TextCraft: Zero-Shot Generation of High-Fidelity and Diverse Shapes from Text

Anonymous authors
Paper under double-blind review

Abstract

Language is one of the primary means by which we describe the 3D world around us. While rapid progress has been made in text-to-2D-image synthesis, similar progress in text-to-3D-shape synthesis has been hindered by the lack of paired (text, shape) data. Moreover, extant methods for text-to-shape generation have limited shape diversity and fidelity. We introduce TextCraft, a method to address these limitations by producing high-fidelity and diverse 3D shapes without the need for (text, shape) pairs for training. TextCraft achieves this by using CLIP and using a multi-resolution approach by first generating in a low-dimensional latent space and then upscaling to a higher resolution, improving the fidelity of the generated shape. To improve shape diversity, we use a discrete latent space which is modelled using a bidirectional transformer conditioned on the interchangeable image-text embedding space induced by CLIP. Moreover, we present a novel variant of classifier-free guidance, which further improves the accuracy-diversity trade-off. Finally, we perform extensive experiments that demonstrate that TextCraft outperforms state-of-the-art baselines.

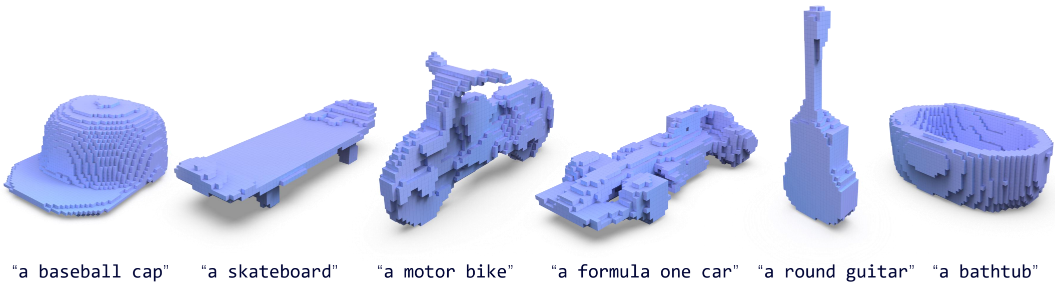

Figure 1: We propose a new zero-shot text-to-shape generation method called TextCraft. The generated shapes are high-quality and can reflect the semantic meaning from the text input in ShapeNet55.

1 Introduction

In recent years, there has been rapid progress in generating images from natural language prompts (Ramesh et al., 2021; 2022; Rombach et al., 2022a), driven by large databases of paired (text, image) data. These natural-language-based image synthesizers have had broad impact in domains ranging from human–robot interaction (Shridhar et al., 2021) to visual creativity, for instance allowing high-fidelity image creation and manipulation (Ramesh et al., 2022). Natural language interfaces could also be valuable in other hitherto unexplored domains such as 3D modeling which has a high skill barrier to entry. Unfortunately, developing text-based 3D shape generators is challenging because of the difficulty of obtaining paired (text, 3D shape) data at large scale. Some prior work has attempted to address this problem (Chen et al., 2018; Liu et al., 2022; Mittal et al., 2022; Fu et al., 2022), but have been limited to a small number of categories for which such data is available.

A promising way around this data bottleneck is to use weak supervision from large-scale vision/language models such as CLIP (Radford et al., 2021). One instantiation of this approach is to directly optimize a 3D representation such that (differentiable) renderings of it are similar to an input text prompt when projected into CLIP space. Prior work has applied this approach for stylizing 3D meshes (Michel et al., 2022) and for creating abstract, "dreamlike" objects represented as neural radiance fields (Jain et al., 2022). Neither produces realistic object geometry, and both require expensive optimization to generate a new 3D output. Another approach, more in line with the text-to-image generators (Ramesh et al., 2021; 2022), is to train a text-conditional generative model. The CLIP-Forge system (Sanghi et al., 2022) builds such a model without paired (text,shape) data by using rendered images of shapes at training time and leveraging the CLIP embedding space to bridge the gap between images and text at test time. It demonstrates compelling zero-shot generation abilities but produces low-fidelity shapes (32^3 occupancy grids) that do not capture the full diversity of shapes in the training data distribution.

In this paper, we address the limitations of previous work: needing **(text, shape) pairs** or producing **low-fidelity outputs** that lack **shape diversity**. Our method, TextCraft, is a text-conditional 3D shape generative model that outputs diverse and high-fidelity 3D shapes using only CLIP as supervision. To learn without (text,shape) pairs TextCraft learns to produce 3D shapes of common object categories by leveraging CLIP embeddings of prompts and rendered images of the shapes. To achieve high-fidelity outputs, TextCraft adopts a multi-resolution approach: it first generates a low-resolution latent grid representation and then upscales it to higher resolution before decoding final geometry. To achieve diversity, these latent representations are discrete: they are obtained using a vector quantization scheme that avoids posterior collapse. TextCraft generates these latent grids via a masked transformer architecture, whose output quality and diversity is further improved by a novel annealed variant of classifier-free guidance (Ho & Salimans, 2022). We demonstrate both quantitatively and qualitatively that TextCraft outperforms other methods on standard metrics of generative model quality and diversity. To sum up, we contribute:

- TextCraft, a multi-resolution text-conditional shape generative model that achieves both high quality and diversity without the need for (text,shape) pairs.
- A novel variant of classifier-free guidance for conditional generative models, using an annealed guidance schedule to achieve better quality for a given diversity level.

2 Related Work

Neural Discrete Representation. Discrete latent spaces for deep generative models(Oord et al., 2017) were first proposed for image generation as a method to address the"posterior collapse" problem while improving image quality for variational autoencoders. For images, the latent space is structured as 2D grids of discrete latent variables. Further works introduced multi-scale hierarchical versions (Razavi et al., 2019; Dhariwal et al., 2020) which further improved generative capabilities. Recently, within the 3D domain as well, models have adopted neural discrete representations (Yan et al., 2022; Mittal et al., 2022) such as discretized voxel and implicit grids. In this work, we take inspiration from hierarchical VQ-VAEs (Razavi et al., 2019; Dhariwal et al., 2020) and propose an architecture capable of generating hierarchical discrete representations for high quality 3D shapes.

Latent Generative Models. Generating high quality 3D shapes requires intensive computing resources. In recent years, latent generative models have been widely adopted, because these models can generate low-dimensional latent representations more effectively. These latent representations can be used to efficiently generate images and shapes, as shown in many works (Chen & Zhang, 2019; Ibing et al., 2021) which use GANs. However, one disadvantage of GANs is that they tend to suffer from training instability and mode collapse. Other types of models like flow-based models (Yang et al., 2019; Sanghi et al., 2022) have been proposed, but they yield low sample quality which is inferior to GAN-based models. Recent works use diffusion (Gu et al., 2022; Rombach et al., 2022b) or masking models (Chang et al., 2022) on the latent space which increase the inference efficiency while giving quality outputs. Building on these works, we propose a hierarchical latent generative model that can further improve the quality and diversity of shape generations.

Text-to-Shape Generation. Text-to-shape generation has gained momentum in recent years. Recent works(Chen et al., 2018; Liu et al., 2022; Mittal et al., 2022; Fu et al., 2022) use supervised text-shape pairs to generate shapes effectively using natural language. However, a major drawback is the availability of text-shape paired datasets, forcing these methods to only generate shapes from a few categories. To solve this, several recent works have successfully leveraged the prior knowledge in image-text latent space of CLIP (Radford et al., 2021) by converting the shapes into image. One line of work uses differentiable renderers (Michel et al., 2022; Jain et al., 2022) and the other learns a mapping from image to shape space (Sanghi et al., 2022). Although these methods are effective, they suffer from the long optimization time and poor quality of generated shapes. Our method is able to generate more diverse shapes of higher quality within a relatively short inference time.

3 Method

Our goal is to generate 3D shapes which conform to a natural language input prompt, and to do this without relying on text labels for 3D shapes at training time. In lieu of text labels training shapes, we will instead use the prior knowledge embedded in the CLIP vision/language model (Radford et al., 2021). For each shape in our training set (D), we assume we have a set of images $\{\mathbf{I}_r | r \in R\}$ of the shape rendered from a set of views R. We also assume two volumetric occupancy grid representations of the shape, $\mathbf{V}_{32}$ and $\mathbf{V}_{64}$, voxelized at resolutions of 32^3 and 64^3, respectively.

Figure 2 illustrates the components of our TextCraft model. Training the model involves three stages (Figure 2 top). In the first stage, we train two vector-quantized variational autoencoders (VQ-VAEs), one to model the coarse-resolution voxel grid $\mathbf{V}_{32}$ and the other to model the fine-resolution voxel grid $\mathbf{V}_{64}$. The encoders of these VQ-VAEs produce discrete latent grids $\mathbf{E}_{32}$ and $\mathbf{E}_{64}$, respectively. In the second stage, we train a coarse transformer model $T_c(.)$ which takes as input a masked quantized latent grid $\mathbf{E}_{32}$ and predicts the original unmasked version. This transformer is conditioned on the CLIP image embeddings for the rendered images $\{\mathbf{I}_r\}$. Finally, in the third stage, we train a *fine* transformer model ($T_f(.)$) takes a masked quantized latent grid $\mathbf{E}_{64}$ and predicts the original unmasked version; this network is conditioned on the corresponding coarse latent grid $\mathbf{E}_{32}$ via cross attention.

During inference (Figure 2 bottom), TextCraft starts with a fully-masked coarse latent grid $\mathbf{E}_{32}$ and uses the coarse transformer T_c to produce an unmasked version via confidence-based iterative decoding scheme (Chang et al., 2022). While this network was conditioned on CLIP image embeddings of the target shape at training time, we can leverage the interchangeability of text and image embeddings in CLIP space to instead condition it on the CLIP text embedding of the input text prompt. This unmasked coarse latent grid is then used to condition the fine transformer T_f, which takes a fully-masked fine latent grid $\mathbf{E}_{64}$ as input and produces an unmasked version via confidence-based iterative sampling. Finally, this unmasked fine latent grid is passed to the 64^3 VQ-VAE decoder to produce output geometry in the form of a 64^3 volumetric occupancy grid.

3.1 Training Stage 1: Voxel VQ-VAEs

The goal of the first training stage is to learn a low dimensional latent space which can effectively reconstruct shapes but also excel at generating novel ones. For this, we use a vector-quantized formulation of the variational autoencoder (VQ-VAE) (Oord et al., 2017), because (a) it has been shown to capture more modes of the data distribution (i.e. to avoid "posterior collapse"), and (b) its discrete representation is amenable to modeling with transformers in the later stages of our architecture. We train two separate VQ-VAEs for the 32^3 and 64^3 resolution voxel representations of our training shapes. For both autoencoders, we use ResNet-based (He et al., 2016) volumetric CNN encoders and a vector quantization layer which maps down-sampled volumes into a discrete space ($\mathbf{E}_{32}$ and $\mathbf{E}_{64}$ for the two resolution levels, respectively) by indexing into an embedding codebook. The decoders first map the indices in $\mathbf{E}_{32}$ and $\mathbf{E}_{64}$ to their respective embeddings in the codebook and then uses ResNet-based 3D convolutions to decode the output shape. We train the network using the mean squared error loss (MSE) with the commitment loss as specified in (Oord et al., 2017). However, we replace the codebook loss from (Oord et al., 2017) with a moving average of the encoder output to update the codebook embeddings. It has been found that using moving averages allows the network to converge faster instead of using codebook loss (Razavi et al., 2019; Łańcucki et al., 2020).

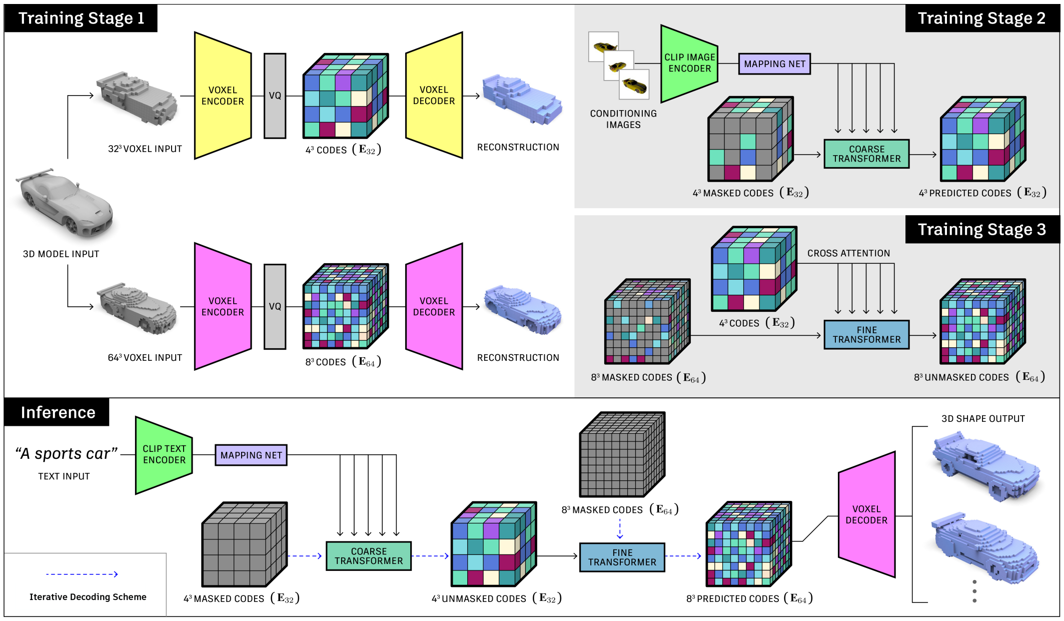
Training Stage 1
3D MODEL INPUT
32^3 VOXEL INPUT
VOXEL ENCODER
VQ
4^3 CODES $(\mathbf{E}_{32})$
VOXEL DECODER
RECONSTRUCTION
64^3 VOXEL INPUT
VOXEL ENCODER
VQ
8^3 CODES $(\mathbf{E}_{64})$
VOXEL DECODER
RECONSTRUCTION
Training Stage 2
CONDITIONING IMAGES
CLIP IMAGE ENCODER
MAPPING NET
4^3 MASKED CODES $(\mathbf{E}_{32})$
COARSE TRANSFORMER
4^3 PREDICTED CODES $(\mathbf{E}_{32})$
Training Stage 3
4^3 CODES $(\mathbf{E}_{32})$
CROSS ATTENTION
8^3 MASKED CODES $(\mathbf{E}_{64})$
FINE TRANSFORMER
8^3 UNMASKED CODES $(\mathbf{E}_{64})$
Inference
"A sports car"
TEXT INPUT
CLIP TEXT ENCODER
MAPPING NET
4^3 MASKED CODES $(\mathbf{E}_{32})$
COARSE TRANSFORMER
4^3 UNMASKED CODES $(\mathbf{E}_{32})$
8^3 MASKED CODES $(\mathbf{E}_{64})$
FINE TRANSFORMER
8^3 PREDICTED CODES $(\mathbf{E}_{64})$
VOXEL DECODER
3D SHAPE OUTPUT
Iterative Decoding Scheme

Figure 2: The TextCraft architecture during training (top) and inference (bottom). TextCraft is trained in three stages. In Stage 1, we train two separate VQ-VAE models for 32^3 and 64^3 resolution voxel grids. In Stage 2 we train a coarse transformer to generate low resolution VQ-VAE latent grids $\mathbf{E}_{32}$ conditioned on a CLIP embedding. In Stage 3, we train a fine transformer to perform super resolution on these latent grids. During inference, a text prompt is passed through the CLIP text encoder and used to condition the coarse transformer to generate a coarse latent grid $\mathbf{E}_{32}$. This coarse grid is then used to condition the fine transformer to generate a fine latent grid $\mathbf{E}_{64}$ that is then used to generate the output shape using the 64^3 VQ-VAE decoder from Training Stage 1.

3.2 Training Stage 2: Coarse Transformer

In this stage, our goal is to train a transformer T_c that can generate a coarse quantized latent grid $\mathbf{E}_{32}$ from an input text prompt. Inspired by the recent success of masking approaches (Chang et al., 2022) and diffusion models (Nichol et al., 2021; Gu et al., 2022; Rombach et al., 2022b), we formulate this task as a conditional unmasking task: given an input masked latent grid and a conditioning vector, the transformer should produce an unmasked latent grid.

The conditioning vector $\mathbf{c}$ is the mechanism by which the input text prompt influences the generative process. Since we assume no text labels for our training shapes, we leverage the interchangeability of text and images in the CLIP embedding space: training on CLIP embeddings produced from image (which we can obtain by rendering a training shape) while testing on CLIP embeddings produced from text prompts. For training, we compute the condition vector $\mathbf{c}$ for a given training shape by passing one of its renderings $\mathbf{I}_r$ through the ViT-based (Dosovitskiy et al., 2020) CLIP image encoder $f_I(\cdot)$ (whose weights are frozen). As the goal is to eventually use text embeddings instead of image embeddings during inference, we add Gaussian noise to better align the text and image embeddings (Zhou et al., 2021):

$$\hat{\mathbf{c}} = f_I(\mathbf{I}_r) + \gamma \cdot \epsilon \cdot \|f_I(\mathbf{I}_r)\|_2 / \|\epsilon\|_2, \quad \text{where} \quad \epsilon \sim \mathcal{N}(0, 1) \tag{1}$$

In the above equation, γ controls the level of perturbation. To obtain the final condition vector, $\mathbf{c}$, we divide $\hat{\mathbf{c}}$ by its norm ($\hat{\mathbf{c}}/\|\hat{\mathbf{c}}\|_2$). The condition vector is then passed through a common mapping network which is a multi-layer perceptron (MLP) to obtain $\tilde{\mathbf{c}}$. The condition vector $\mathbf{c}$ influences the transformer by predicting the affine transform parameters of each transformer block's layernorm. This is an important design choice which we investigate in an ablation study (Section 4). We also drop the condition vector (i.e. replace with the null embedding $\mathbf{c} = 0$) with a $p\%$ probability during training for classifier-free guidance (Ho & Salimans, 2022).

To train the coarse transformer T_c, we first select a masking ratio $\rho \in [0, 1]$ and randomly replace $\rho\%$ of the indices in $\mathbf{E}_{32}$ with special "mask" tokens to obtain $\mathbf{Y}_{32}$. The training objective is then to minimize the negative log-likelihood of the coarse grid given the masked grid and the condition vector $\mathbf{c}$. However, we find that using this objective does not solve the issue of accumulation of error during inference-time sampling (i.e. autoregressive drift), since the network sees only ground truth tokens (and not its own predicted samples) during training. To alleviate this issue, we take inspiration from the NLP literature (Savinov et al., 2021a) and propose using a similar two-step unrolled training loss:

$$\mathcal{L} = - \mathop{\mathbb{E}}_{\mathbf{E}_{32} \in D} \left[\sum_{i}^{N} \Big(\log p(E_{32}^{i} | \mathbf{Y}_{32}, \mathbf{c}) + \log p(E_{32}^{i} | \tilde{\mathbf{Y}}_{32}, \mathbf{c}) \Big) \right] \tag{2}$$

In the above equation, $\tilde{\mathbf{Y}}_{32}$ is calculated by masking a sample from $T_c(\cdot|\mathbf{Y}_{32}, \mathbf{c})$, i.e. the network's own prediction. Note that this differs from Savinov et al. (2021a), as we predict the entire unmasked grid at each time step.

3.3 Training Stage 3: Fine Transformer

The goal of the fine transformer T_f is to take an unmasked coarse latent grid and produce a higher-resolution latent grid—essentially, it performs super-resolution in the discrete latent space. As with the coarse transformer T_c, we formulate this transformer's learning task as unmasking the masked indices of $\mathbf{E}_{64}$. Where T_c is conditioned on a CLIP embedding via layer norm parameters, T_f is instead conditioned on the unmasked coarse grid $\mathbf{E}_{32}$ via cross attention. Empirically, we find that this network performs best when this conditioning comes from the predictions of the coarse transformer instead of the ground-truth coarse latent grid, i.e. using $\hat{\mathbf{E}}_{32} \sim T_c(\cdot|\mathbf{Y}_{32}, \mathbf{c})$ instead of $\mathbf{E}_{32}$. This intuitively makes sense as we use the actual results observed during sampling. Moreover, we find that additionally conditioning on CLIP image features hurts performance, indicating most of the relevant information is already present in the coarse latent grid.

3.4 Inference

At inference time, we first convert a given input text prompt into a CLIP text embedding using the CLIP text encoder f_T. We exploit the interchangeability property of the CLIP embedding space (Sanghi et al., 2022), using the output of f_T in place of the output of the CLIP image encoder f_I in the computation of the coarse transformer's condition vector $\mathbf{c}$ (Equation 1). The initial input to the coarse transformer is a completely masked latent grid $\mathbf{E}_{32}$. We use the iterative decoding scheme (Chang et al., 2022) to slowly unmask the grid over a sequence of T steps. At each time step t, we condition the coarse transformer with $\mathbf{c}$ and the predicted output latent grid from the previous time step. We then take the output from the coarse transformer and mask all other tokens except the previously predicted tokens and the most confident token predictions at this time step by looking at their probability outputs. We repeat this process until the process unmasks all the tokens, which is ensured by a cosine masking schedule (Chang et al., 2022).

During this iterative decoding scheme, we apply a new variant of classifier-free guidance (Ho & Salimans, 2022). Classifier-free guidance extrapolates an unconditional sample in the direction of a conditional sample, where the amount of extrapolation is controlled by a *guidance scale* parameter. Varying this parameter results in varying the tradeoff between fidelity to the conditioning vector and sample diversity. As proposed in the original paper, the guidance scale is kept constant for all time steps. To improve the accuracy/diversity tradeoff, we propose instead to vary the guidance scale over time according to an annealing schedule. The intuition is that during the initial steps of sampling (when the input to the coarse transformer is mostly masked indices), a larger guidance scale is important to keep the network "on task"; later on in the sampling process, a lower guidance scale can help produce more sample diversity. The overall equation is given below:

$$\hat{p}(\mathbf{E}_{32}^{t} | \mathbf{E}_{32}^{t-1}, \mathbf{c}) = p(\mathbf{E}_{32}^{t} | \mathbf{E}_{32}^{t-1}, \mathbf{0}) + a(t)(p(\mathbf{E}_{32}^{t} | \mathbf{E}_{32}^{t-1}, \mathbf{c}) - p(\mathbf{E}_{32}^{t} | \mathbf{E}_{32}^{t-1}, \mathbf{0})) \tag{3}$$

Here, $a(t)$ is the guidance scale annealing schedule, which is continuous and monotonically decreasing. We can recover the original classifier-free guidance scheme by setting $a(t) = k$ for some constant k. In this paper, we experimentally evaluate several annealing schedules (Chang et al. (2022)), including linear, cosine, and square root functions.

Finally, the unmasked coarse latent grid $\mathbf{E}_{32}$ is used to condition the fine transformer T_f, which uses the iterative decoding scheme to unmask an initially fully-masked fine latent grid $\mathbf{E}_{64}$. The unmasked fine grid is then passed to the 64^3 VQ-VAE decoder to obtain the final voxelized output shape.

4 Experiments

In this section, we report experimental results to evaluate the generation quality, diversity, and class accuracy of TextCraft. We provide the details for the hyperparameters and experimental details in the Appendix. We run the experiments 3 times for each of the below sections and report the mean in each case, except the final comparisons with Clip-Forge where we use the best seed. Additional results can also be found in the Appendix.

Dataset. We conduct our experiments on two subsets of the ShapeNet(v2) dataset (Chang et al., 2015). The first subset, *ShapeNet13*, contains 13 categories from ShapeNet as used in (Choy et al., 2016; Mescheder et al., 2019). We use the same train/test split as specified in (Mescheder et al., 2019). Our second subset is *ShapeNet55* which contains all 55 ShapeNet categories. For ShapeNet55, we render images as described in (Choy et al., 2016) and the training dataset contains 51784 datapoints whereas the test set contains 6101 datapoints.

Evaluation Metrics. We use Mean Square Error (**MSE**) and Intersection over Union (**IoU**) as metrics for reconstruction accuracy to compare different hyperparameters in Stage 1. We calculate **MSE** and **IoU** on the ShapeNet(v2) test set at 32^3 voxel resolution as in Mescheder et al. (2019). For generative capabilities, we use Fréchet Inception Distance (**FID**) (Heusel et al. (2017)) to evaluate diversity and Classifier Accuracy (**Acc**) to evaluate how well the generated shapes match a given text query. These metrics follow (Sanghi et al., 2022) where they first generate single mean shapes for 234 predetermined text queries and then pass all shapes through a classifier to measure **Acc**. The latent space of this classifier is also used for **FID**.

Baseline. We compare the performance of our method against CLIP-Forge (Sanghi et al., 2022) and DreamFields (Jain et al., 2022), which are currently state of the art for zero-shot text-to-shape generation. In CLIP-Forge, they only report results on single shape generation for the predetermined 234 text queries which we refer to as CF-MS. The single shape is generated using the mean of the prior which is the Gaussian distribution. Note that this does not capture the diversity of multiple shapes generated given a text query. To capture results on more shape generations, we sample 32 shapes instead of one using either a Gaussian (CF-G), truncated Gaussian (CF-TG) or clipped Gaussian (CF-CG) distribution. We follow the same protocol for our method as we do not have the ability to generate single mean shape. We only compare qualitatively with DreamFields as it does not use prior knowledge from the shape dataset and it would be unfair to report those results.

4.1 Evaluating Shape Diversity and Accuracy

In this section, we first quantitatively evaluate diversity and accuracy of a given shape matching a given text query on the ShapeNet13 dataset. We compare with CLIP-Forge using the **Acc** metric and **FID** metric. The results are shown in Table 1. The first four columns represent CLIP-Forge and different sampling techniques. The other columns represent our method with different annealing scale strategies. Two major things can be observed. First, all variants of our method outperform CLIP-Forge significantly. This indicates that the method produces more diverse and higher fidelity shapes of increasing accuracy. Second, it can be seen that annealing strategies for guidance scale gives a better accuracy versus diversity trade-off then constant scale guidance. This is discussed more below.

We qualitatively also compare our method to CLIP-Forge and DreamFields. The results are shown in Figure 3. It can be observed that our method generates higher quality shapes (view ”a machine gun”), with more detail (view ”an office chair”) and higher diversity (view ”a jet”). Clip-Forge usually produces the same shape with small variations (view ”a rectangular table”). We also note that DreamFields produces very abstract results which might not be useful in many applications.

Table 1: Comparisons of CLIP-Forge(CF) baseline (across different sampling strategies) with TextCraft (TC) on Accuracy(ACC) and FID. Accuracy is based on the match between the prediction of a pretrained voxel classifier and the category label.

Method	CF-*MS*	CF-*G*	CF-*TG*	CF-*CG*	TC-*const.*	TC-*sqrt*	TC-*linear*	TC-*cosine*
FID ↓	2425.25	2233.48	2141.61	2100.67	1821.78	**1480.11**	1629.51	1725.63
ACC ↑	83.33	62.81	68.71	71.11	86.59	87.08	**87.50**	87.27

Table 2: Left Table: Effect of varying the *Noise* parameter . Center Table: Effect of varying number of layers (L) in the mapping network . Right Table: Comparison with baselines on super resolution

γ	**FID**↓	**Acc**↑
×	1720.02	64.87
0.5	1764.98	75.73
0.8	1484.61	77.41
1.0	1703.38	79.09
1.2	**1447.91**	**79.63**
1.5	1478.17	78.47

L	**FID**↓	**Acc**↑
0	2874.87	62.72
1	1716.73	78.70
2	1518.97	79.17
3	1447.91	**79.63**
4	1532.50	77.16
5	**1424.46**	76.09

Method	**FID**↓	**Acc**↑
3D-UNet	2056.92	86.65
TT-Net	2196.96	77.92
TextCraft	1910.28	86.85

Finally, we also show results on ShapeNet55 in the last row of Figure 3. We could not get CLIP-Forge to produce sensible shapes for most text queries on ShapeNet55 which we attribute to the data imbalance issue of ShapeNet55 whereas our method produces high quality shapes.

4.2 Major Components of Stage 2 Training

Effect of Noise Parameter. We investigate the effects of different noise levels(γ) added to the image condition vector during training. The results are shown in Table 2, left. The first row has no noise added whereas the remaining rows show varying levels of noise. We keep the number of mapping layers fixed in this experiment. We observe that adding Gaussian noise drastically improves both the diversity and accuracy of generation with the optimal noise parameter being around 1.2. These results indicate that adding noise during training helps with the alignment between the text features observed during inference and the image features observed during training.

Size of Mapping Network. We next probe the importance of the mapping network. We show the results in the center table of Table 2, where L represents the number of layer of the mapping function. For 0 layers we directly project conditional embeddings to the layernorm parameters using a linear layer. We make two observations from the results: 1) A common mapping network improves both the accuracy and FID. 2) Increasing the number of mapping network layers beyond a certain number decreases accuracy.

Classifier-Free Guidance. We next explore the relationship between dropping out image conditioning at $\rho\%$ during training and using classifier-free guidance during generation. In Table 3, the columns indicate the variation in scale parameter. It can be seen that with the increase in scale, accuracy typically increases whereas there is a decrease in FID. This indicates the method is giving more accurate results on a given text query while sacrificing diversity. Moreover, a low dropout rate (5-15%) gives a good trade-off between accuracy and diversity.

Step-Unrolled Training (SUT). Finally, to further improve the accuracy we investigate the use of step-unrolled training (Savinov et al., 2021b). The results are shown in the last row of Table 3. From the table it can be observed that step unroll training enables higher accuracy across all scales. This indicates that the model learns to unmask samples it would encounter during sample time.

4.3 Annealing Strategy for Guidance Scale Parameter

An important idea we propose in this paper is a scale annealing technique for classifier-free guidance. We employ classifier-free guidance to identify a better accuracy versus diversity trade-off with

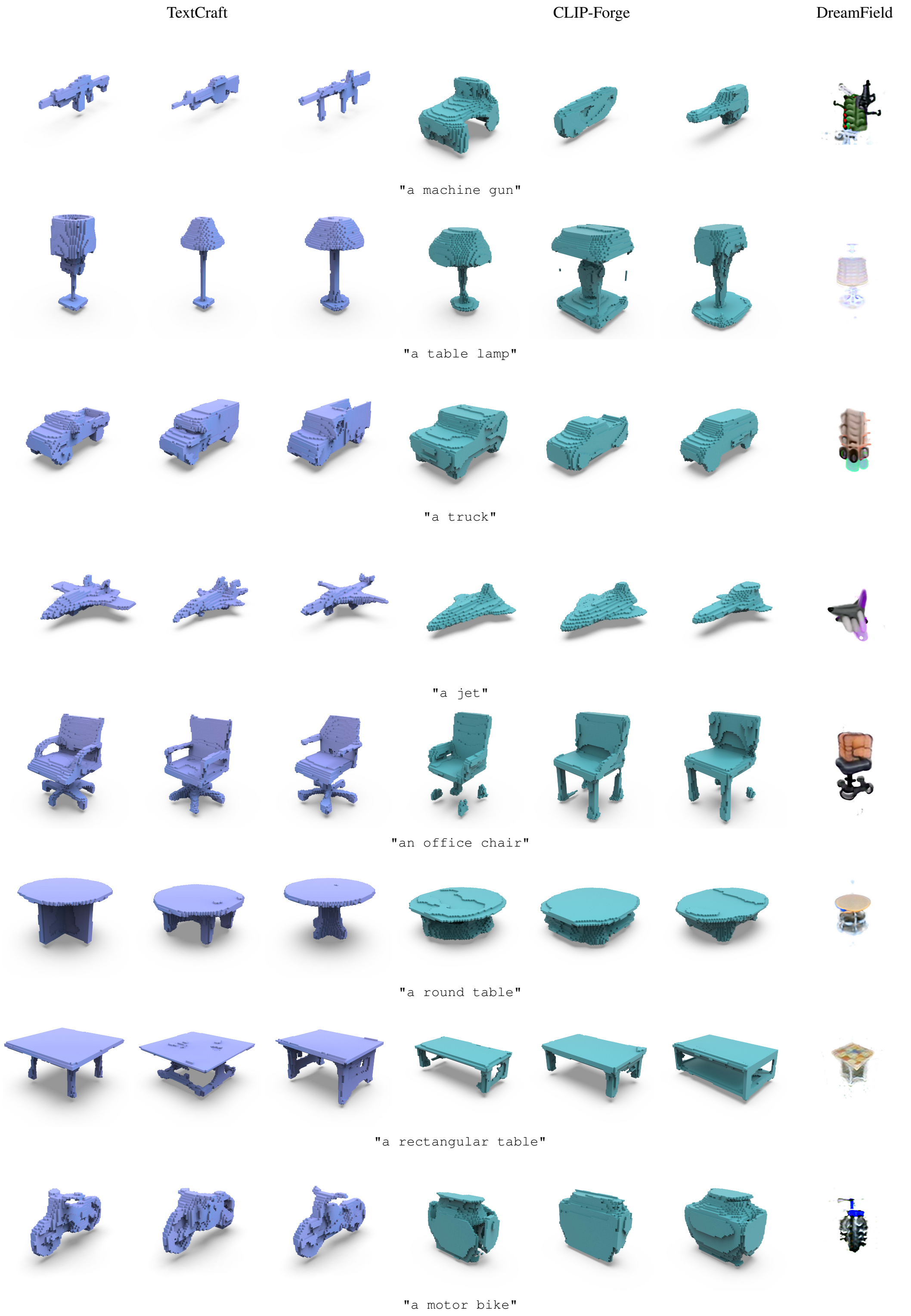

Figure 3: Qualitative comparison among TextCraft (rendered in purple), CLIP-Forge (Sanghi et al., 2022)(rendered in green), and DreamField (Jain et al., 2022) on text-conditioned generation.

different annealing schedules: constant, linear, cosine and square root. A constant schedule refers to the use of the same scale value across all time steps as proposed in Ho & Salimans (2022); Nichol

Table 3: Classifier-Free Guidance and Step-Unrolled Training (SUT) experiment results. *p* represents the dropout of conditioning. SUT indicates the use of Step-Unrolled Training. The remaining columns indicate the variation in scale parameter.

ρ%	SUT	3		2.5		2		1.5		1	
		FID↓	**Acc**↑	**FID**↓	**Acc**↑	**FID**↓	**Acc**↑	**FID**↓	**Acc**↑	**FID**↓	**Acc**↑
5	×	2059.0	85.73	1970.1	85.47	1790.9	85.43	1536.5	83.98	1227.5	79.17
10	×	1893.2	84.46	1821.2	84.50	1684.3	83.41	1522.4	82.34	1348.4	77.76
15	×	2086.9	85.03	1964.7	84.99	1851.9	84.36	1660.9	83.14	1485.5	78.19
20	×	2062.5	83.39	1972.3	82.95	1892.9	82.74	1733.3	81.13	1566.6	75.94
5	✓	2039.8	87.69	2011.1	87.39	1811.8	87.40	1678.6	86.24	1517.9	82.18

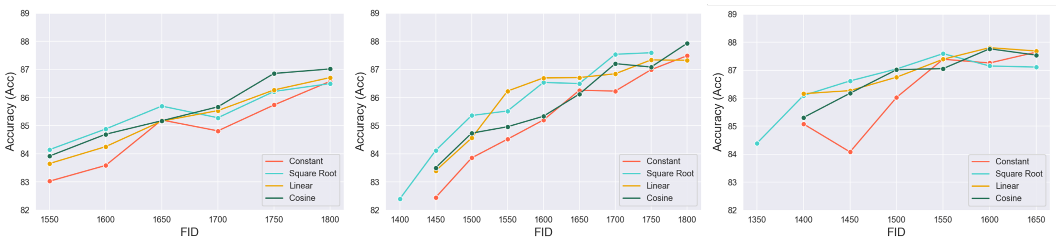

Figure 4: **FID** and **Acc** results with different classifier-free guidance scale annealing strategies (constant, cosine, square root, linear) across three different runs (different seeds) of the TextCraft Stage 2 Transformer.

et al. (2021). To determine which annealing technique works the best, we fix the FID values and plotted the accuracy at each fixed FID value (Figure 4). As different scale parameters give different FID values, we conducted an extensive grid search over the starting scale parameter. Note that finding the exact FID is not always feasible, so we pick the closest FID. Figure 4 shows results of FID versus accuracy on three different runs of the Stage 2 Transformer. We find that across all three runs the accuracy is typically lower for a given FID in the case of a constant schedule, when compared to other schedules. This is especially the case at the lower range of FID values. The results indicate that having a large scale at the beginning of sampling is more important than later stages especially in use cases where diversity is paramount.

4.4 Super-Resolution

Finally, we investigate the importance of hierarchy based super-resolution. We compare our method with 2 baselines. In the first baseline, we directly use the 32^3 resolution results from coarse transformer and use a 3D U-NET (Ronneberger et al., 2015) based super-resolution network to translate from 32^3 to 64^3. For the second baseline, we train a transformer directly on 64^3 VQ-VAE which is conditioned on text features instead of coarse resolution grid. We refer to this as TT-Net. The results are shown in Table 2, right. It can be seen from the table that indeed latent based super-resolution outperforms the baselines in both accuracy and diversity.

5 Conclusion

We present TextCraft, a text-to-3D-shape generation method that is capable of producing shapes of high fidelity and diversity without the need for (text, shape) pairs during training. To achieve this, TextCraft leverages CLIP and implements super-resolution in a discrete latent space with a hierarchical architecture and a novel annealed variant of classifier-free guidance on a mask-based model. We validate TextCraft by comparing it with a number of baselines in terms of FID and accuracy, finding that TextCraft is the new state of the art for this problem. In experimenting with different guidance scale scheduling, we find that constant scale scheduling did not always work the best, an important finding for the diffusion modeling community that may improve generation quality. Our paper takes an step in diversifying and improving the quality of 3D text-to-shape generation outcomes.

REFERENCES

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository, 2015. URL `http://arxiv.org/abs/1512.03012`. cite arxiv:1512.03012.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11315–11325, 2022.

Kevin Chen, Christopher B Choy, Manolis Savva, Angel X Chang, Thomas Funkhouser, and Silvio Savarese. Text2shape: Generating shapes from natural language by learning joint embeddings. In *Asian Conference on Computer Vision*, pp. 100–116. Springer, 2018.

Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5939–5948, 2019.

Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pp. 628–644. Springer, 2016.

Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Rao Fu, Xiao Zhan, Yiwen Chen, Daniel Ritchie, and Srinath Sridhar. Shapecrafter: A recursive text-conditioned 3d shape generation model, 2022. URL `https://arxiv.org/abs/2207.09446`.

Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10696–10706, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Moritz Ibing, Isaak Lim, and Leif Kobbelt. 3d shape generation with grid-based implicit functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13559–13568, 2021.

Ajay Jain, Ben Mildenhall, Jonathan T Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 867–876, 2022.

Zhengzhe Liu, Yi Wang, Xiaojuan Qi, and Chi-Wing Fu. Towards implicit text-guided 3d shape generation. *arXiv preprint arXiv:2203.14622*, 2022.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13492–13502, 2022.

Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. Autosdf: Shape priors for 3d completion, reconstruction and generation. *arXiv preprint arXiv:2203.09516*, 2022.

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, 2021. URL `https://arxiv.org/abs/2112.10741`.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *arXiv preprint arXiv:1711.00937*, 2017.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pp. 8821–8831. PMLR, 2021.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

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 (CVPR)*, pp. 10684–10695, June 2022a.

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*, pp. 10684–10695, 2022b.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18603–18613, 2022.

Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. Step-unrolled denoising autoencoders for text generation. *arXiv preprint arXiv:2112.06749*, 2021a.

Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. Step-unrolled denoising autoencoders for text generation, 2021b. URL `https://arxiv.org/abs/2112.06749`.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Proceedings of the 5th Conference on Robot Learning (CoRL)*, 2021.

Xingguang Yan, Liqiang Lin, Niloy J Mitra, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Shapeformer: Transformer-based shape completion via sparse representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6239–6249, 2022.

Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4541–4550, 2019.

Yufan Zhou, Ruiyi Zhang, Changyou Chen, Chunyuan Li, Chris Tensmeyer, Tong Yu, Jiuxiang Gu, Jinhui Xu, and Tong Sun. Lafite: Towards language-free training for text-to-image generation. *arXiv preprint arXiv:2111.13792*, 2021.

Adrian Łańcucki, Jan Chorowski, Guillaume Sanchez, Ricard Marxer, Nanxin Chen, Hans J. G. A. Dolfing, Sameer Khurana, Tanel Alumäe, and Antoine Laurent. Robust training of vector quantized bottleneck models, 2020. URL `https://arxiv.org/abs/2005.08520`.