Probabilistic Programming - Project 3

- Diffusion Ensemble, 2022-2023 -

Ionescu Andrei, 407

andreei.ionescu@gmail.com

Abstract

This paper serves as a research initiative and has the objective to explore ways in which images produced by different text-to-image diffusion algorithms can be combined in order to produce new results that contain elements from both, using a strength factor which adjusts the weight between them.

1 Related Works

In the domain of image synthesis tasks, recent works such as DALL-E (Ramesh et al., 2021) and Stable Diffusion (Rombach, Blattmann, Lorenz, Esser, & Ommer, 2021) have shown very impressive results in comparison to existing GAN networks. Moreover, works such as these follow a multi-modal approach and guide the training process with encodings of different types of data (text, images, etc.). However, the computational requirements become quite high in order to learn representations across a vast domain and so trying to retrain the parameters is not always feasible. As a consequence, some of the latest papers such as (Hu et al., 2021), (Ruiz et al., 2022) and (Gal et al., 2022) try to handle this aspect either through finetuning pretrained models, optimizing and finding embeddings that guide the model or reducing the training process.

2 Introduction

The objective that is tackled in the following sections is to be able to **synthesize** new images by combining the resulting outputs from multiple **text-to-image** based **diffusion models**. To do this it's not enough to apply operations at pixel space level because the images can contain objects, textures, and other patterns that have great semantic value, not only structure. Moreover, the task is a bit *ill-posed* since the model should make some assumption when selecting features from each image, as no other guiding input is given by a user.

A starting point is to make use one of the precursors of the Stable Diffusion, which is VQGAN (Esser, Rombach, & Ommer, 2020), a vector-quantized variational autoencoder that is trained to learn a **discrete codebook** representation and could infer new images using a transformer. According to the cited paper, the model is first trained to reconstruct the given images by applying various losses: *mse reconstruction loss*, *adversarial loss* and a loss that minimizes the distance between the latent code representations given by the encoder and the *codebook representation*. So, instead of operating directly in the pixel space, I can leverage the latent discrete representation that is offered by the VQGAN, and aggregate the features at a more abstract level before reconstructing them. In turn, one of the key motivation points behind this work is to avoid fully training models because of computational requirements.

3 Preliminary Image Synthesis Through Diffusion

The first step is to build a framework that can take a varying number of diffusion models[1] and to use them as the *image data source* component in the ensemble. I considered that a user would give initial input through natural language form by either:

- Sending a single prompt to all *diffusers* - which could be useful to combine outputs that should represent the same content, yet learnt in different prior distributions.

[1] The models that are used for experimentation are: StableDiffusionV1.5 (Rombach, Blattmann, Lorenz, Esser, & Ommer, 2022), *GhibliDiffusion* (n.d.) and *GhibliDiffusion* (n.d.).

- Sending a list of prompts to all *diffusers* - where each prompt follows the previous case, and could be used to compute multiple different examples.
- Sending a *2d matrix* of prompts - where each row indicates a different case and the columns represent prompts that will be associated to the respective diffuser.

As mentioned before, the computational costs are quite high and it's very **memory costly** to produce the images in parallel across all different models. To ease this burden, a *sequential* approach is taken where:

1. The parameters for a diffusion algorithm are loaded on the GPU[2].
2. One or more images are processed in parallel using small batch sizes.
3. The images are collected and sent back to the CPU.
4. The current diffusion model is unloaded and the process repeats with the next one.

Note: There are other ways of obtaining input data could be taken such as: generating a dataset offline or using already existing images online, but the main point is to offer as much *flexibility* in experimentation, during both development and for the end user.

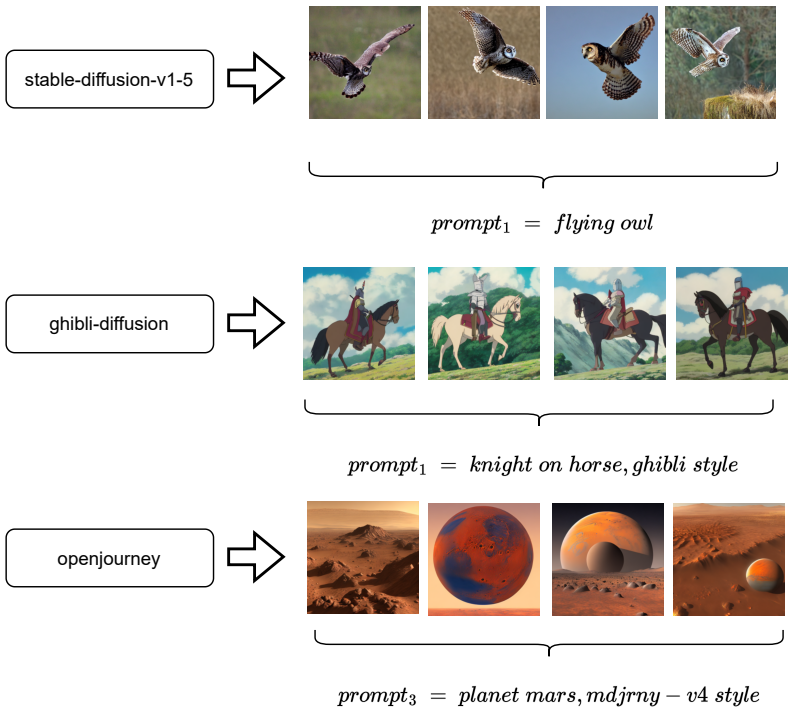

Figure 1: Pipeline for Generating Input Images

The outputs are then collected for further processing into a single tensor of shape $N_{prompts} \times M_{models} \times H \times W \times C$, where C represents the three color channels and $H = W = 512$, which are later resized to 256×256.

4 Image Synthesis Through Ensembling

As mentioned in the Introduction 2, a pretrained VQGAN model on the ImageNet (Deng et al., 2009) is used as the foundation for the following experiments as it is able to reconstruct many images, assuming that they share at least some common characteristics with the image distribution of the ImageNet dataset.

$$\overline{x_{diff}} = G(z_{codebook}) = G(quantize(E(x_{diff}))) \tag{1}$$

[2]The GPU used in the experiments is a RTX 4090 that has 24 GB VRAM.

4.1 Composing Latent Vectors with Linear Combinations

One approach of mixing the input images is to encode all of them near the near the *bottleneck* and extract the latent representations given by the VQGAN, which are then combined linearly with an array of constants, $w = \{w_i | w_i \in [0, 1], i \in \overline{1, M}\}$, given by the user. After this step, the latent representation can be transformed back to its initial pixel space through the decoder component as highlighted in the Figure below.

$$\overline{x_{diff}} = G(\sum_{m=1}^{M} w_i \cdot quantize(E(x_{diff_i}))) \tag{2}$$

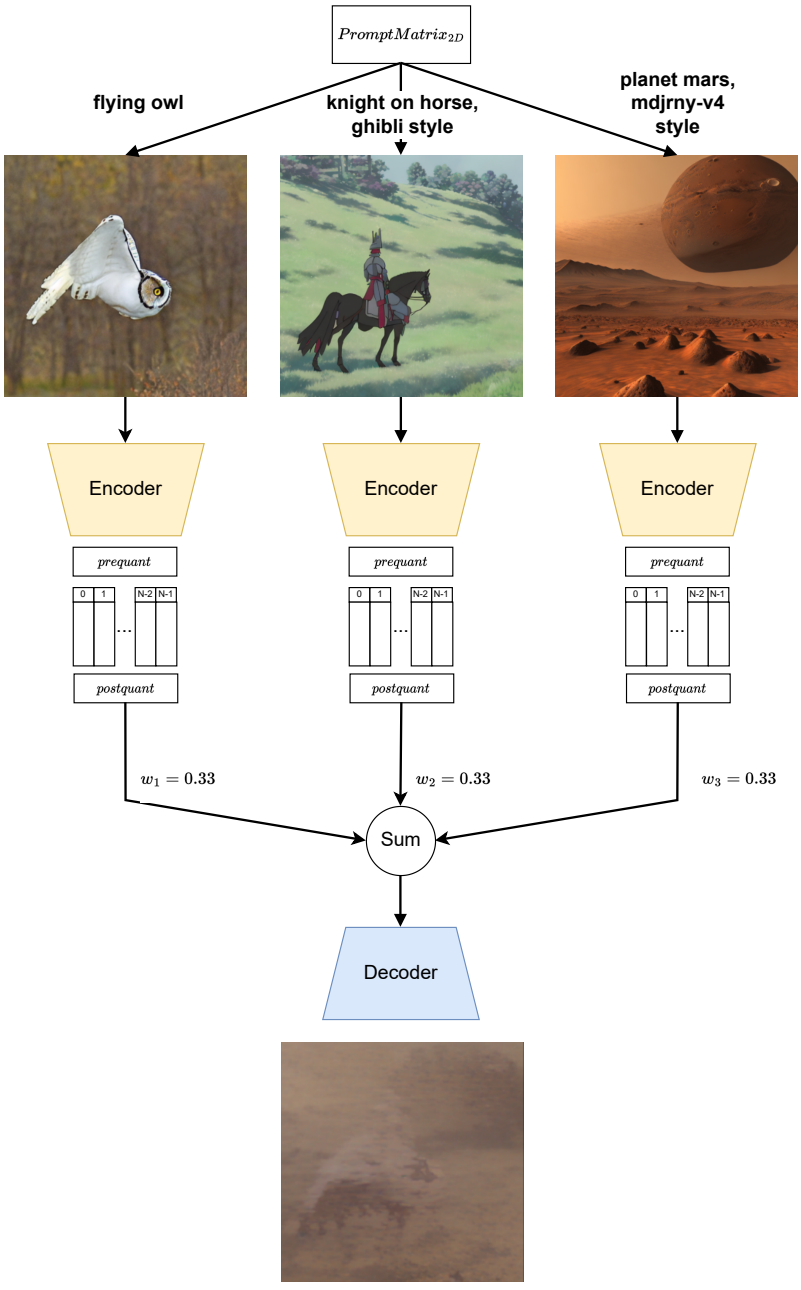

Figure 2: The Diffusion Ensemble Pipeline

The images are encoded to a latent representation that is either: *preconv*, *prequant* or *postquant*. This means that the feature vectors can be extracted before or after they are swapped with the ones present in the codebook, though they mostly offer similar results. Despite applying a linear operation in latent space, the result is mostly just a *blend* of the input images in the shown case.

This task can also be restricted to use only two diffusion algorithms as input data sources, and the composing operation can be interpreted as a **baricentric equation**. The results shown below are obtained using Ghibli and OpenJourney Diffusion models:

$$\overline{x_{diff}} = G((1-\gamma) \cdot quantize(E(x_{diff_1})) + \gamma \cdot quantize(E(x_{diff_2}))) \tag{3}$$

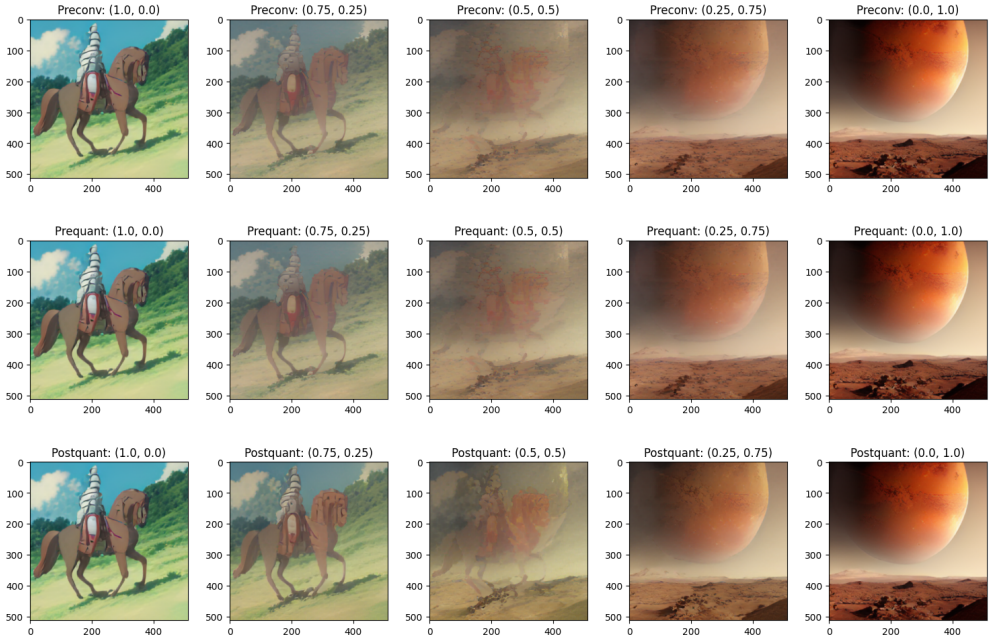

Figure 3: Baricentric Composition of Two Images Containing Different Objects
Each row indicates the layer from where the activations vectors were extracted and each column shows the baricentric factors used in the equation.

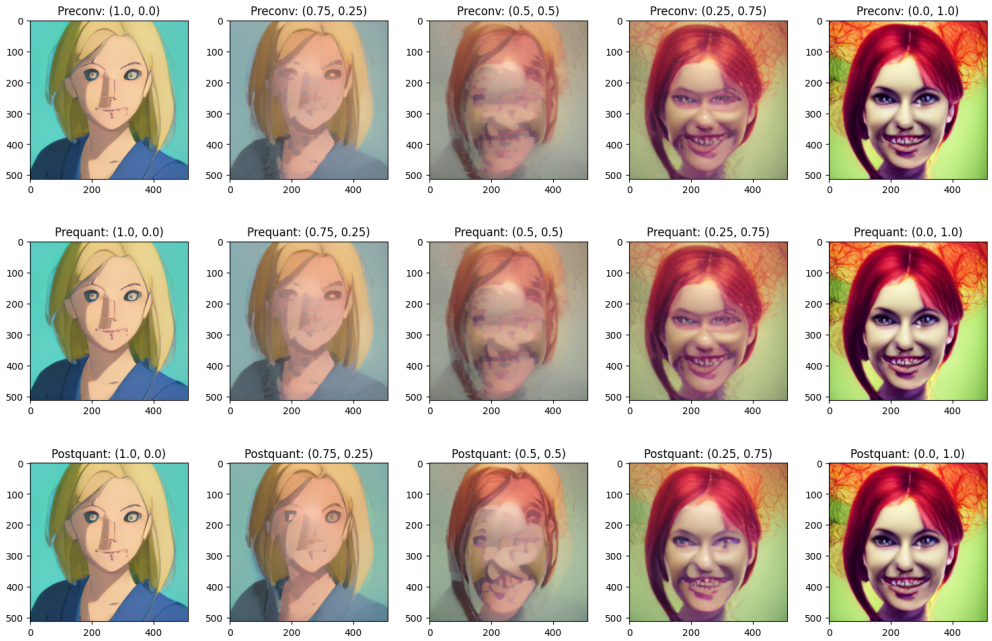

Figure 4: Baricentric Composition of Two Images Containing Human Faces
The faces have some artifacts due to the reconstruction error, and the composition does not do anything more than blending.

Figure 5: Baricentric Composition of Two Images Containing Landscape and People
Similarly to the other examples, the images are just mixed together with many details lost.

Observations: The results of this approach indicate that the way the features of the two images are combined is a bit too *naive*. Composing the *latent representations* using either *linear* or *baricentric* operations results in a simple mixtures of feature vectors, which are then swapped with the most similar vectors in the codebook representation. Because the model was trained to minimize these distances, it will choose vectors that are similar to the given combinations, which might be why I am getting a blending as if I just applied it in pixel space.

4.2 Latent Candidate Selection Through Minimization

Another perspective on the text-to-image multimodal paradigm is shown by **VQGAN+CLIP** (Crowson et al., 2022), where they leverage CLIP's (Radford et al., 2021) contrastive learning and similarity objective to minimize the distances between the crops of the generated image and a textual representation given by the user. This is done by differentiating in an iterative fashion, propagating the gradients of the loss function with respect to the latent candidate that is either given initially by the user or randomly generated.

Based on this, I adopt a similar extension for this task by replacing the textual input representation with the current image input representation. Candidate selection is done by integrating two losses that are used to update the latent activations:

- **Perceptual Loss** - reusing the VGG16 discriminator found in VQGAN, it can be repurposed to add enforce structure in the final image, by computing an L_2 *Loss* over the intermediary last-5 layers activations for each of the the $2 * M$ pairs (each diffusion model generates an image that is compared with the combined one).
- **CLIP Loss** - the CLIP model has very good zero-shot performance and its activations, obtained after applying the image encoder, can be used such that the generated image contains concepts from all input images generated by the diffusers;

The *perceptual loss* is weighted **significantly lower** in order to avoid the previous blending problem, otherwise the structures of all given images would be enforced and the results would be *poor semantic-wise*. On the other hand, solely applying the *CLIP loss* is not good enough as the model has to make ambiguous decisions which result in a mixture of patterns. Moreover, the *seed* for the candidate can be obtained as discussed in the previous subsection or by sampling a random latent vector however, through experimentation they converge to a similar image.

Also, an enhancement is introduced to compute the loss crop-wise as suggested by Crowson et al. (2022), which reduces noise that can be propagated by CLIP. The the optimization process is done over

many epochs with interesting results starting from around epoch 600, and Adam (Kingma & Ba, 2014) is used with a 0.35 learning rate that decays slowly by a factor of 0.9 every 100 epochs.

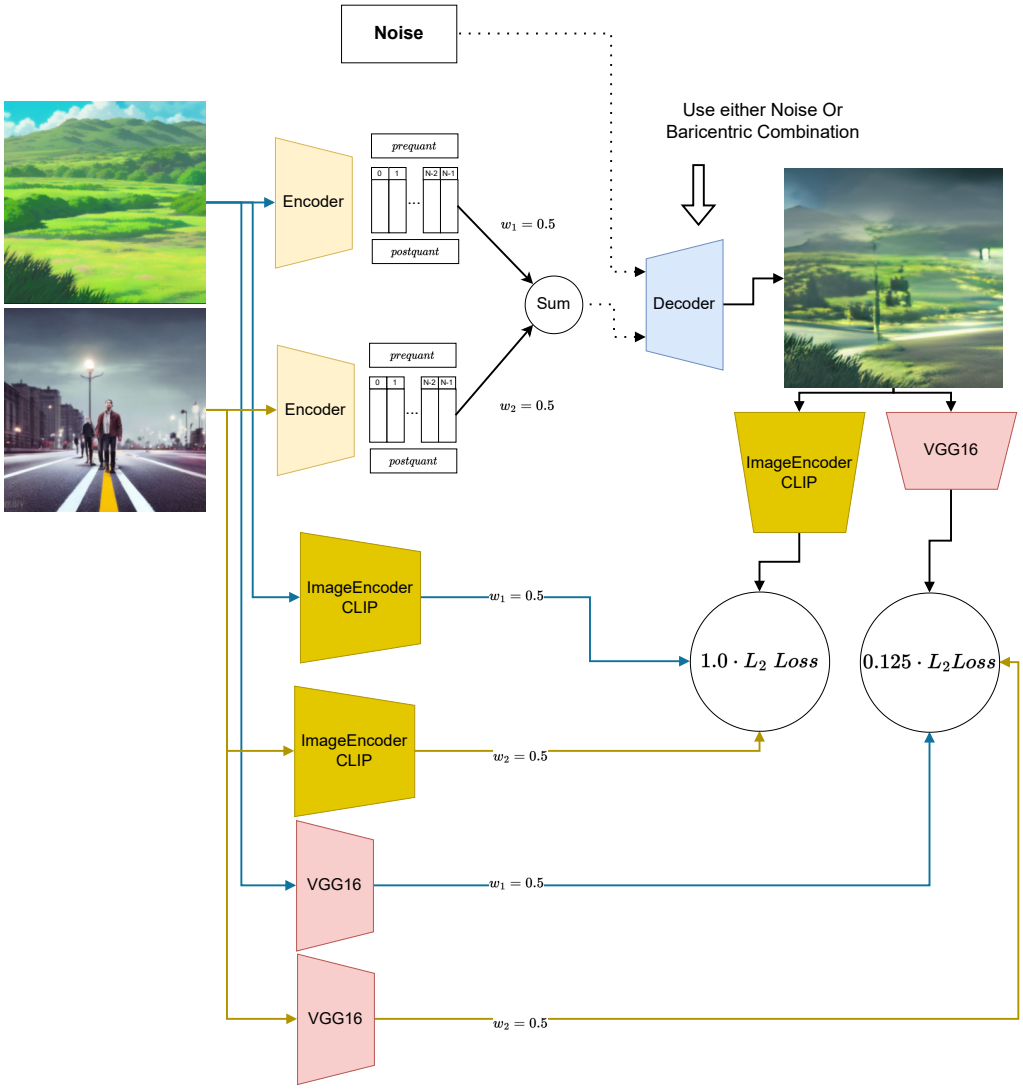

Figure 6: Introducing CLIP and Perceptual Losses Into the Diffusion Ensemble Pipeline

5 Results

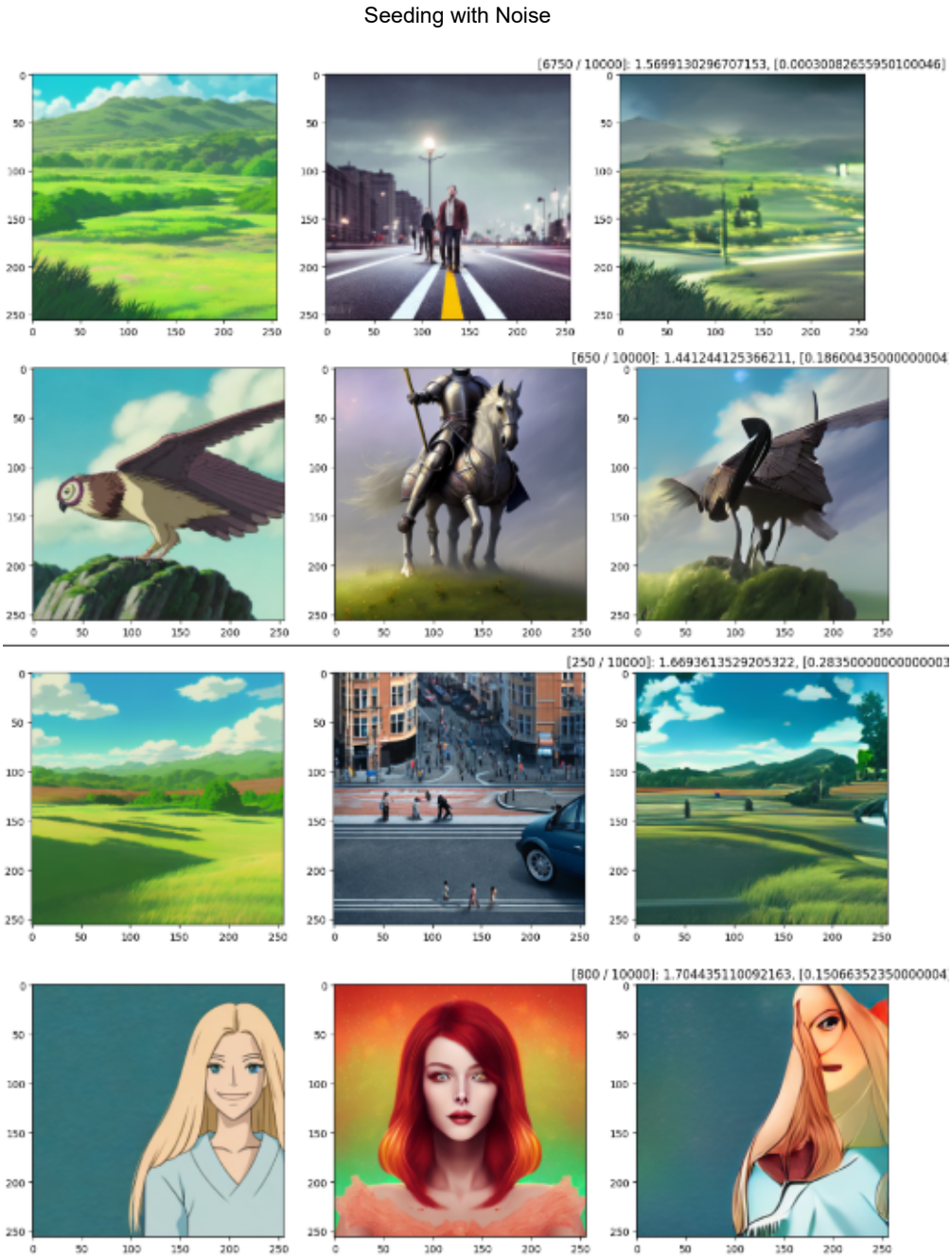

Figure 7: Synthesized Images with CLIP, Perceptual Loss and Noise Seed

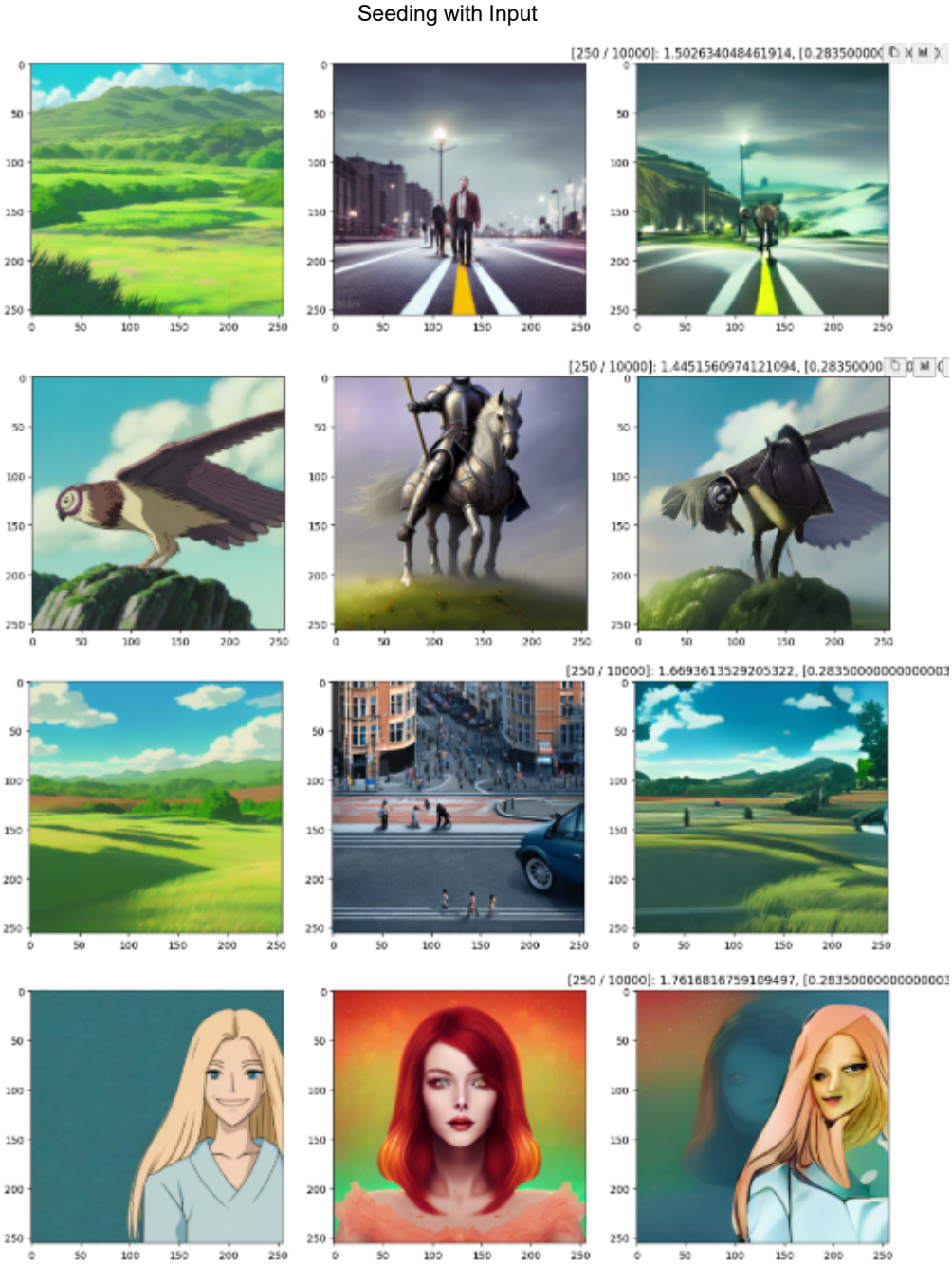

Figure 8: Synthesized Images with CLIP, Perceptual Loss and Input Seed

6 Conclusion

While the results are not quite satisfactory as they do not look plausible, it was nonetheless a great learning experience on which further knowledge can be built on. As a future idea, the problem should be rephrased to better address a clearer objective and minimize the ambiguities that a model may face.

References

Crowson, K., Biderman, S., Kornis, D., Stander, D., Hallahan, E., Castricato, L., & Raff, E. (2022). *Vqgan-clip: Open domain image generation and editing with natural language guidance.* arXiv. Retrieved from https://arxiv.org/abs/2204.08583 DOI: 10.48550/ARXIV.2204.08583

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 ieee conference on computer vision and pattern recognition* (pp. 248–255).

Esser, P., Rombach, R., & Ommer, B. (2020). *Taming transformers for high-resolution image synthesis.* arXiv. Retrieved from https://arxiv.org/abs/2012.09841 DOI: 10.48550/ARXIV.2012.09841

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., & Cohen-Or, D. (2022). *An image is worth one word: Personalizing text-to-image generation using textual inversion.* arXiv. Retrieved from https://arxiv.org/abs/2208.01618 DOI: 10.48550/ARXIV.2208.01618

Ghiblidiffusion. (n.d.). Retrieved February 8, 2023, from https://huggingface.co/nitrosocke/Ghibli-Diffusion

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ... Chen, W. (2021). *Lora: Low-rank adaptation of large language models.* arXiv. Retrieved from https://arxiv.org/abs/2106.09685 DOI: 10.48550/ARXIV.2106.09685

Kingma, D. P., & Ba, J. (2014). *Adam: A method for stochastic optimization.* arXiv. Retrieved from https://arxiv.org/abs/1412.6980 DOI: 10.48550/ARXIV.1412.6980

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... Sutskever, I. (2021). *Learning transferable visual models from natural language supervision.* arXiv. Retrieved from https://arxiv.org/abs/2103.00020 DOI: 10.48550/ARXIV.2103.00020

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., ... Sutskever, I. (2021). *Zero-shot text-to-image generation.* arXiv. Retrieved from https://arxiv.org/abs/2102.12092 DOI: 10.48550/ARXIV.2102.12092

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2021). *High-resolution image synthesis with latent diffusion models.* arXiv. Retrieved from https://arxiv.org/abs/2112.10752 DOI: 10.48550/ARXIV.2112.10752

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022, June). High-resolution image synthesis with latent diffusion models. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition (cvpr)* (p. 10684-10695).

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., & Aberman, K. (2022). *Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation.* arXiv. Retrieved from https://arxiv.org/abs/2208.12242 DOI: 10.48550/ARXIV.2208.12242