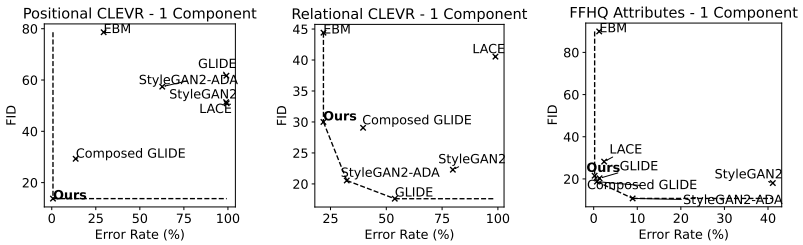

Figure 8. Scatter plots of compositional generation error vs FID on 3 datasets (1 input component): Our method lies on the Pareto front of all results while achieving state-of-the-art (lowest or joint lowest) error.

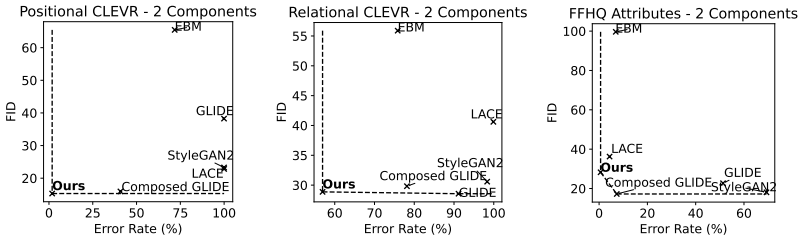

Figure 9. Scatter plots of compositional generation error vs FID on 3 datasets (2 input component): Our method lies on the Pareto front of all results while achieving state-of-the-art (lowest or joint lowest) error.

Appendix

A. Full Dataset Details

Below we provide a full description of each dataset used in the quantitative evaluation of our method.

- **FFHQ** [26]: FFHQ is a dataset of 70,000 aligned images of human faces. Three binary attribute labels are available for each image: "smile"/"no smile", "glasses"/"no glasses", "male"/"female", which we use to condition the generation process.
- **Positional CLEVR** [24]: CLEVR is a synthetic dataset of rendered 3D objects of various colours, shapes, sizes and textures. In the Positional variant of CLEVR, object attribute and position annotations are available. Following [31], we use a 30,000-image subset of CLEVR (restricted to contain between 1 and 5 objects per image). For this task, image generation is conditioned on object position only.
- **Relational CLEVR** [24, 31]: Relational CLEVR is similar in appearance to Positional CLEVR, with the addition of text annotations for objects and their relationships (e.g. "*the red cube is above the blue sphere*"). Image generation is conditioned on (tokenized) text descriptions of object attributes and relationships, including object shape, size, material, colour, and relative position.

B. Error vs FID Plots

Figures 8, 9 and 10 are scatter plots of error rate against FID corresponding to results in Tables 1, 2 and 3. Included on the same axes of each plot are the empirical Pareto front of the data, which we define as the unique linear piece-wise form that is convex, monotonic, and is as close as possible to the lowermost and leftmost (best) of the data points while extending to the top and right of each plot. Our results all lie on, or very close to, the empirical Pareto front of their respective plots while having the lowest error rate when ranked among the baselines.

C. Derivations

Here we include additional derivations which were omitted from the main paper for brevity.

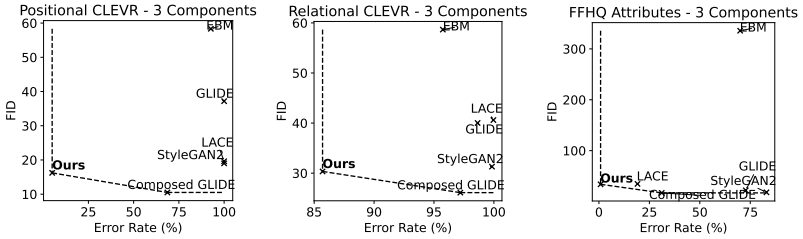

Figure 10. Scatter plots of compositional generation error vs FID on 3 datasets (3 input component): Our method lies on the Pareto front of all results while achieving state-of-the-art (lowest or joint lowest) error.

C.1. Cancelling the $P(\boldsymbol{c}_i)$ terms

The following is a full derivation of the result given in (2) of the main text.

$$P(\boldsymbol{x} = x_j|\boldsymbol{c}_1, ..., \boldsymbol{c}_n) = \frac{P(\boldsymbol{x} = x_j)\prod_{i=1}^{n}\frac{P(\boldsymbol{x}=x_j|\boldsymbol{c}_i)P(\boldsymbol{c}_i)}{P(\boldsymbol{x}=x_j)}}{\sum_{u=1}^{k}\left[P(\boldsymbol{x} = x_u)\prod_{i=1}^{n}\frac{P(\boldsymbol{x}=x_j|\boldsymbol{c}_i)P(\boldsymbol{c}_i)}{P(\boldsymbol{x}=x_j)}\right]}, \tag{8}$$

$$= \frac{P(\boldsymbol{x} = x_j)\prod_{i=1}^{n}\frac{P(\boldsymbol{x}=x_j|\boldsymbol{c}_i)}{P(\boldsymbol{x}=x_j)}}{\sum_{u=1}^{k}\left[P(\boldsymbol{x} = x_u)\prod_{i=1}^{n}\frac{P(\boldsymbol{x}=x_u|\boldsymbol{c}_i)}{P(\boldsymbol{x}=x_u)}\right]}. \tag{9}$$

In this way, the contribution of $P(\boldsymbol{c}_i)$ terms is cancelled out, since $P(\boldsymbol{c}_i)$ are constant with respect to different values of x_j. In subsequent sections, we show how this brief but important result for conditional categorical distributions can be successfully extended to parallel token prediction for image generation, achieving state-of-the art error rate and speed on 3 datasets.

C.2. Extension of product of experts to sequential discrete generation

Here we clarify the relationship between equations (2) and (3) in the main text.

Equation (3) holds if and only if the joint distributions $\boldsymbol{s}_t, \boldsymbol{c}_i$ and $\boldsymbol{s}_t, \boldsymbol{c}_i$ are independent conditional on $\boldsymbol{s}_{t+1}$, i.e. $P(\boldsymbol{s}_t, \boldsymbol{c}_i, \boldsymbol{c}_j|\boldsymbol{s}_{t+1}) \propto P(\boldsymbol{s}_t, \boldsymbol{c}_i|\boldsymbol{s}_{t+1})P(\boldsymbol{s}_t, \boldsymbol{c}_j|\boldsymbol{s}_{t+1})$ for all pairs of distinct input conditions $\boldsymbol{c}_i, \boldsymbol{c}_j$ (this is the product of experts assumption, extended to sequential conditional generation).

$P(\boldsymbol{s}_t, \boldsymbol{V}|\boldsymbol{s}_{t+1}) = P(\boldsymbol{V}|\boldsymbol{s}_{t+1})$ for any random variable or collection of variables $\boldsymbol{V}$, since $\boldsymbol{s}_t$ is entirely determined by $\boldsymbol{s}_{t+1}$ (the representation of $\boldsymbol{s}_t$ is contained in that of $\boldsymbol{s}_{t+1}$). Therefore the expression of the extended product of experts assumption can be simplified to

$$P(\boldsymbol{c}_i, \boldsymbol{c}_j|\boldsymbol{s}_{t+1}) \propto P(\boldsymbol{c}_i|\boldsymbol{s}_{t+1})P(\boldsymbol{c}_j|\boldsymbol{s}_{t+1}),$$

i.e. the extended product of experts assumption is equivalent to the original product of experts assumption and therefore (3) follows from (2) with the substitution of a categorical variable $\boldsymbol{x}$ with a stateful discrete generative process $\boldsymbol{s}_{t+1}|\boldsymbol{s}_t$.

C.3. Compositional Next-Token Prediction

In the main text we claim that our discrete compositional method can be applied to arbitrary generative methods provided they are discrete and iterative. Here we back up this claim by showing how our method can be adapted to autoregressive (next-token) sampling.

In the specific case of discrete autoregressive image modelling, a single latent code (token) is generated at each time step, conditioned on some initial context in addition to previously generated tokens. In this situation, each successive state $\boldsymbol{s}_{t+1}$ is simply the concatenation of the previous state $\boldsymbol{s}_t$ and the subsequent generated token $\boldsymbol{x}_{t+1}$. Thus the random variable $\boldsymbol{s}_{t+1}$ can be restated as

$$\boldsymbol{s}_{t+1} = \boldsymbol{s}_t \oplus \boldsymbol{x}_{t+1}, \tag{10}$$

where $\oplus$ denotes the concatenation of two tokens or strings of tokens. Consequently, the conditional generation task in (3) can be reformulated in terms of sampling the next token given the previous state:

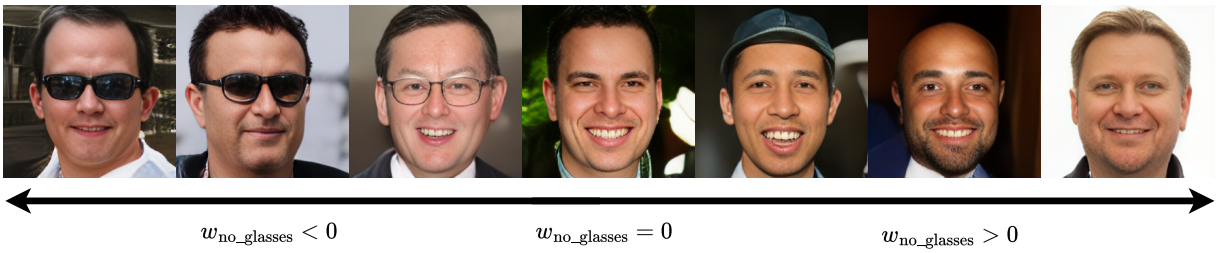

Figure 11. Effect of varying the $w_{\text{no\_glasses}}$ concept weight from -3.0 to 3.0 while keeping $w_{\text{male}} = w_{\text{smile}} = 3.0$.

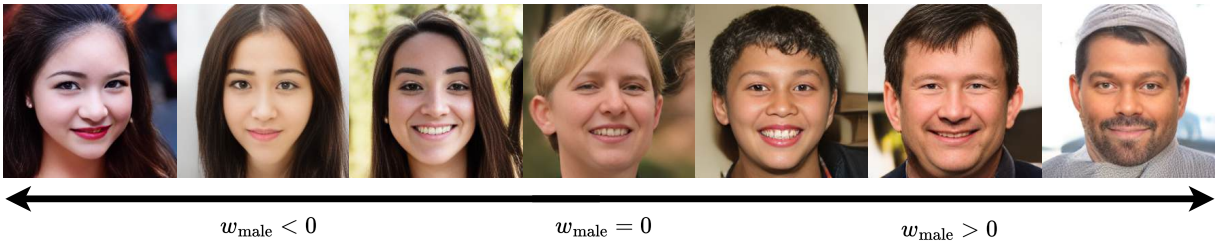

Figure 12. Effect of varying concept weight w_{male} for our model trained on the FFHQ dataset: Concept weighting (both positive and negative) allows for fine-grained control of output attributes.

$$P(\boldsymbol{x}_{t+1}|\boldsymbol{s}_t, \boldsymbol{c}_1, ..., \boldsymbol{c}_n) \propto P(\boldsymbol{x}_{t+1}|\boldsymbol{s}_t) \prod_{i=1}^{n} \frac{P(\boldsymbol{x}_{t+1}|\boldsymbol{s}_t, \boldsymbol{c}_i)}{P(\boldsymbol{x}_{t+1}|\boldsymbol{s}_t)}, \tag{11}$$

i.e. only a single token is considered at each generation step, making our formulation compatible with autoregressive (next-token) prediction. In practice, we speculate that the autoregressive case is less compatible with our compositional method than parallel token prediction due to being less strongly regularised (and hence more prone to over-fitting: parallel token prediction is strongly regularised by design [2]), in addition to being more sensitive to the accumulation of errors due to tokens being generated "left-to-right, top to bottom" in the image. Furthermore, there is no guarantee that autoregressive models provide a calibrated estimate of conditional/unconditional probabilities, which may further limit hypothetical performance. For these reasons, we maintain our focus on parallel token prediction which is found to to outperform the previous state-of-the-art on image generation error rate when applied alongside our discrete composition method.

C.4. Omission of the disjunction (OR) operator

While the implementation conjunction ("AND") and negation ("NOT") operators are highly effective for controllable generation, they do not correspond exactly to Boolean algebra: in our framework, negation is distributive, i.e. $-(a + b) \equiv -a - b$, but in Boolean algebra, it is not: NOT(a AND b) $\neq$ NOT a AND NOT b for $a \neq b$. For this reason, our framework does not allow for the straightforward implementation of the disjunction (OR) operator, which we do not explore in our qualitative or quantitative experiments.

D. Additional Qualitative Experiments

D.1. Varying Concept Weight for FFHQ

In the main text we discuss the effect of varying w_{male} for the FFHQ dataset. Figures 12 and 11 visualise the effect of varying the weights of the remaining two concepts in FFHQ (w_{male} and w_{smile} respectively).

E. Full Model Training Details

Here we give the full technical details of our training runs in addition to hardware considerations.

For each dataset, we utilise a deep residual convolutional vector-quantized autoencoder architecture following the protocol of [12] and [2] which have been previously shown to produce high-fidelity reconstructions for a variety of image datasets. We

use a training batch size of 20 for CLEVR and Relational CLEVR, with a smaller batch size of 8 for FFHQ due to the larger image size. CLEVR and Compositional CLEVR VQ-VAE models are trained for $20,000$ iterations each, while the FFHQ VQ-GAN is trained for $100,000$ iterations due to the smaller batch size and larger resolution, with adverserial loss starting at $30,000$ iterations.

We set the codebook size at 256 for all three models, which we find to be sufficient for obtaining high-quality reconstructions. For sampling, we use the encoder-only transformer architecture, using the exact same same architecture for all three datasets (from [2]) (24 layer, embedding dimension 512, fully connected hidden dimension 2048). We train a total of 3 samplers (one for each dataset) for $300,000$ iterations each. Training each sampler model took 1.5 days per model, with an additional 0.5 days for evaluations. Training of all models and running evaluation took approximately 5 days in total, using a single NVIDIA GeForce RTX 3090 and implementation in Pytorch. Preliminary and failed experiments (e.g. due to bugs) made up for around 3 days of compute on the same hardware.

For each dataset, we encode input conditions as additional embeddings which are concatenated to the latent embeddings before being fed to the transformer. Object position for CLEVR is encoded using a learned linear map $M_{pos} : \mathbb{R}^2 \to \mathbb{R}^d$ where d is the hidden dimension of the transformer. Face attributes for FFHQ are encoded using learned embedding $M_{face} : \{$"smile", "no smile", "glasses", "no glasses", "female", "male"$\} \to \mathbb{R}^d$. For Relational CLEVR, we tokenize text descriptions, mapping to learned token embeddings and adding positional embeddings before concatenating with the learned image token embeddings (also adding learned position embeddings for the image token embeddings).

F. Baselines Details

In the quantitative evaluations of our method, we compare against the baseline results provided in [31] in addition to the results of the method in [31]. For completeness, below we provide a brief description of how the baseline results were obtained in [31]:

- **StyleGAN2-ADA** [27] - The StyleGAN2-ADA results were obtained in [31] using the off-the-shelf model provided by [27].
- **StyleGAN2** [28] - Compositional StyleGAN2 results were obtained in [29] by training classifiers on the latent space of StyleGAN2, which were then used to generate novel latent representations. StyleGAN2 models were either used off-the-shelf (for FFHQ) or trained from scratch (for Positional and Relational CLEVR).
- **LACE** [34] - LACE results were obtained in [31] by composing energy-based models acting on the latent space as in [34], with training data generated by StyleGAN2-ADA (above).
- **GLIDE** [33] - The (non-composed) GLIDE results were obtained in [31] by encoding input conditions as a single, long sentence, with outputs being upsampled separately from 64×64.
- **EBM** [11] - Composed EBM results were obtained in [31] by composing conditional energy functions for multiple concepts as in [11].
- **Composed GLIDE** [31] - Composed GLIDE results were obtained by [31] using the method for composing diffusion outputs proposed by [31].

We had insufficient resources to train our own models from scratch, and so we instead precisely replicated the evaluation protocol of [31] to enable fair comparison with our own method.

G. Binary Classifier Details

In the main text we include generation error rate metrics for baseline methods (from [31]) and our own experiments with Positional CLEVR, Relational CLEVR and FFHQ. Here we document fully how we obtained our own error rate scores by following the evaluation approach of [31] so as to maintain valid comparisons with the baseline results reported by [31].

Accuracy is determined by a binary classifier for each of the three datasets, which takes both an image and an attribute as input and produces a binary output corresponding to whether the specified attribute is present. We obtained the error rate scores following the exact same evaluation approach of [31] so as to maintain valid comparisons with the baseline results reported by [31]. For each experiment we generate 5000 images, computing accuracy (Acc) and FID for each group of 5000. Samples are taken over 30 time steps (corresponding to unmasking $256/30 \approx 8.53$ tokens per time step on average). We fix the concept weight w_i at 3.0 for all experiments. We detail all quantitative results in Table 1 and in Tables 2 and 3 in Appendix **??** (best performance is written in bold for each column, second-best is underlined).

For CLEVR and Relational CLEVR, we use the pre-trained classifiers provided by [31], which have validation classification accuracy scores of 99.05% and 99.80% respectively. For FFHQ, since no pre-trained classifier was available we trained binary classifiers following the same procedure as [31] (one for each attribute, with a $80:20$ train-validation split). The classifiers achieve equal or greater validation accuracy than the classifiers usef by [31]. The high validation accuracy scores

for the evaluation classifiers are deemed sufficient to allow reliable estimation of the error rate for our generated images. Our quantitative evaluations follow the exact same procedure used to obtain the baseline results [31], allowing for a fair comparison with the baselines.

H. Standard Uncertainty Computation

In Tables 1, 2 and 3 we include standard uncertainties at two standard deviations (2σ). We base this computation on the number of sampled images in order to give context to the difference between baseline results and our own results (this is especially useful when results are close together). We are unable to report error bars based on multiple repeats of training runs because such error bars were not reported in [31] and we lack the resources to perform our own runs of their experiments. For these reasons, the value of 2σ (two standard deviations) is derived and computed as follows for a percentage accuracy score p:

We assume that a given method generates an image correctly (consistent with the specified conditions) with probability p, independently for each of n trials (generated samples). It follows that the number of "correct" samples X is distributed according to the Binomial distribution:

$$X \sim B(n, p). \tag{12}$$

The variance in the number of correct samples X is then $np(1-p)$ and thus the standard deviation is $\sqrt{np(1-p)}$. The standard deviation σ of the accuracy score (or equivalently, that of the error rate) is then $SD(X)$ divided by n:

$$\sigma = \frac{\sqrt{np(1-p)}}{n}, \tag{13}$$

$$= \frac{\sqrt{p(1-p)}}{\sqrt{n}}, \tag{14}$$

$$= \sqrt{\frac{p(1-p)}{n}} (\times 100\%). \tag{15}$$

$$\tag{16}$$

Finally, we clip values for 2σ to be no greater than p and no greater than $1-p$ so as to avoid giving confidence bounds greater than 100% or smaller than 0%. We compute 2σ in the same way for all accuracy results (including those reported by [31]) since they are all computed based on 5000 generated samples.

We omit uncertainties for FID for two reasons: (1) the uncertainty in FID for comparing two sets of 5000 images is expected to be low [17] and (2) it would take too long with our available computational resources to compute these by repeating all experiments (including running the baselines) multiple times.

I. Nearest Neighbours of Generated Samples

To verify that our method does not simply reproduce samples from the training data (over-fitting), for each dataset we generate a batch of 8 images based on a random selection of 3 input conditions (positions, relations and attributes for CLEVR, Relational CLEVR and FFHQ respectively). In this experiment, we choose to study the composition of 3 input conditions (as opposed to 1 or 2) as this situation is the most likely to produce over-fit images (due to it finding the "smallest", or lowest-entropy section of the sample space). We compute the 8 nearest neighbours of each sample from the original training data based on perceptual distance. Figures 13, 14 and 15 visualise the results. The leftmost column of each figure contains the (non-cherry-picked) generated samples, while the remaining 8 images in each row are the nearest neighbours. These figures show qualitatively that none of the 8 generated samples from each dataset perfectly match the nearest neighbours, indicating strong generalisation performance. All samples were taken at temperature 0.9 and condition weight 3.0, in accordance with quantitative experiments in the main text.

J. Code License and Running Instructions

Please see LICENSE in the code for full license(s).

Please see README.md in the code for running instructions for reproducing our quantitative experiments, including model training, classifier training, and obtaining results.

Figure 13. Perceptual nearest neighbours for the (positional) CLEVR dataset. Leftmost column is (non-cherry-picked) generated samples, remaining images in each row are the 8 nearest neighbours (left to right goes from nearest to farthest).

.

Figure 14. Perceptual nearest neighbours for the Relational CLEVR dataset. Leftmost column is (non-cherry-picked) generated samples, remaining images in each row are the 8 nearest neighbours (left to right goes from nearest to farthest).

.

Figure 15. Perceptual nearest neighbours for the FFHQ dataset. Leftmost column is (non-cherry-picked) generated samples, remaining images in each row are the 8 nearest neighbours (left to right goes from nearest to farthest).

.

Figure 15. Perceptual nearest neighbours for the FFHQ dataset. Leftmost column is (non-cherry-picked) generated samples, remaining images in each row are the 8 nearest neighbours (left to right goes from nearest to farthest).

.