Heretic Ventures Furniture Generation Report

Part 1

1:42 p.m. – Started

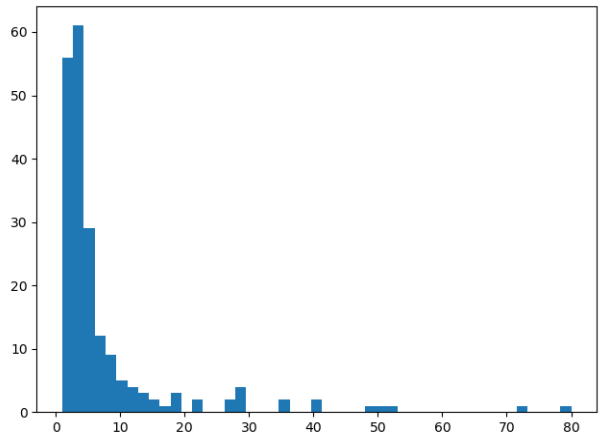
Exploratory Data Analysis

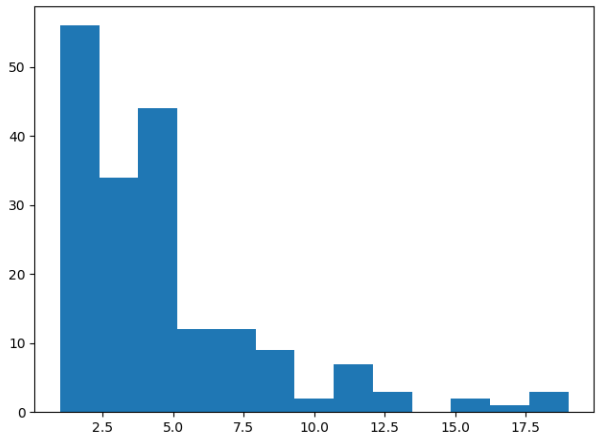
I removed columns filled with white color from "Furniture-Text Captions.xlsx" and converted it to a comma delimited .csv for convenience. I also converted it to UTF-8 because of some pandas decoding exceptions and manually removed empty end lines caused by Excel conversion.

The example prompts contain 1 big instance "living room" (160 images) and smaller instances like "Rectangular Coffee Table with Solid Wood top and Solid Wood Iron base" (9 images).
To understand if we have some outliers in the number of images, let us perform an EDA:

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv(r'D:\Datasets\Case Data\Furniture - Text Captions.csv')
counts = df.groupby('text').nunique().sort_values(by='file_name')
plt.hist(counts, bins='auto')
plt.show()
plt.hist(counts[counts < 20], bins='auto')
plt.show()
```





```
text
armchair with round and flared arms                          1
genuine leather armchair with arms                           1
100% polyester sectional with round and recesse...           1
round coffee table with solid + manufactured wo...           1
round coffee table with glass top and stainless...           1
                                                           ...
genuine leather loveseat with square arms                   76
genuine leather armchair with square arms                   86
polyester blend loveseat with square arms                   89
100% polyester sectional with square arms                  107
polyester blend sectional with square arms                 118

[217 rows x 1 columns]
```

We now see that we have 252 instances. Empirically 5-10-20 images are enough for methods like Dreambooth. Therefore in future, we need to use augmentations or synthetic data for underrepresented instances and limit the number of images for overrepresented instances. But for now, we can skip this because it is proof of concept. It will randomly sample images within an instance during training – not make us any problems.

What we actually can do at this step is to sample test prompts “uniformly” based on class frequency. After merging rare classes (armchair -> chair, ottoman -> table, couch -> sofa), there are 7 classes: loveseat, sectional, storage, table, chair, chaise, sofa. Like that:

```python
# Generate test prompts
prompts = df.merge(counts, left_on='instance', right_index=True)
# It sometimes generates too rare instance numbers because they are uniform
# Also we want as different text prompts as possible, so we need to sample several
for _ in range(30):
    best_uniformity = float('-inf')
    best_prompts = None
    best_diff = None
    for i in tqdm(range(5000)):
        # Sample 1 per class
        prompts1 = prompts.groupby(['class']).sample()

        diffs = np.diff(sorted(prompts1['count']))
        # We want to know how different number of instances affects the generation results,
        # so we try to seek a uniform distribution of instance numbers
        uniformity = np.mean(diffs) - np.std(diffs)
        if uniformity > best_uniformity:
            best_prompts = prompts1.copy(deep=True)
            best_uniformity = uniformity
            best_diff = diffs
    best_prompts = best_prompts.sort_values('count')
    print(best_prompts['count'].values)
    print('\n'.join(best_prompts['instance']))
```

It will help us understand how the model deals with under- and overrepresented instances. The final list of test prompts will be (the #4–#10 instance distribution is [2 10 18 29 40 50 72]):

1. A living room with two leather armchairs.
2. A living room with an armchair with round and flared arms.
3. A living room with a concrete wall, and a round coffee table with marble top and metal.
4. A living room with a square coffee table with solid + manufactured wood top and iron base.
5. An armchair with arms in a living room.
6. A 100% polyester loveseat with round arms in a living room.
7. A living room with a round coffee table with solid wood top and solid wood base.
8. A living room containing a velvet sectional with square arms.
9. A living room with a genuine leather loveseat with square arms.
10. A living room with a red brick wall, and a 100% polyester sectional with square arms.

Evaluation:

In this particular case, we can use the CLIP score, Frechet Inception Distance (FID), and some aesthetics models like recent Google Research's VILA.
https://paperswithcode.com/task/image-quality-assessment
https://paperswithcode.com/task/aesthetics-quality-assessment
https://colab.research.google.com/github/google-research/google-research/blob/master/vila/tfhub_inference.ipynb#scrollTo=C-Vgd9Rh4yWV

2:30 p.m.–3:30 p.m. – Late lunch

Questions:

1. Please explain which text-to-image model you chose to use and why.

   I believe in baselines and do not want to reinvent the wheel. There are good quality Colab notebooks like:

   - https://colab.research.google.com/github/ShivamShrirao/diffusers/blob/main/examples/dreambooth/DreamBooth_Stable_Diffusion.ipynb
   - https://colab.research.google.com/github/TheLastBen/fast-stable-diffusion/blob/main/fast-DreamBooth.ipynb

   I prefer the first one due to the code quality. This notebook supports the training of multiple concepts out of the box, as well as LoRA and Flax (Google JAX). Unfortunately, it does not support the newest Stable Diffusion XL (SDXL v1.0) yet, but it should be easy to adapt DreamBoothDataset from original diffusers for the multi-concept setting by an analogy:

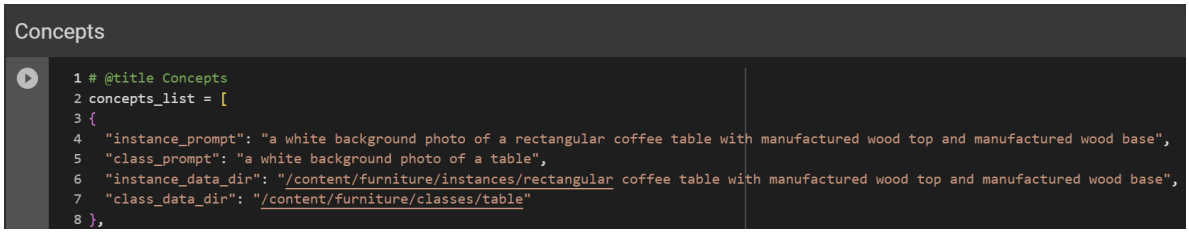
   diffusers/train_dreambooth.py -> ShivamShrirao/train_dreambooth.py
   diffusers/train_dreambooth_lora_sdxl.py -> ?
   https://github.com/huggingface/diffusers/tree/main/examples/dreambooth
   https://github.com/ShivamShrirao/diffusers/tree/main/examples/dreambooth

2. Please outline the process by which you took to fine-tune the model.

I adapted the ShivamShrirao notebook to SD v2.1 because, originally, it was written for 1.5. Also, I performed a comprehensive data filtering:

- Filtered images that do not have the same color around the corners, which means that object is not fully in the focus of the camera (cropped)
- Filtered images without sufficient white background (usage of domain knowledge: most of the furniture is on a white background)
- Filtered texture images
- Manually filtered images that carry too less information and rather confuse the model (e.g., top views)

In the end, the images were restructured by instance (text description) and by class (sofa, table, …) to use in the notebook:



3. List out the hyperparameters values you used for fine-tuning and how you determined each parameter. We want to see your scientific thinking around experimental design.

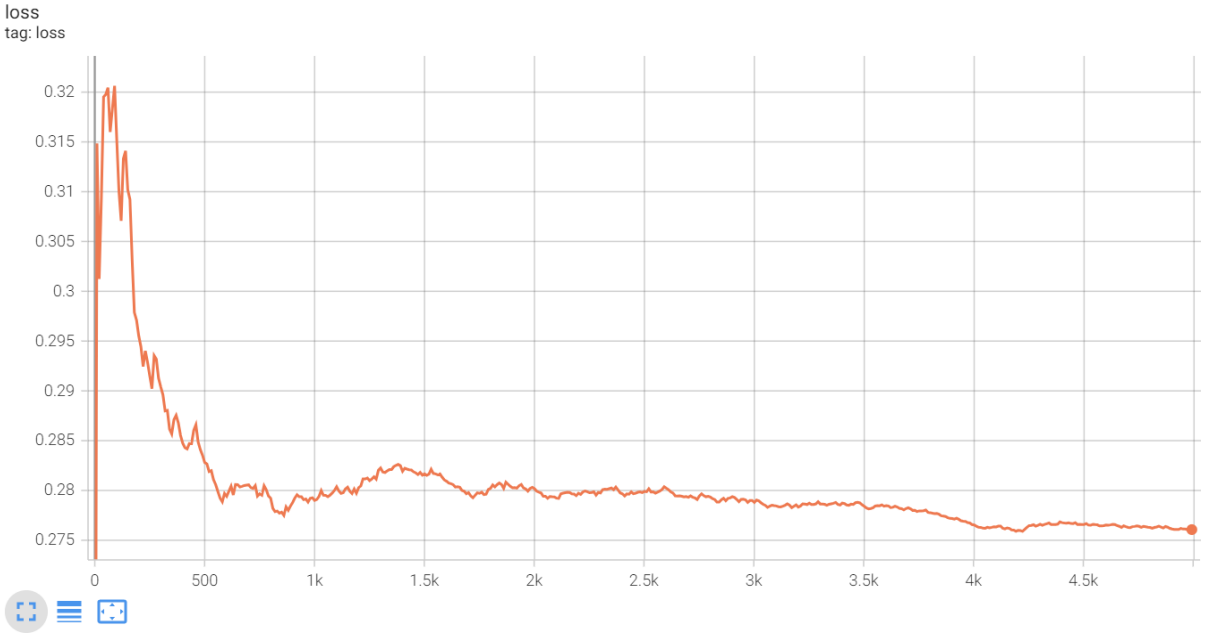
Stable Diffusion models are very sensitive to hyperparameters. There is a common set of hyperparameters that empirically results in stable training.

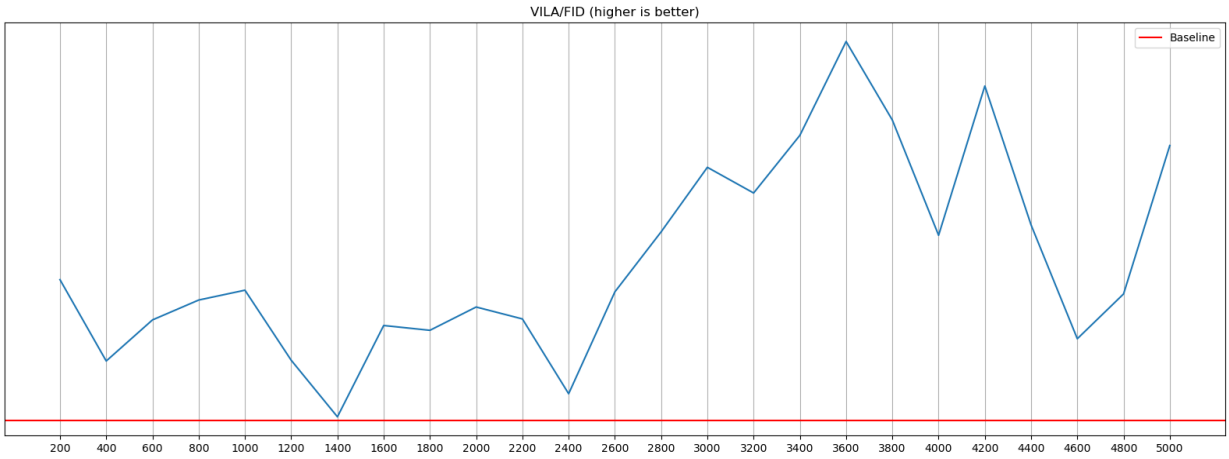
```
--revision="fp16" \
--with_prior_preservation --prior_loss_weight=1.0 \
--seed=1337 \
--resolution=512 \
--train_batch_size=1 \
--train_text_encoder \
--mixed_precision="fp16" \
--gradient_accumulation_steps=1 \
--learning_rate=1e-6 \
--lr_scheduler="constant" \
--lr_warmup_steps=0 \
--num_class_images=50 \
```

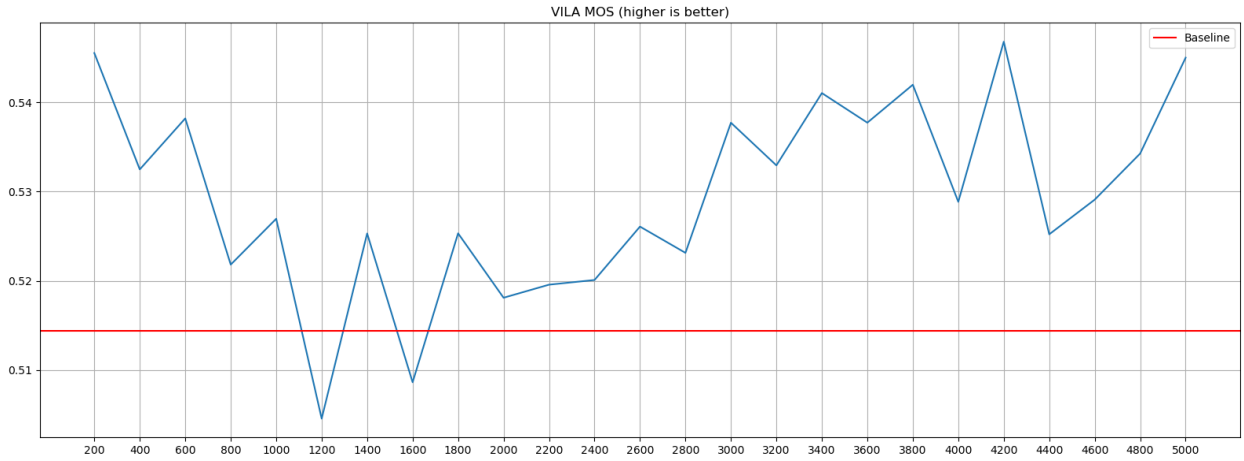
I consider the guidance scale of 7.5-8 and 25-50 steps as a golden mean for SD. These values are often used in papers, for example, SDXL. The number of steps is due to the fact that most of the models reach saturation with it. The ShivamShrirao's notebook also uses DDIMScheduler instead of more advanced DPM++ because the authors of DreamBooth think DDIMScheduler results are better.

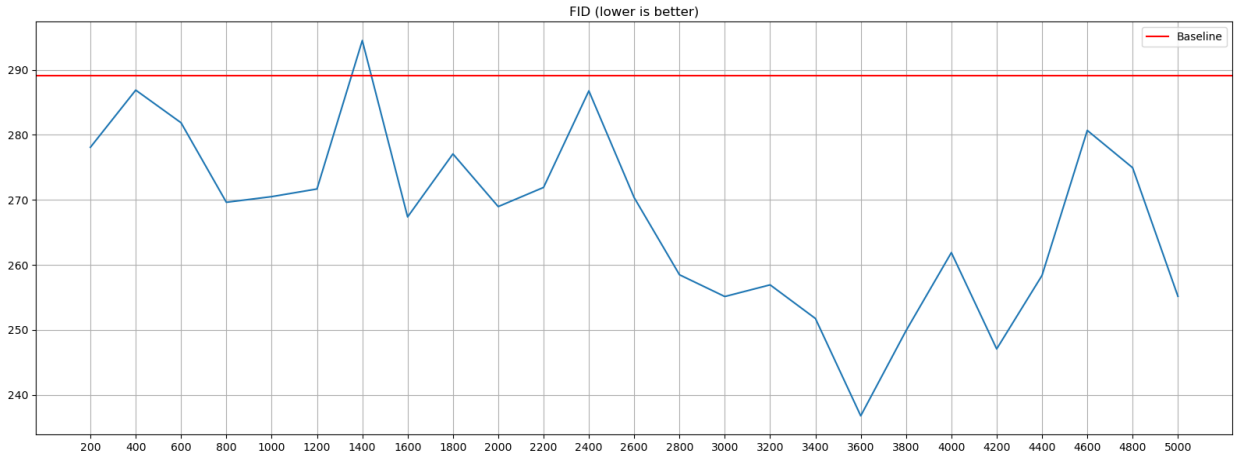
4. Evaluate the results of your model output qualitatively. How does it compare to output from other models (explicit statement of strengths vs. weakness)? If helpful, include outputs from the other models (including baseline pre-trained models).

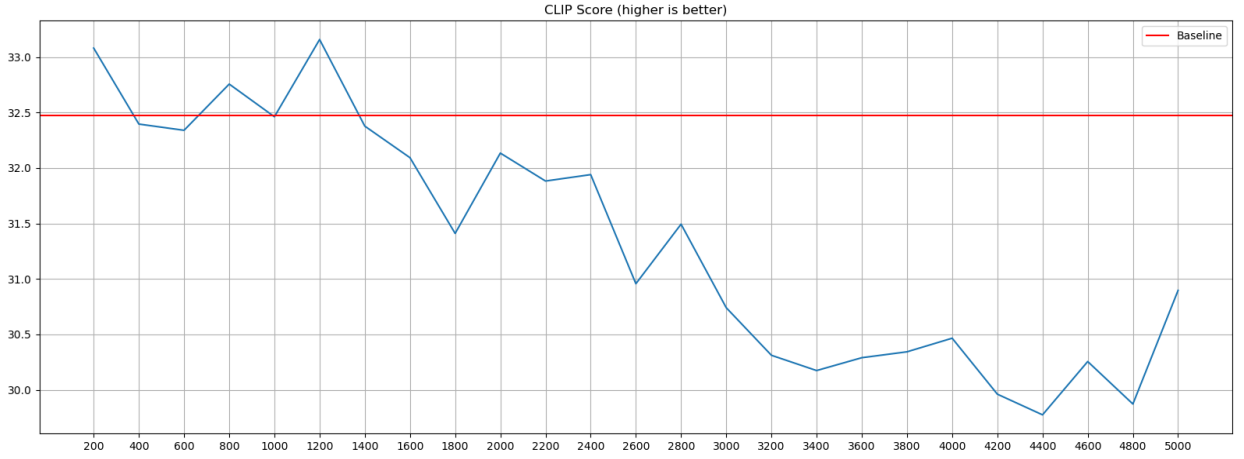
As I already mentioned in the “Evaluation” section, we will use CLIP score, FID, VILA. Our results are strictly higher than the baseline in the main metric VILA÷FID. This metric follows our intuition that the model has to produce aesthetic results (higher VILA option score), and has to produce in the style we want (lower FID, – lower distance). Low CLIP score may be a result of using the same CLIP as SD v2.1 (needs reevaluation).






VILA MOS (higher is better)
Baseline
0.54
0.53
0.52
0.51
200 400 600 800 1000 1200 1400 1600 1800 2000 2200 2400 2600 2800 3000 3200 3400 3600 3800 4000 4200 4400 4600 4800 5000


FID (lower is better)
Baseline
290
280
270
260
250
240
200 400 600 800 1000 1200 1400 1600 1800 2000 2200 2400 2600 2800 3000 3200 3400 3600 3800 4000 4200 4400 4600 4800 5000


CLIP Score (higher is better)
Baseline
33.0
32.5
32.0
31.5
31.0
30.5
30.0
200 400 600 800 1000 1200 1400 1600 1800 2000 2200 2400 2600 2800 3000 3200 3400 3600 3800 4000 4200 4400 4600 4800 5000

Original v2.1 model vs. 3x Fine-tuned (step #3600, no cherry-picking):

1. A living room with two leather armchairs.

2. A living room with an armchair with round and flared arms.

3. A living room with a concrete wall, and a round coffee table with marble top and metal.

4. A living room with a square coffee table with solid + manufactured wood top and iron base.

5. An armchair with arms in a living room.

6. A 100% polyester loveseat with round arms in a living room.

7. A living room with a round coffee table with solid wood top and solid wood base.

8. A living room containing a velvet sectional with square arms.

9. A living room with a genuine leather loveseat with square arms.

10. A living room with a red brick wall, and a 100% polyester sectional with square arms

5. What benchmarks / metrics would you use to compare output quality in a realistic experimental setup?

In addition to mentioned metrics, we can use Mean Opinion Score from crowdsourcing.

6. If you were to continue refining this model, what would your next steps be?

LoRA reparametrization, quantization, and pruning, there is also a brand new paper called HyperDreamBooth that leverages hypernetworks to “fine-tune” models in a few- or zero-shot manners.

Additional features may include: background removal, imperfections cleanup, relight, enhance/upscale, uncrop.

7. Outline the computational requirements and what it would take to put this model in production for scaled use by many users.

24GB VRAM would be enough, something like RTX 4090. I used Colab PRO and 40 GB A100 because 16 GB V100 does not fit the model without quantization and LoRA.

- 512 resolution — 11 GB for training, 19 GB when saving checkpoint
- 1024 resolution — 17 GB for training, 19 GB when saving checkpoint

Without optimizations (8 bit Adam, …), the peak memory usage is 32.5 GB.

To put this in production, we first need to optimize the model as listed in the previous (#6) answer so that it does not consume a lot of cloud computing, then pack it into a container, for example, using github:replicate/cog. Then we need some RabbitMQ/Redis priority queues with gRPC/GraphQL.

1 month for a proof of concept
2 month for a minimum viable product
6 month for a production

05:33 p.m. – Done.
2.85 hours total + one night without sleep for fine-tuning and evaluation.

https://huggingface.co/Arkan0ID/furniture-model-sd-2-1-200
https://huggingface.co/Arkan0ID/furniture-model-sd-2-1-3600
https://huggingface.co/Arkan0ID/furniture-model-sd-2-1-4200 (best)
https://huggingface.co/Arkan0ID/furniture-model-sd-2-1-5000

https://www.kaggle.com/datasets/dmitryvinnik/furniture
https://www.kaggle.com/datasets/dmitryvinnik/furniture-full

Part 2

Questions:

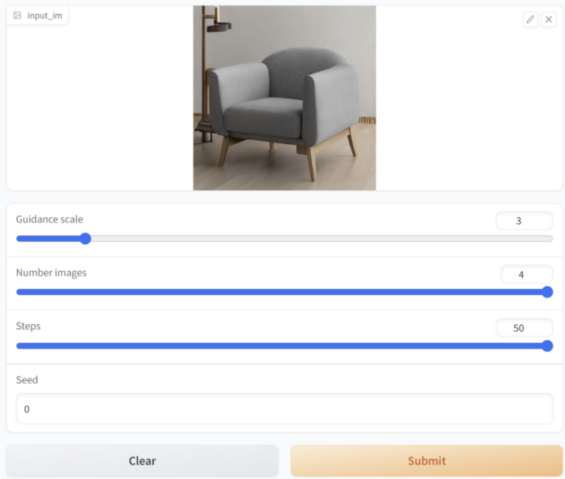
1. Please outline the technique you used or developed. Include the research you did to discover the technique used, and any other techniques you considered. For techniques you did not use, please briefly explain why you made that initial choice.

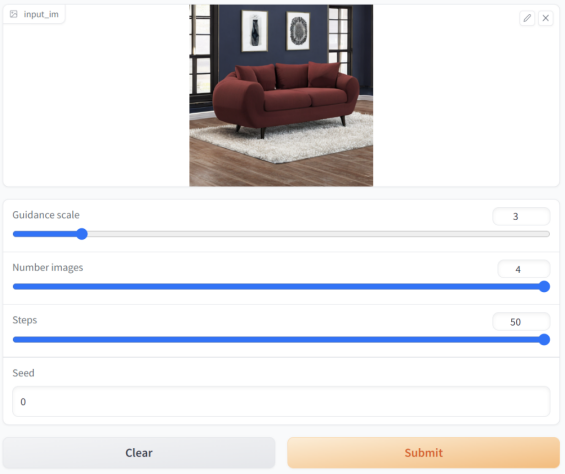
There are several ways to handle this:

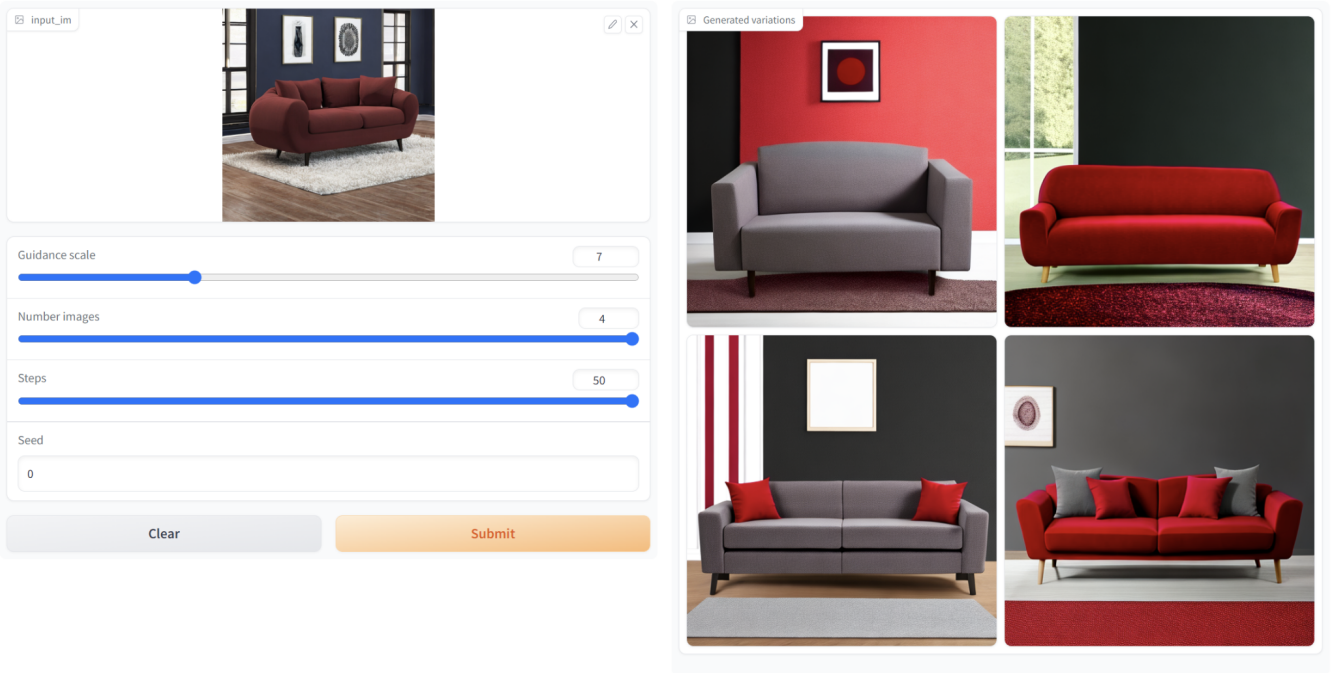
- lambdalabs/stable-diffusion-image-variations with small guidance from the original image lets SD be more creative:

  https://huggingface.co/spaces/lambdalabs/stable-diffusion-image-variations

  Variations from a generated image (constant seed, same guidance):

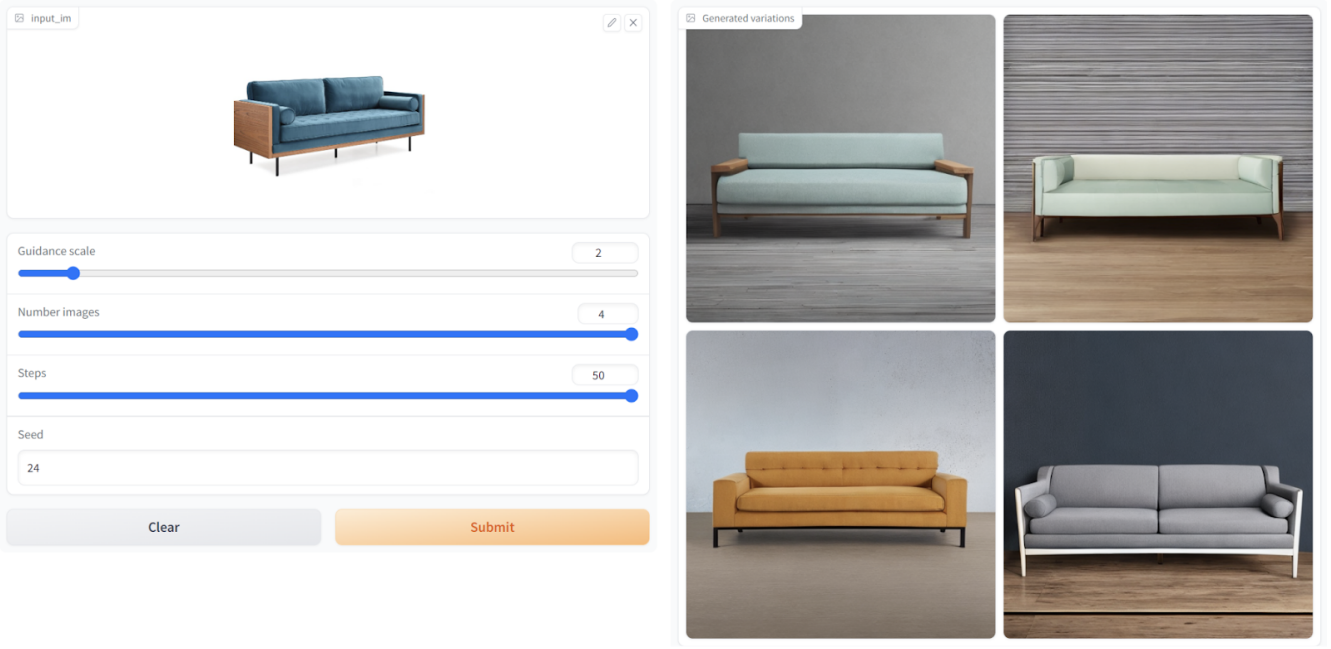






– A guidance increase does not preserve the environment.

Variations from a dataset image (a perfect image/seed cherry-picking):



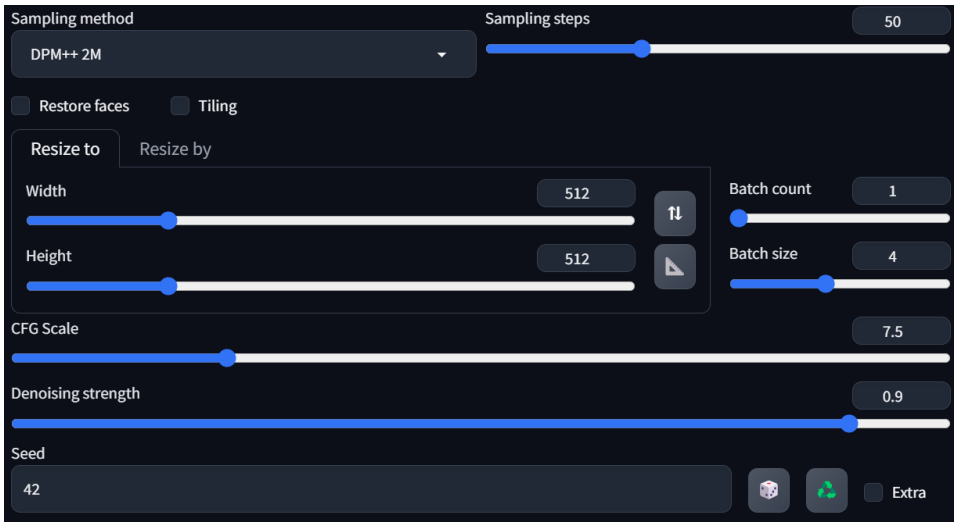
– As we can see, this method does not preserve the environment as was stated in the task. Therefore it does not suit us.

- stabilityai/stable-diffusion-2-1-unclip
  https://clipdrop.co/stable-diffusion-reimagine

  Variations from a generated image:

  Variations from a dataset image (a perfect image):

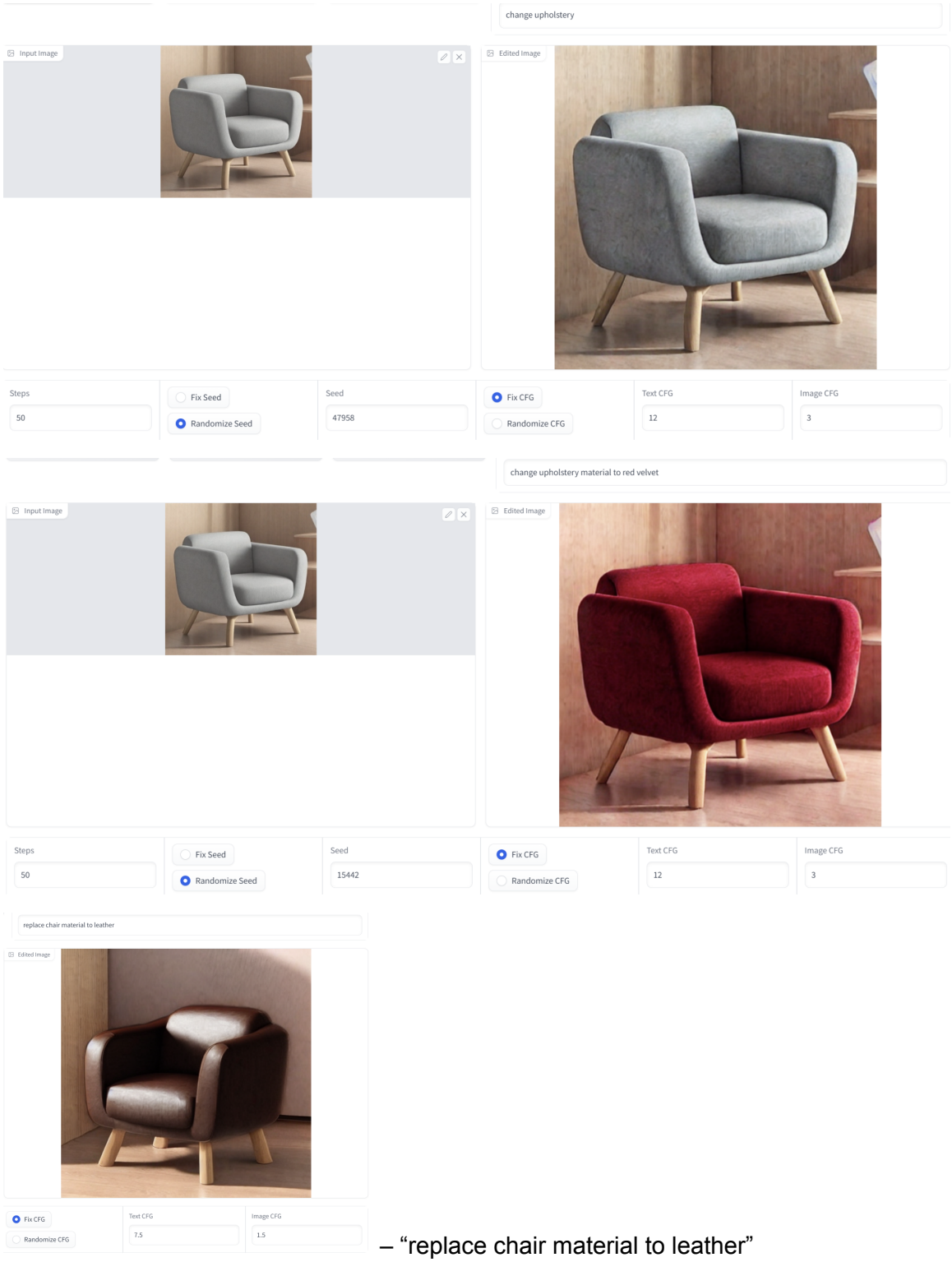
  I tried sd21-unclip-h.ckpt in automatic1111/stable-diffusion-webui to reproduce the ClipDrop results. The general rule is to use 0.7+ Denoising strength because at lower values the noise is noticeable. My parameters:



This model changes the form, but textures mostly remain the same.

- InstructPix2Pix, Learning to Follow Image Editing Instructions: https://huggingface.co/spaces/timbrooks/instruct-pix2pix
  Variations from a generated image (“change the upholstery” prompt, text 7.5, img 1.5):

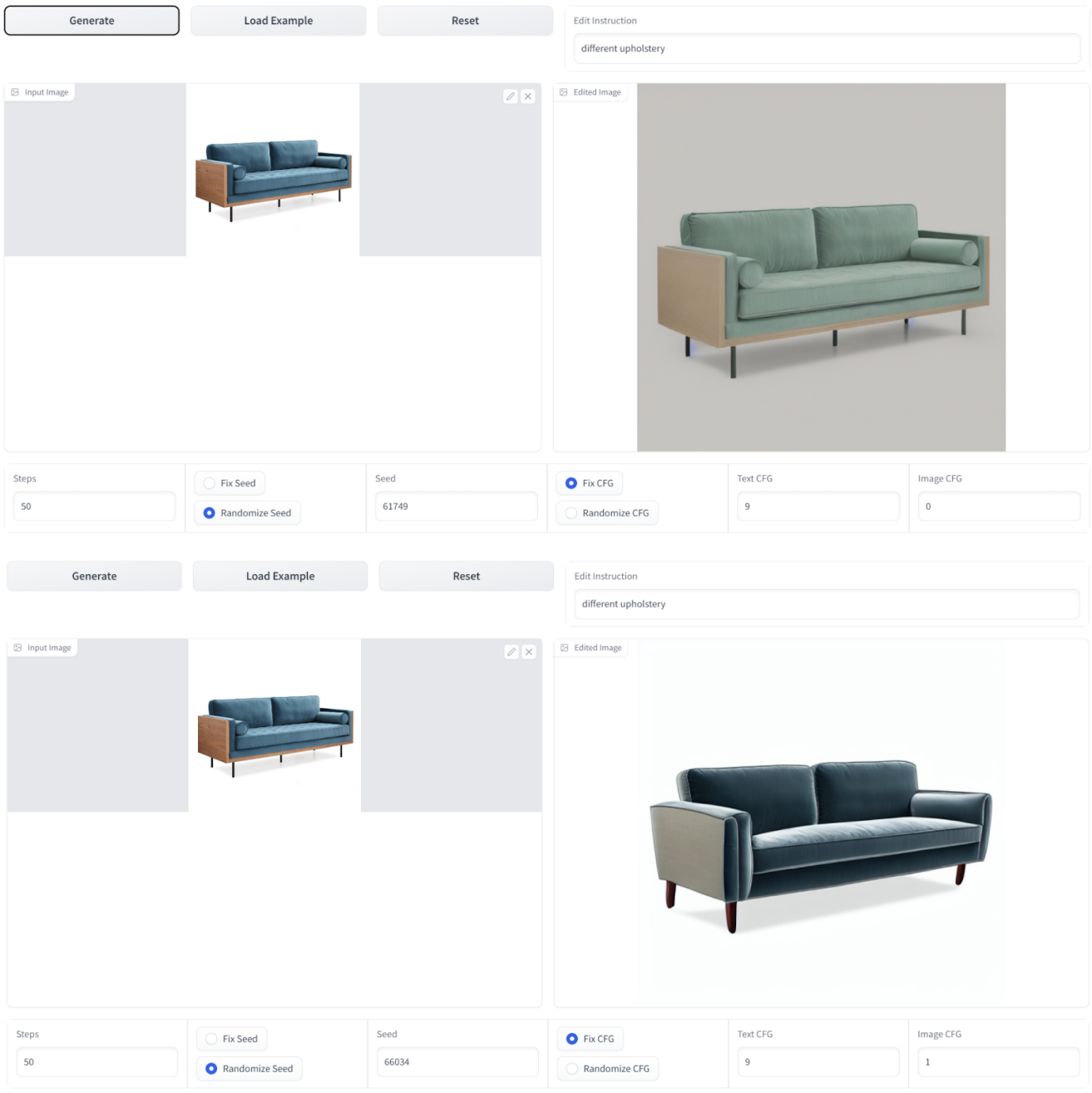
The sofa config does not work for chairs. We need image guidance to be 3 to avoid junk results. Also, the model is not creative (lazy) and needs specific instructions. It does not change the material with the first prompt:



– “replace chair material to leather”

Previously mentioned issues can be fixed with several configs + a ranking model.

Variations from a data set image (a perfect image):



<u>This model is the most suitable for us but requires domain knowledge</u> (an instruction prompt). If we have a limited scope of furniture, it is not a problem to make a set of prompts to modify different aspects.

2. Please outline the process you used for training and evaluation of the chosen technique. Did you need to train a new model, use a base model, or the model you had trained in part 1?

> In the first part, we fine-tuned the model to align it with our specific understanding of: What it should generate, How we imagine faux leather and how genuine leather, What perspective we want to have on the living room. When we use image-to-image models to generate variation, most of this semantics is preserved. In addition, the Lambda variation model has no open-source training code, instruct-pix2pix is not fine-tunable from a small subset because it does not contain a prior preservation mechanism. Therefore they are not intended to fine-tune and I did not fine-tune them.

3. How did you encourage diversity in the variations while maintaining consistency to the original image? Is there an automated measure that you could build here to quantify level of variation vs. the original image?

> Text/image guidance scale parameters for lambdalabs and InstructPix2Pix, Denoising strength for unCLIP. FID is a good automatic measure as it uses the distance between embeddings which are dense carriers of semantics.

4. What do you wish you would have done differently in part 1 based on your experience working in part 2?

> I enlarged the number of steps from 25 to 50 because 25 is not always enough.