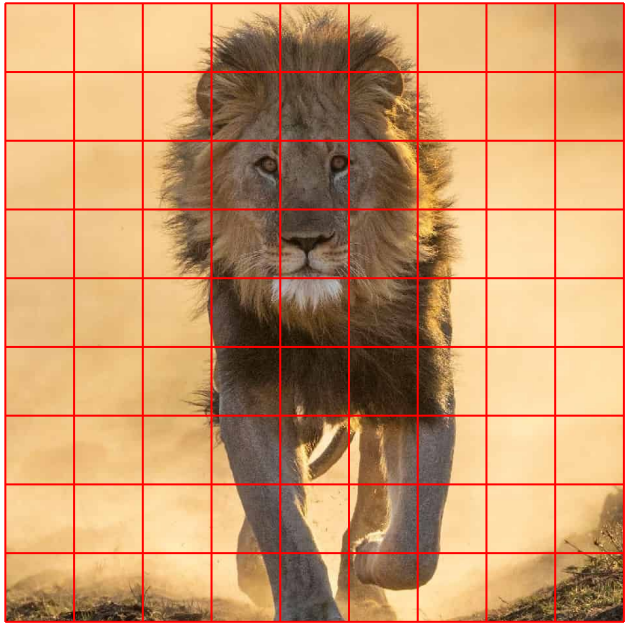


Markov Random Fields (MRFs)

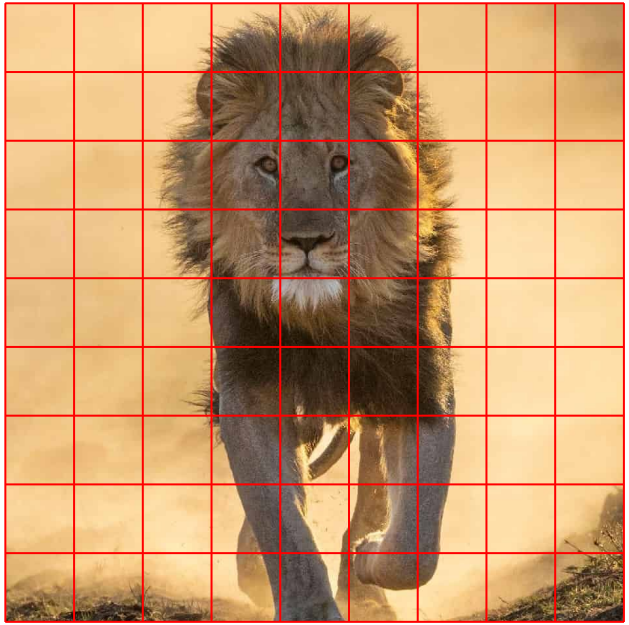


Motivation

Motivation

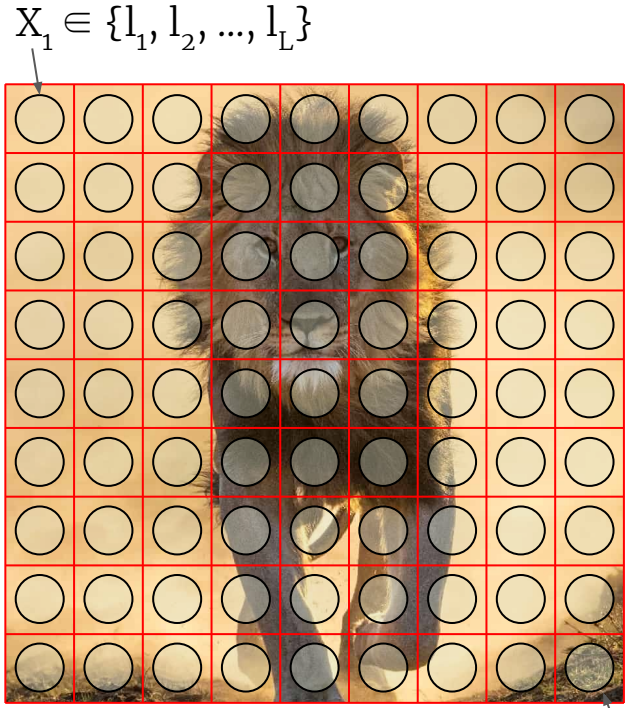


Motivation



- 8192 tokens in the vocab.
- Number of permutations:
  - 2x2 patch: $O(10^{15})$
  - 3x3 patch: $O(10^{35})$
  - 16x16 patch: $O(10^{1002})$
- Only a small subset of token arrangements will be “valid”.
- Highly confident tokens should be able to influence nearby tokens

Markov Random Fields (MRFs)

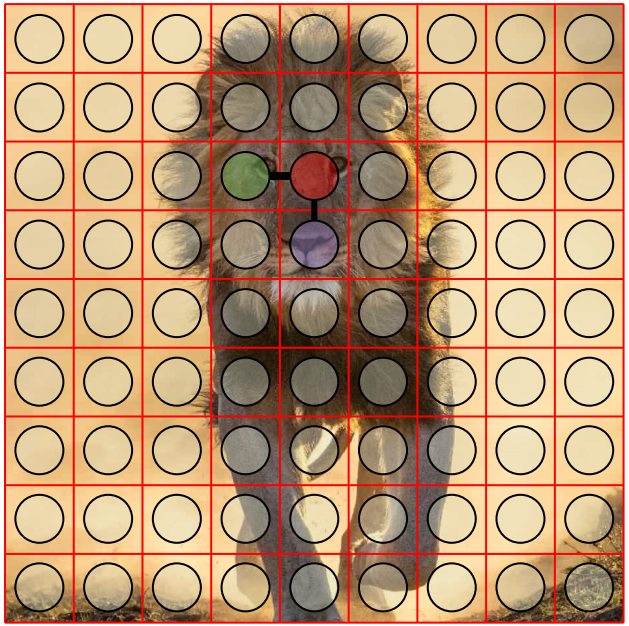


- Define a discrete random variable X_i at each cell i.
- Connect the random variables to form a random field.
- An assignment to the random field $X_1, X_2, ..., X_N \Rightarrow$ an image.

Markov Random Fields (MRFs)

$$P(X_1 = x_1, X_2 = x_2, \ldots, X_N = x_N) = P(\mathbf{X} = \mathbf{x})$$

$$P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z} \exp(-E(\mathbf{x}))$$



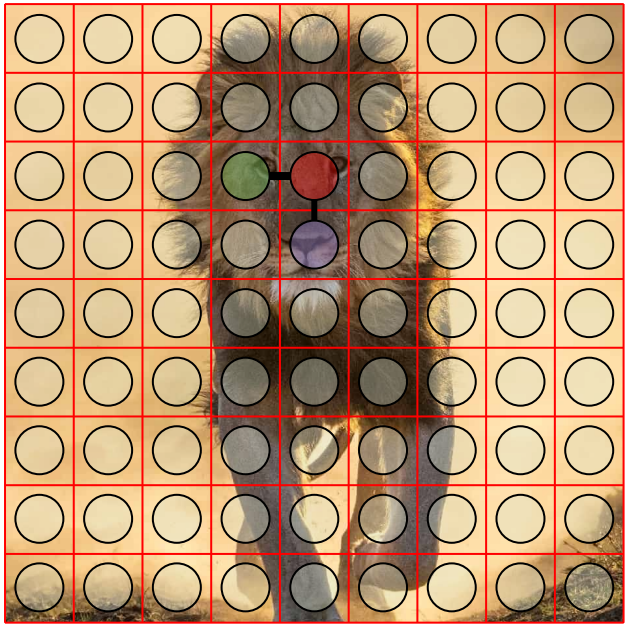
- Maximize $P(\mathbf{X} = \mathbf{x}) \implies$ Minimize $E(\mathbf{X} = \mathbf{x})$
- We now need to define $E(\mathbf{x})$ such that a photorealistic image will have low $E(\mathbf{x})$.

Model Formulation

$$E(\mathbf{x}) = \text{unary\_cost} + \text{pairwise\_cost}$$

Unary Cost

- $\text{cost}(X_i = l) = ?$
- You pay a penalty if your label doesn't agree with the classifier.



Model Formulation

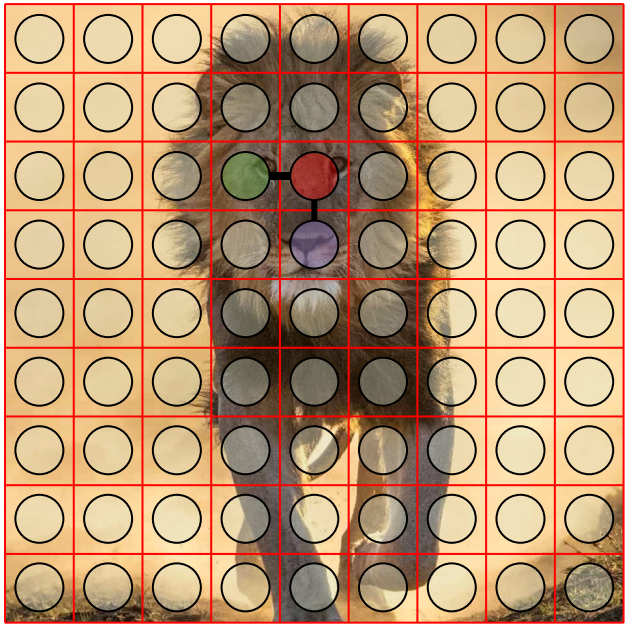
$$E(\mathbf{x}) = \text{unary\_cost} + \text{pairwise\_cost}$$

Unary Cost

- $\text{cost}(X_i = l) = ?$
- You pay a penalty if your label doesn't agree with the classifier.

Pairwise cost

- $\text{cost}(X_i = l', X_j = l'') = ?$
- You pay a penalty if you assign *"incompatible"* labels to two *"neighboring"* pixels.



Model Formulation

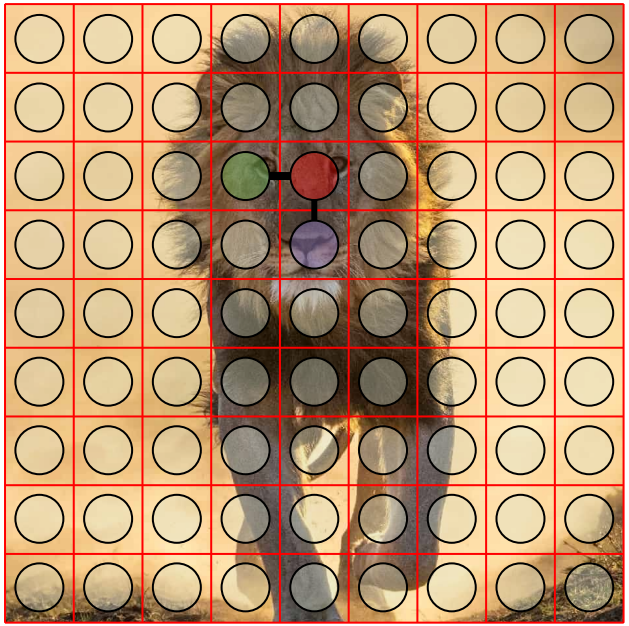
$$E(\mathbf{x}) = \text{unary\_cost} + \text{pairwise\_cost}$$

Unary Cost

- $\text{cost}(X_i = l) = ?$
- You pay a penalty if your label doesn't agree with the classifier.

Pairwise cost

- $\text{cost}(X_i = l', X_j = l'') = ?$
- You pay a penalty if you assign *"incompatible"* labels to two *"neighboring"* pixels.



$$\text{cost}(X_i = l) = -\text{logit}_i(l)$$

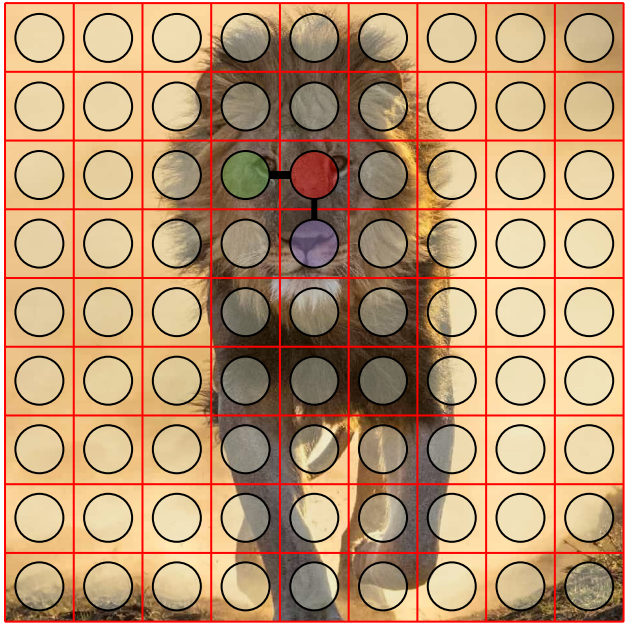
$$\text{cost}(X_i = l', X_j = l'') = -c(l', l'')s(i, j)$$

Difference Compared to Semantic Segmentation

- The graph is truly fully-connected.
- Spatial relationships are not fixed.
- Label compatibilities are not fixed.

$$\text{cost}(X_i = l) = -\text{logit}_\text{i}(l)$$

$$\text{cost}(X_i = l', X_j = l'') = -c(l', l'')s(i, j)$$



Inference Algorithm

$$E(\mathbf{x}|\mathbf{I}) = \sum_i \text{unary}(x_i) + \sum_{i>j} \text{pairwise}(x_i, x_j)$$

Inference Algorithm

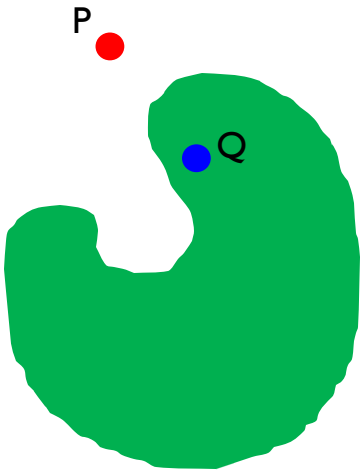
$$E(\mathbf{x}|\mathbf{I}) = \sum_i \text{unary}(x_i) + \sum_{i>j} \text{pairwise}(x_i, x_j)$$

$$\frac{1}{Z(\mathbf{I})} \exp\left(-E(\mathbf{x}|\mathbf{I})\right) = P(\mathbf{X} = \mathbf{x}|\mathbf{I}) \approx \prod_{i=1}^{N} Q_i(x_i)$$

Inference Algorithm

$$E(\mathbf{x}|\mathbf{I}) = \sum_i \text{unary}(x_i) + \sum_{i>j} \text{pairwise}(x_i, x_j)$$

$$\frac{1}{Z(\mathbf{I})} \exp\left(-E(\mathbf{x}|\mathbf{I})\right) = P(\mathbf{X} = \mathbf{x}|\mathbf{I}) \approx \prod_{i=1}^{N} Q_i(x_i)$$

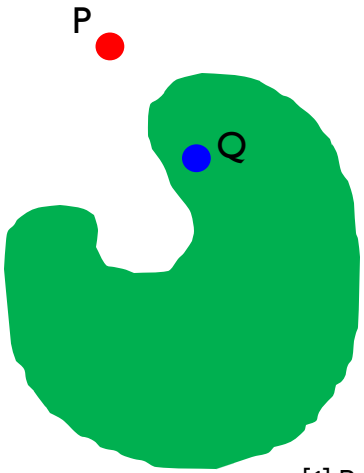


Inference Algorithm

$$E(\mathbf{x}|\mathbf{I}) = \sum_i \text{unary}(x_i) + \sum_{i>j} \text{pairwise}(x_i, x_j)$$

$$\frac{1}{Z(\mathbf{I})} \exp\left(-E(\mathbf{x}|\mathbf{I})\right) = P(\mathbf{X} = \mathbf{x}|\mathbf{I}) \approx \prod_{i=1}^{N} Q_i(x_i)$$

$$D_{\text{KL}}(Q\|P) = \mathbb{E}_Q[\log(Q(\mathbf{x})) - \log(P(\mathbf{x}))]$$



[1] P. Krähenbühl and V. Koltun. Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials, NeurIPS , 2011
[2] D. Koller and N. Friedman. Probabilistic Graphical Models: Principles and Techniques. MIT Press, 2009

Inference Algorithm

Algorithm 1 Inference Algorithm

$Q_i(k) \leftarrow \text{softmax}(f_i(k)), \forall(i,k)$
for `num_iterations` **do**
  $Q_i(k) \leftarrow \sum_{j=1}^{n} \mathbf{W^s}_{ij} Q_j(k), \forall(i,k)$
  $Q_i(k) \leftarrow \sum_{k'=1}^{V} \mathbf{W^c}_{kk'} Q_i(k'), \forall(i,k)$
  $Q_i(k) \leftarrow Q_i(k) + f_i(k), \forall(i,k)$
  $Q_i(k) \leftarrow \text{softmax}(Q_i)(k), \forall(i,k)$
end for
return Q

MRFs for Fast Image Generation

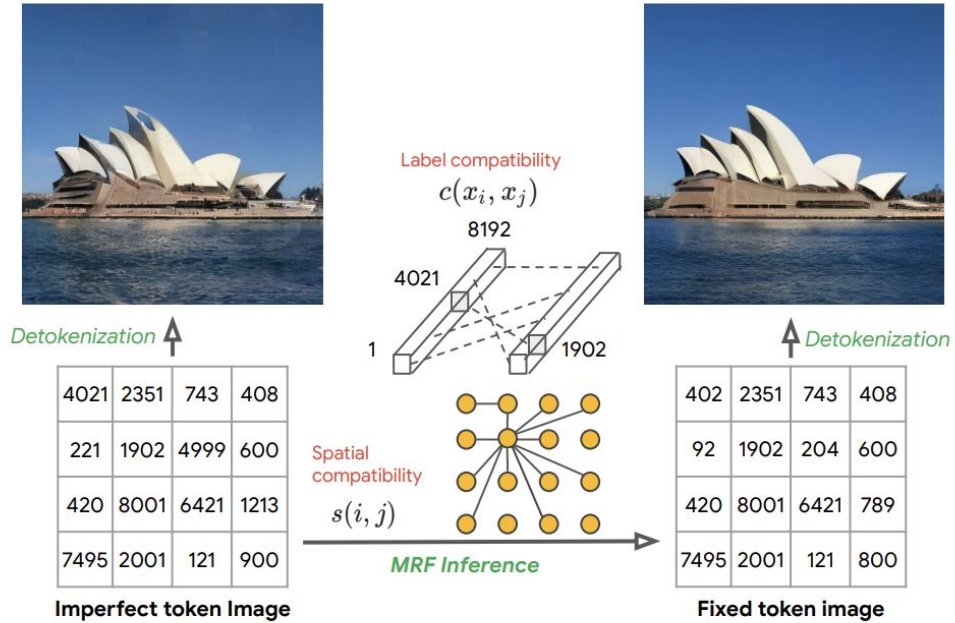
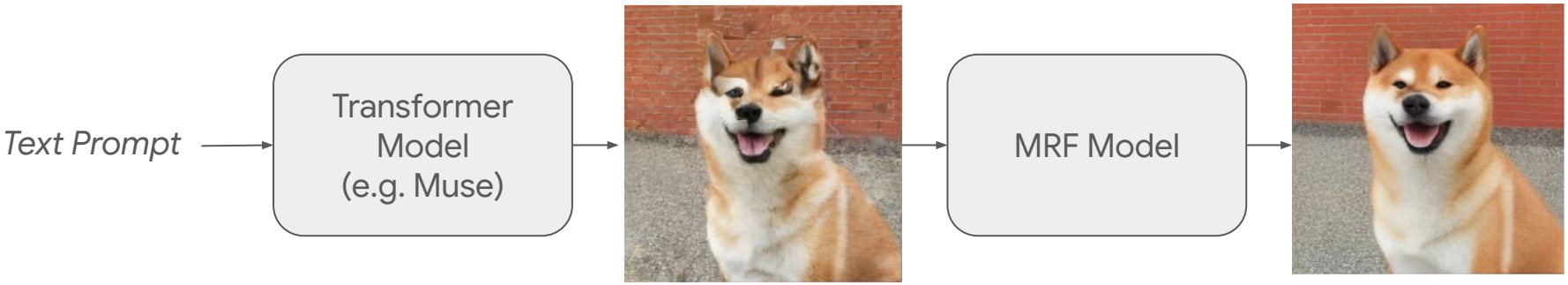


Figure 3. *Given individual token probabilities from an underlying Transformer-based image generation backbone, the MRF improves image quality by utilizing learned spatial and label compatibility relations in the latent token space.*

MRFs for Fast Image Generation



- Much of the heavy-lifting is done here.
- Bulky, slow model

- Fixes the incompatible tokens
- Light-weight and super fast

Speeding Up Inference with MRFs

Model	Time (ms)
Muse base (single step)	10.40
Muse super-resolution (single step)	24.00
MRF inference on base	0.29
MRF inference on super-resolution	0.29
Detokenizer	0.15
Muse	442.05
MarkovGen (ours)	281.03

Generation Quality

Model	FID
Muse base (18 iters)	15.48
Muse base (24 iters)	14.13
Muse base (18 iters) + MRF	13.00

Table 1: Quantitative evaluation of FID scores on the MS-COCO [Lin et al., 2014] dataset for 256×256 image resolution. The Muse model with MRF applied after 18 steps outperforms both the Muse model with 24 as well as the model with 18 steps.

Qualitative Results

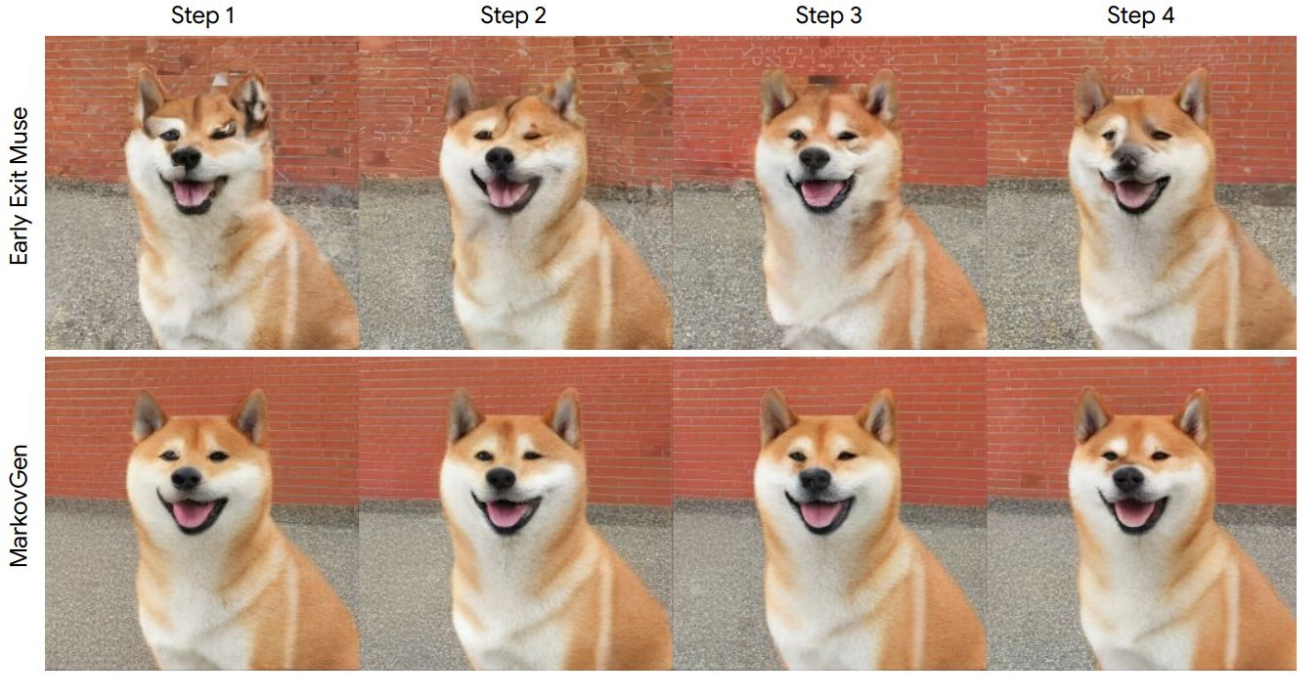


Figure 4. *The first four steps of the Muse super-resolution model without (top) and with (bottom) the application of the MarkovGen MRF model. Note that the MRF fixes complex object structures such as the dog's face as well as texture-inconsistencies in areas such as the brick wall. MarkovGen generates good looking high quality images starting from the first step.*

The finale of a fireworks display

An oil painting of two rabbits in the style of American Gothic, wearing the same clothes as in the original.

A robot painted as graffiti on a brick wall. a sidewalk is in front of the wall, and grass is growing out of cracks in the concrete.

A set of 2x2 emoji icons with happy, angry, surprised and sobbing faces. The emoji icons look like pandas. All of the pandas are wearing colorful sunglasses.

Figure 6. *Within each set of three, MarkovGen (right) speeds up Muse (left) by* $1.5\times$ *and improves image quality. A similar speed up by only reducing the step count with early exit Muse (middle) results in a significant loss of quality.*

Qualitative Results

Figure 7. *Example generations of the Early Exit Muse super-resolution model running for 3 (out of 8) steps (top) and the MarkovGen model after the application of the* MRF *model (bottom). We observe a significant reduction in visual artifacts, e.g., in the brick wall behind the car. We further see key improvements to complex object structures such as the blue car and the teddy bear's face.*

Quantitative Results

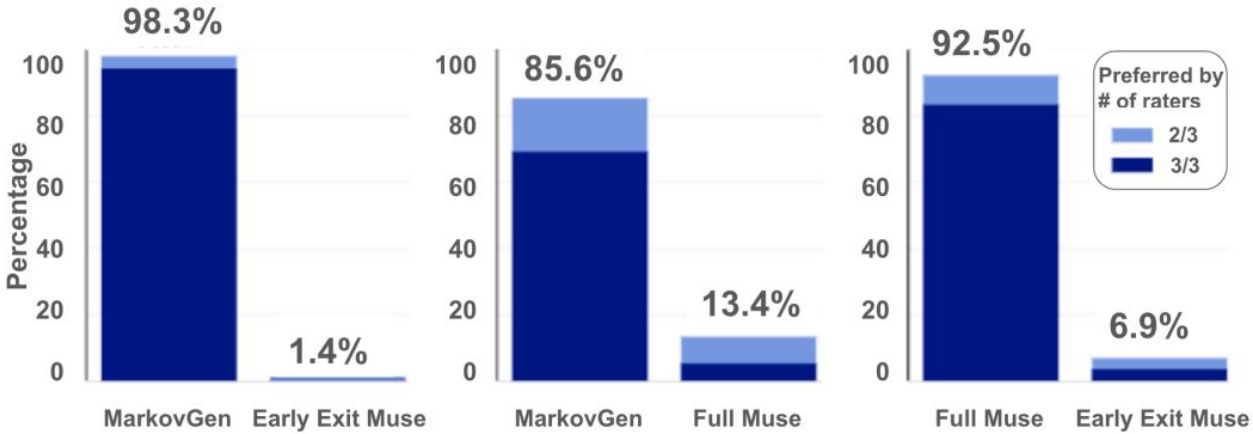


Figure 5. *Percentage of prompts for which human raters prefer images by a given model in a side-by-side comparison. We observe that human raters strongly prefer the images generated by MarkovGen over those of both early exit Muse (left) and even the more expensive and slower full Muse model (center).*

Thank you!