# An improvement on "Guiding Text-to-Image Diffusion Model Towards Grounded Generation"

Walter Simoncini
walter@ashita.nl

Ankit
ankit.uva.nl@gmail.com

University of Amsterdam
Amsterdam, The Netherlands

https://github.com/WalterSimoncini/grounded-diffusers

## Abstract

*This paper improves an existing approach to ground the generations of a text-to-image diffusion model which simultaneously generates an image and a segmentation mask for the target objects described in the text prompt. Our contributions are the following: (i) we created a flexible pipeline that allows grounding several diffusion models in a plug-and-play fashion, (ii) we investigated the impact of prompt engineering on the grounding abilities of the model, (iii) we improved the training procedure of the grounding model by using a log-cosh loss, inducing regularization using dropout and training for multiple epochs, as well as decoupling the dataset generation from training. Finally, we evaluate the performance of open-vocabulary object grounding against a previous approach and show that we improve the mIoU score by 4% while using a quarter of the data.*

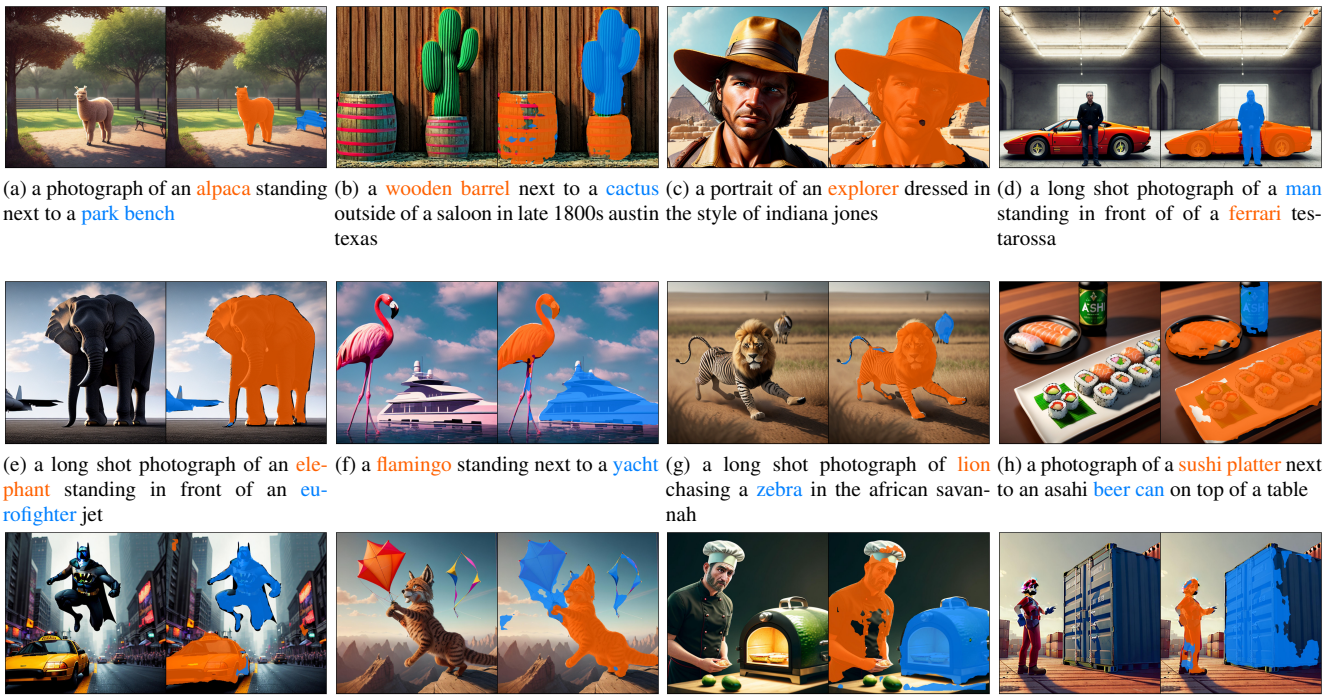
## 1. Introduction

Semantic segmentation is a fundamental computer vision task whose goal is classifying whether an image pixel belongs to a semantic category or the "background". Many approaches have successfully tackled this with supervised methods [25, 31]. While classification datasets (e.g., ImageNet-1k [7]) cover thousands of classes, semantic segmentation datasets are more restricted, covering tens [8, 23] to hundreds of semantic classes [41, 42]. This limitation is due to the cost of producing such datasets, as human annotators have to provide pixel-level annotations for semantic entities [20]. To tackle this problem, several zero-shot [3, 13, 20], and few-shot [2, 33] methods have been proposed, but they still rely on annotated datasets for training or on a pre-trained model. Parallel to these developments, text-to-image generative models have received increasing attention. These models can generate photorealistic images with robust vision-language correspondence given a text prompt [30]. Previous work [22] has shown that it is possible to generate images and ground semantic entities simultaneously [22], paving the way toward a controlled generation of semantic segmentation datasets. This is beneficial in low-data scenarios or tasks with the goal of segmenting long-tail objects, as significantly larger datasets are needed to find sufficient training examples for rare categories [12, 14]. Even though the grounding model can accurately ground an object in an image, its performance quickly deteriorates when multiple objects are present. This paper tackles this limitation by making the following contributions: (i) We make the grounding pipeline flexible by adapting the HuggingFace diffusers [38] library for visual grounding, allowing an easy comparison of several diffusion models. This paper focuses on Stable Diffusion version 1.5, its fine-tuned checkpoint, and version 2. (ii) We investigate the impact of prompt engineering for training the grounding model and propose a straightforward and more effective prompt: "x and y" for a two-class image. (iii) We improve the training process by using the log-cosh loss, dropout, and decoupling the dataset generation from training, which allows training the grounding model for multiple epochs. Our grounding model can segment multiple objects in a single image even when long-tail labels are used, such as "sushi platter" and "eurofighter" as shown in Figure 1. We follow the first evaluation protocol proposed by [22] to quantitatively assess our model. We show that our contributions not only achieve similar results in grounding objects of seen classes but also improve the open-vocabulary grounding abilities of the model while using a quarter of the data.

## 2. Related Work

### 2.1. Image Generation

Image generation is challenging due to the high dimensionality and variance of image data. Several methods have been proposed for this task, such as Generative Adversar-



(a) a photograph of an alpaca standing next to a park bench

(b) a wooden barrel next to a cactus outside of a saloon in late 1800s austin texas

(c) a portrait of an explorer dressed in the style of indiana jones

(d) a long shot photograph of a man standing in front of of a ferrari testarossa

(e) a long shot photograph of an elephant standing in front of an eurofighter jet

(f) a flamingo standing next to a yacht

(g) a long shot photograph of lion chasing a zebra in the african savannah

(h) a photograph of a sushi platter next to an asahi beer can on top of a table

(i) a long shot photograph of batman jumping over a taxi

(j) a photograph of a lynxes flying a kite

(k) a long shot photograph of a chef looking at an avocado shaped oven

(l) a photograph of super mario standing in front of a shipping container

Figure 1. **Qualitative results.** The original image and the segmentation masks generated by our grounding model for one or two classes. The model is able to ground long tail objects such as a **sushi platter**, **eurofighter**, **batman** and **super mario**

ial Networks (GAN) [10], Variational Autoencoders (VAE) [19] and autoregressive models [4, 37]. Out of these, GANs have the highest fidelity, but they have a non-trivial training process and may suffer from catastrophic forgetting, resulting in mode collapse and non-convergence [36]. Recently, diffusion models have become popular due to their ability to synthesize high-quality and diverse images [11, 15, 34]. These models can be conditioned on text, enabling text-guided image generation. Examples are DALL-E 2 [29] and Stable Diffusion [30], which exploit the vision-language correspondence learned by CLIP [27], and Imagen [32], which can synthesize high-quality image despite using T5 [28], a language model trained exclusively on text.

### 2.2. Visual Grounding

Visual grounding can be defined as the task of understanding a natural language query and finding the target object(s) in an image [22]. Previous works operate either in a two-stage or single-stage approach. Two-stage methods, such as the object detector ViLD [12], propose a set of image regions to be queried and classify those with an open-vocabulary classification model. OpenSeg [9] approaches semantic segmentation by first predicting mask proposals, refined in a second stage via region-word alignment. Single-stage approaches have been shown to achieve better performances. One such method is LSeg [20], where the authors train an image encoder to output pixel-aligned CLIP embeddings using a labeled segmentation dataset. Other single-stage methods, such as CRIS [40] and CLIPSeg [26], train a decoder to output pixel-aligned relevancy maps given a CLIP language embedding and intermediate outputs of its visual encoder. The latter also supports multimodal queries, as the query can be either text or an image. Our method, heavily based on [22], is more similar to these approaches, but instead of image features from CLIP, we use feature maps from the Stable Diffusion U-Net. Also, we do not require any labeled data for training. More recent approaches, such as LERF [17], tackle grounding in a 3D space by learning a language field inside a neural radiance field.

## 3. Methodology

First, we start with a brief overview of diffusion models and Image Quality Assessment in Sections 3.1 and 3.2. In Section 3.3, we provide a detailed description of the procedure used to generate our dataset, followed by the grounding model architecture in Section 3.4. Next, we discuss the prompt engineering process in Section 3.5 and finally we delve into the training process in Section 3.6.

### 3.1. Diffusion Models

Diffusion Models are probabilistic generative models that learn a data distribution by iteratively denoising a Gaussian sample. They can be decomposed into two processes: the forward process, which iteratively adds noise to a real data sample, and the backward process, which, through a time-conditioned neural network, learns to progressively denoise a sample. In this paper, we employ Stable Diffusion version 1.5 [30], a text-to-image diffusion model, which conditions the reverse process on a textual input often referred to as "prompt." Stable Diffusion has three main components: a pre-trained variational autoencoder (VAE), used to encode images in its latent space; a text encoder, CLIP [27], which encodes variable-length prompts into a fixed size text embedding and a time conditional U-Net [31], which predicts the noise to be removed at each time step from the sampled latent vector. The U-Net also enforces the visual-language interaction using cross-attention layers. During inference, we sample a vector $z^T \sim \mathcal{N}(0, I)$ and apply $T$ denoising steps using the U-Net. The resulting vector $z^0$ is then mapped back to the image space using the VAE.

### 3.2. Image Quality Assessment

The quality of images generated by Stable Diffusion may vary (see Figure 12, 13), thus the original paper [22] handpicks testing samples to provide a more realistic evaluation of the grounding model. We automate this procedure by picking the top 80% of the testing set using a no-reference metric, CLIP-IQA+ [39]. This metric can quantitatively assess the quality of a generated sample without needing a reference image. This is done by calculating the cosine similarity of the image embedding from CLIP with the text embeddings of antonym prompts (e.g., "good photo" and "bad photo") and computing a softmax over the two scores. While simple, this strategy effectively assesses the image quality and its abstract perception or "feel."

### 3.3. Dataset Construction

Following the approach of [22], we build a dataset for training the segmentation model. Each sample can be expressed as a tuple of (image, U-Net feature maps, segmentation masks, object classes). The original paper generates the dataset during training, whereas we decouple the dataset generation from it, separating them into two different procedures. This makes experimenting with the learning process computationally cheaper as we no longer need to generate the data on every run. The dataset was generated using Stable Diffusion version 1.5 [30] as implemented in the HuggingFace diffusers [38] library. Images are generated using the prompt "a photograph of a x and a y", where x and y are two randomly selected classes from the PASCAL VOC [8] dataset. The dataset consists of 20 classes, split into 15 seen classes (to generate the training set) and 5 unseen classes, using the Split-1 of [22] as shown in Table 1. We obtain the ground truth segmentation mask for each generated example using the pre-trained off-the-shelf segmentation model Mask R-CNN [24]. This choice motivated us to use classes from PASCAL VOC since they are a subset of COCO [23], the dataset used to train Mask R-CNN. The generated samples for which Mask R-CNN cannot find as many masks as the classes specified in the prompt are discarded.

| Split | Classes |
|---|---|
| Seen | aeroplane, bicycle, bird, boat, bottle, bus, cat, chair, cow, diningtable, sheep horse, motorbike, person, pottedplant |
| Unseen | tvmonitor, car, dog, sofa, train |

Table 1. **Split-1 for Pascal VOC.** The seen classes are used to train the grounding model, and the unseen ones to evaluate its open-vocabulary grounding abilities

For testing, we generate three subsets: one using only seen categories, one whose images contain both a seen and an unseen object and one whose objects are both from unseen categories following [22]. The last setup directly evaluates the open-vocabulary object grounding abilities of our model.

### 3.4. Grounding Model Architecture

We use the grounding model architecture proposed in [22], which consists of a diffusion model, a visual encoder, a text encoder, and a fusion model, as shown in Figure 2. Before training, We pre-compute the features maps generated by the U-Net, significantly reducing training time. The visual encoder takes the U-Net features as input and produces visual tokens for the image. The text embeddings for target classes x and y are computed from the prompt "x and y" using the CLIP [27] text encoder, which is also used in Stable Diffusion. The text encoder is kept frozen during training. Finally, the fusion model takes the visual tokens and the text embeddings from the visual and text encoders and outputs a mask embedding for each object in the image. The fusion model consists of a transformer decoder, with the text embedding as the `Query` and the visual tokens as the `Key` and `Value`, followed by a Multi-Layer Perceptron (MLP) consisting of 3 layers. Finally, the mask embeddings are combined with the visual tokens through a dot product operator to generate class-specific segmentation masks.

### 3.5. Prompt Engineering

The original architecture [22] uses the CLIP [27] embeddings of the `<|endoftext|>` token of the prompt "a photograph of x" as the language input for the segmentation model. We trained the segmentation model with sev-

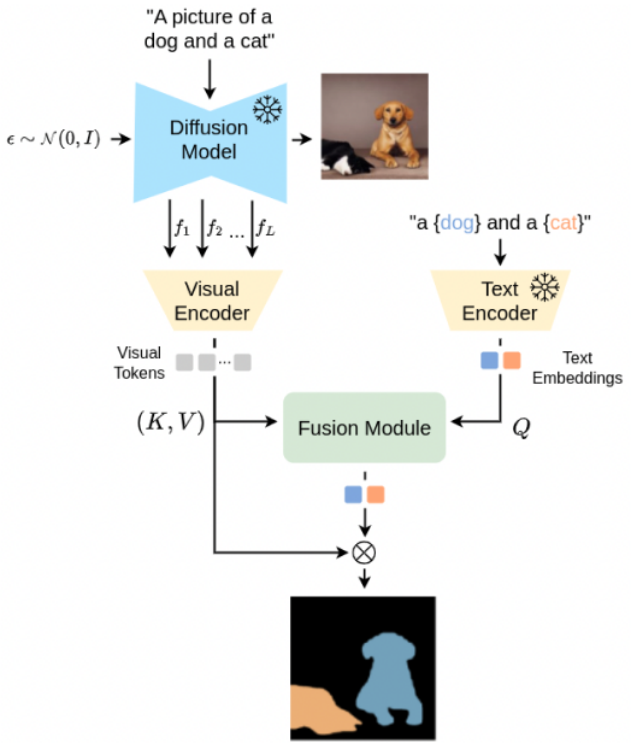



Figure 2. **Grounding model architecture:** The visual and text encoder respectively encode the visual tokens from stable diffusion using features from its U-Net and the text embeddings from the prompt using CLIP. The visual tokens and the text embeddings are given as input to the fusion module, composed by a transformer decoder followed by an MLP that outputs mask embeddings. These are finally combined via a dot product with visual tokens to obtain class-specific masks. During training, both the diffusion model and the text encoder are kept frozen.

eral prompts for one epoch without dropout and, as seen in Table 3, we found that for images with two objects, the prompt "x and y" works best, granting a 5% improvement on the unseen test set. In addition, we implemented an algorithm that selects all the embeddings corresponding to an object's class given its class name and averages them. This is necessary because CLIP's tokenizer uses byte-level Byte-Pair-Encoding, and thus a word may map to multiple tokens. The process is illustrated in Figure 3, using the "pottedplant" class from Pascal VOC as an example.

### 3.6. Training

We train the grounding model in a supervised fashion using the generated dataset. For optimization, we use a hybrid loss function composed of two terms: a binary cross entropy loss and a logarithm of the hyperbolic cosine function applied to the dice loss. Using $m_i$ and $m_i^{\text{gt}}$ to respectively indicate the predicted and ground truth masks, we define the loss as follows:

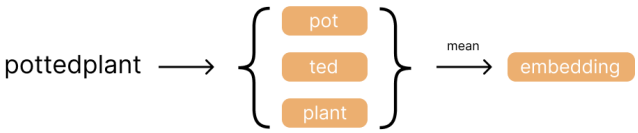



Figure 3. **Text embedding computation**: the algorithm finds all tokens that correspond to the class name and averages them, resulting in a single class embedding.

$$L_{\text{BCE}}(m_i^{\text{gt}}, m_i) = -\frac{m_i^{\text{gt}} \log(m_i) + (1 - m_i^{\text{gt}}) \log(1 - m_i)}{N}$$

$$L_{\text{DICE}}(m_i^{\text{gt}}, m_i) = 1 - \frac{2 m_i^{\text{gt}} m_i + 1}{m_i^{\text{gt}} + m_i + 1}$$

$$L(m_i^{\text{gt}}, m_i) = \log\Big(\cosh\big(L_{\text{DICE}}\big)\Big) + L_{\text{BCE}}$$

Our loss function differs from the one in [22], where Binary Cross Entropy was used to train the grounding model. We use dice loss because of its increased performance in segmentation tasks, as demonstrated in [16], which compared several loss functions for the task of semantic segmentation. The results showed that the logarithm of the hyperbolic cosine of the dice loss had the best performance. The performance of several loss functions for this task are shown in Table 4.

## 4. Experiments and Results

### 4.1. Datasets

We generated training and validation datasets containing 10k and 1k images using the training classes from Split-1 of Pascal VOC from [22] ensuring that each image contained two objects from different classes. For testing, we generated three datasets, each composed of 1k images: one containing only seen classes, one whose images contained both a seen and an unseen class, and one using only unseen classes.

### 4.2. Evaluation Metrics

To evaluate the grounding model performance we use the category-wise mean Intersection-over-Union (mIoU) following [22], which is defined as:

$$\text{mIoU} = \frac{1}{C} \sum_{c=1}^{C} \text{IoU}_c$$

where $C$ is the total number of target categories, and $\text{IoU}_c$ is the Intersection-over-Union for the category with index $c$.

| | Dataset Size | | Test Set mIoU | | |
|---|---|---|---|---|---|
| | Training Images | Test Images | Seen | Seen+Unseen | Unseen |
| Baseline [22] | 10k | 1k | 73.07 | 57.08 | 46.59 |
| Baseline [22] | 40k | 1k | 78.93 | 66.07 | 57.93 |
| Improved model | 10k | 1k | 77.20 | **67.50** | **58.43** |
| Improved model (with scoring) | 10k | 800 | 78.05 | **68.08** | **61.77** |

Table 2. **Quantitative results from the evaluation of grounded generation.** Our model has been trained on a synthetic training set, which consists of images of two objects from only seen categories, and tested on a synthetic test set, which consists of images with two objects of both seen and unseen categories. Our model (**with dropout = 0.1**), as observed in rows 3 and 4, outperforms the model from the paper [22], which we consider as the baseline. The "with scoring" in row 4 refers to scoring the test images using the **CLIP-IQA+** [39] metric, ranking them and selecting the top 800 ones for evaluation.

### 4.3. Implementation Details

We use the pre-trained Stable Diffusion v1.5 from RunwayML [30] as implemented in the HuggingFace diffusers library [38] as the diffusion model and CLIP [27] as our text encoder. We follow the approach in [22] and use a Mask R-CNN [24] model trained on COCO [23] to generate the ground truth segmentation masks. We train our grounding model on an NVIDIA TITAN RTX GPU for 10 epochs, using a batch size of 1, while [22] trains the model for a single epoch with a batch size of 8. We use the Adam optimizer [18] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and a constant learning rate of $1e-4$.

### 4.4. Quantitative Results

#### 4.4.1 Grounded Generation

Table 2 shows the results from the evaluation of our grounding model. We compare our results to [22] (trained on 10k and 40k samples), which we use as a baseline for our experiments on 10k samples. Our model significantly outperforms the baseline on the unseen and seen+unseen setups, which suggests it can generalize well and clearly highlights its open-vocabulary abilities. Its performance on seen categories is on par with the baseline. This is because we induced better regularization by using a dropout of 0.1 in our grounding model. In [22] the test images are cherry-picked, ensuring the generation quality and the accuracy of the segmentation masks predicted by the off-the-shelf segmentation model. To mimic this we evaluate our model on the top 80% of the test set as scored with CLIP-IQA+ [39] in the "with scoring" row, in addition to an evaluation on the full test set.

#### 4.4.2 Prompt Engineering

We compared the effectiveness of different text prompts to ground images, as shown in Table 3. Our model performs best when the input prompt is "x and y" where x and y are the class names of the depicted objects. We assume this behavior is due to CLIP representing text as a bag of words, and a more extensive prompt would push the embedding semantics further away from the grounding targets.

| Prompt | Training Seen | Test Unseen |
|---|---|---|
| $x$ and $y$ | **61.34** | **39.28** |
| a photograph of $x$ and $y$ | 60.39 | 37.95 |
| a photograph of $x$ | 60.34 | 34.16 |
| $x$ | 61.14 | 33.54 |

Table 3. **Quantitative results for comparison of different prompts.** Segmentation accuracy (mIoU) comparison on the training and test unseen subsets of pascal-sim for different prompts used to train the segmentation model. The model was trained for a single epoch without dropout.

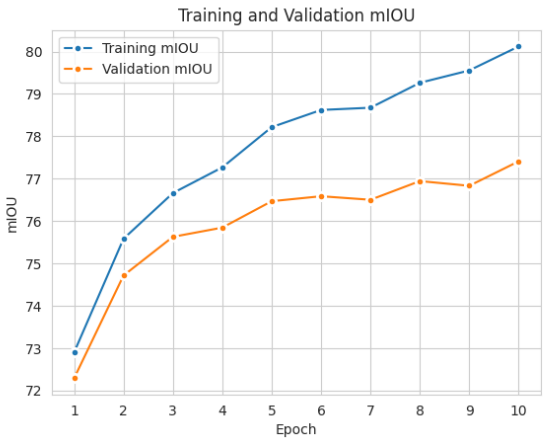



Figure 4. **Semantic learning curve of our model measured using the mIoU.** The model was trained on 10k and validated on 1k synthetic images generated using Stable Diffusion v1.5.

#### 4.4.3 Epoch-Based Training

Figure 4 shows that our grounding model performance improves when training with multiple epochs. We train our model for 10 epochs in contrast to the approach in [22], where the authors train the grounding model for a single epoch. We use the best checkpoint on the validation set (obtained at epoch 10). The iterative learning process grants more opportunities for our model to capture more complex information regarding the shape of the objects, which ultimately helps in predicting better masks.

Figure 5. **Comparison between Mask R-CNN and the grounding model:** each column shows the image generated by Stable Diffusion, the masks identified by Mask R-CNN, and the masks from the grounding model.

### 4.5. Visualization

We illustrate our results in Figure 5. Our grounding model is able to successfully ground objects in the segmentation masks, even beyond the capabilities of the off-the-shelf detector used for the ground truth. This demonstrates the open-vocabulary abilities of our model.

### 4.6. Ablation Studies

We conduct several ablation studies using a smaller dataset of 1k images containing two objects from the training classes of Pascal VOC Split-1 [22]. We use a validation set of 200 images with seen classes and three test sets, one for each of the seen, seen+unseen, and unseen setups, each containing 200 samples.

#### 4.6.1 Loss Function Comparison

We compared the effectiveness of four loss functions: Binary Cross Entropy (BCE), as used by [22], which we consider the baseline, Dice [35], a combination of Dice and BCE and our loss function, which combines BCE and the logarithm of the hyperbolic cosine function (log-cosh) of the dice loss as shown in Table 4. We trained the model for one epoch without dropout and found that all the loss functions improved over the baseline for the seen and unseen setups. However, only our loss function significantly outperformed BCE on the unseen setup, improving the mIoU by 3%. Moreover, it also achieves the best mIoU on the seen+unseen setup, with an improvement by 2% over BCE.

| Loss Function | Seen | Seen+Unseen | Unseen |
|---|---|---|---|
| BCE | 56.98 | 42.07 | 35.51 |
| Dice [35] | **59.16** | 43.75 | 35.44 |
| Dice [35] + BCE | 58.72 | 42.73 | 35.16 |
| Log-Cosh Dice [16] + BCE | 58.20 | **44.26** | **38.23** |

Table 4. **Ablation study (mIoU) for the loss function**. The model was trained for one epoch without dropout on 1k images and tested on 200 images.

#### 4.6.2 Stable Diffusion Versions Comparison

We assessed the impact of using different versions of Stable Diffusion to generate the training dataset as shown in Table 5. We also provide a qualitative comparison, shown in Figure 6. In particular, we compared the version 1.5 base model to a fine-tuned checkpoint [6] and to version 2.0. We followed the same experimental setup as other ablations experiments, generating a training set of 1k images, a validation set of 200 images, and three test sets, each composed of 200 images for every competing model.

We trained the segmentation models for 10 epochs, using a dropout of 0.2. The evaluation results in Table 5 show that version 2.0 performs worse on all three test setups, scoring between 1% and 3% less. We presume the issue is in the quality of generated images, which are qualitatively worse and do not correctly represent the two objects specified in the prompt. Moreover, generating a single sample from version 2.0 takes around twice the time of version 1.5 (7 seconds versus 3 seconds on an NVIDIA A100), and it more

| Model | Seen | Seen+Unseen | Unseen |
|---|---|---|---|
| Stable Diffusion 1.5 | 67.96 | 51.61 | 41.90 |
| Stable Diffusion 2 | 66.40 | 48.04 | 39.20 |

Table 5. **Ablation on Stable Diffusion versions.** Comparison of performance (mIoU) using images generated by Stable Diffusion v1.5 and Stable Diffusion v2. Both models were trained for 10 epochs using the log-cosh loss and a dropout of 0.2

Figure 6. **Qualitative model comparison**: images generated using Stable Diffusion version 1.5, its fine-tuned checkpoint and version 2.0 from top to bottom. The prompt classes were **person and cow**, **person and pottedplant**, **boat and sheep** from left to right.

than doubles the storage requirements, amounting to 120 MBs for a single sample versus 53 MBs. For the fine-tuned checkpoints we used a similar setup and, since they share the same U-Net architecture, a cross-comparison between the training datasets was possible by training the segmentation model on the dataset from either the base (standard) or fine-tuned (high quality) models and evaluating on both test sets. The results are shown in Table 6. As expected, both models perform slightly worse on the opposite dataset since the feature maps encoded by the U-Net may change as a result of additional training, but, surprisingly, the model trained using the fine-tuned dataset (high quality) improves on the unseen setup for the standard dataset. This model is also significantly better on the high-quality dataset, beating the model trained on the standard dataset by up to 9% on the unseen setup. This is most likely due to the higher quality of generated images. The CLIP-IQA+ [39] scores of the two datasets, shown in Table 7, also confirms the higher quality of the fine-tuned checkpoint samples. Some hand-picked high-quality generations can be seen in Figure 7.

Figure 7. **High-quality samples**. Hand-picked images generated using the fine-tuned checkpoint [6] of Stable Diffusion v1.5

#### 4.6.3 Impact of Dropout

We compared the impact of dropout on model training using the 10k dataset and its 1k test sets by training the model for a single epoch with various dropout values, as shown in Table 8. The results show that dropout is beneficial for training, and the best results were obtained by using a 0.1 dropout.

## 5. Limitations

Figure 8. **Overview of limitations.** From left to right: texture transfer from a sheep to a cow, wrong proportions between a sheep and a bottle, large mask for a missing object (umbrella) in an image and overlap of masks due to the bag of words behavior of CLIP

While the model shows impressive grounding abilities, it still has some limitations stemming from itself or the diffusion model as shown in Figure 8. First, generating images with multiple object categories is not trivial: the images may contain only a single object, may not respect proportions between objects (e.g., generating a bottle as tall as a cow), and may mix visual properties of the target semantic classes (e.g., generating a horse with the texture of a sheep). This is an inherent limitation of the chosen diffusion model, and other approaches [21] have been shown to partially tackle some of these issues. Regarding the segmentation model, there are three main limitations, the first being that CLIP embeddings behave as a bag of words, thus for a prompt such as "x and y" the masks might overlap, especially if the two objects are from close semantic categories

| | Testing Dataset | | | | | |
|---|---|---|---|---|---|---|
| | Standard | | | High-Quality | | |
| Training Dataset | Seen | Seen+Unseen | Unseen | Seen | Seen+Unseen | Unseen |
| Standard | 67.96 | 51.61 | 41.90 | 68.24 | 50.95 | 35.63 |
| High-Quality | 65.07 | 49.63 | 42.31 | **70.98** | **57.15** | **44.74** |

Table 6. **Ablation on standard and high-quality data.** Comparison of models trained on the Stable Diffusion v1.5 base model and a fine-tuned checkpoint that generates higher quality samples. All models were trained for 10 epochs with the log-cosh loss using a 0.2 dropout.

| Dataset | CLIP-IQA+ Score |
|---|---|
| Standard | 70.34 |
| High-Quality | 75.50 |

Table 7. **Ablation on standard and high-quality data using CLIP-IQA+.** CLIP-IQA+ [39] scores of the 1k training dataset generated by the base Stable Diffusion model v1.5 and of the one by its fine-tuned checkpoint.

| Dropout | Seen | Seen+Unseen | Unseen |
|---|---|---|---|
| 0 | 74.08 | 57.52 | 45.48 |
| 0.05 | 72.66 | 58.46 | 46.16 |
| 0.1 | 73.54 | 59.72 | 49.99 |
| 0.2 | 73.60 | 57.72 | 43.05 |

Table 8. **Ablation on effect of dropout on model performance** Impact of dropout regularization on models trained for one epoch using the 10k training dataset and the 1k test sets.

(e.g., horse and cow). This effect may be reduced using simple embeddings arithmetics, namely subtracting all the other objects embeddings from the input to the segmentation model, but further research is needed. The second limitation is that the model is not explicitly trained with negative examples: the training procedure assumes that the class represented by the embedding exists in the image, which results in the model outputting a mask that covers the whole image if the target object is not represented. Finally, as the model is trained using the U-Net feature maps, it cannot be directly applied to a real image, and thus a second zero-shot object detection method must be trained. This limitation may be overcome by applying one step of the forward process of diffusion to a real image and obtaining the features from the reverse process step.

## 6. Discussion and Conclusion

In this paper, we presented an approach to overcome the constraints of the grounding model proposed in [22] to segment multiple classes in the generated image. To overcome the limitations, we used Stable Diffusion version 1.5, a better prompt engineering strategy, and improved the training process via regularization and a hybrid loss that operates locally and globally to segment complex shapes. Extensive qualitative and quantitative experiments show that our method is superior to the baseline model proposed in [22], even when less data is available. Future works may investigate fine-tuning Stable Diffusion v1.5 with a segmentation dataset to generate images closer to the target distribution and preprocessing the object embeddings by subtracting the embeddings of all other represented objects, as this has been shown to produce sharper masks in our qualitative experiments. Moreover, our training procedure assumes that the two classes in the prompt are present in an image, causing the model to output a large mask covering most of the image for non-existing objects. Adding negative examples to the training procedure could counter this issue.

## 7. Difficulties and Contribution

The primary challenge of this project was the quality of the code from the original paper [22]: most scripts were incomplete and models for segmenting generated images with multiple objects were not provided thus we had to re-write the code from scratch. Moreover, the dataset generation was slow and cumbersome, requiring around 2.5 days to generate the 10k training dataset, which occupied around 500GBs of storage. The validation and testing datasets required an additional 1 day. The individual contributions of each group member are listed in Section **??** of the appendix.

## References


[1] Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. Deep vit features as dense visual descriptors. *arXiv preprint arXiv:2112.05814*, 2021. 11

[2] Malik Boudiaf, Hoel Kervadec, Ziko Imtiaz Masud, Pablo Piantanida, Ismail Ben Ayed, and Jose Dolz. Few-shot segmentation without meta-learning: A good transductive inference is all you need? In *Proceedings of the IEEE/CVF Con-*

*ference on Computer Vision and Pattern Recognition*, pages 13979–13988, 2021. 1

[3] Maxime Bucher, Tuan-Hung Vu, Matthieu Cord, and Patrick Pérez. Zero-shot semantic segmentation. *Advances in Neural Information Processing Systems*, 32, 2019. 1

[4] Xi Chen, Nikhil Mishra, Mostafa Rohaninejad, and Pieter Abbeel. Pixelsnail: An improved autoregressive generative model. In *International Conference on Machine Learning*, pages 864–872. PMLR, 2018. 2

[5] Ho Kei Cheng, Jihoon Chung, Yu-Wing Tai, and Chi-Keung Tang. CascadePSP: Toward class-agnostic and very high-resolution segmentation via global and local refinement. In *CVPR*, 2020. 11, 12

[6] Deliberate stable diffusion checkpoint. https://civitai.com/models/4823/deliberate. Accessed: 2023-05-27. 6, 7

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1

[8] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010. 1, 3

[9] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Scaling open-vocabulary image segmentation with image-level labels. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVI*, pages 540–557. Springer, 2022. 2

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 2

[11] 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*, pages 10696–10706, 2022. 2

[12] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation. *arXiv preprint arXiv:2104.13921*, 2021. 1, 2

[13] Zhangxuan Gu, Siyuan Zhou, Li Niu, Zihan Zhao, and Liqing Zhang. Context-aware feature generation for zero-shot semantic segmentation. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1921–1929, 2020. 1

[14] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019. 1

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 2

[16] Shruti Jadon. A survey of loss functions for semantic segmentation. In *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. IEEE, oct 2020. 4, 6

[17] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. *arXiv preprint arXiv:2303.09553*, 2023. 2, 11

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 5

[19] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[20] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and René Ranftl. Language-driven semantic segmentation. *arXiv preprint arXiv:2201.03546*, 2022. 1, 2

[21] Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. Gligen: Open-set grounded text-to-image generation. *arXiv preprint arXiv:2301.07093*, 2023. 7

[22] Ziyi Li, Qinye Zhou, Xiaoyun Zhang, Ya Zhang, Yanfeng Wang, and Weidi Xie. Guiding text-to-image diffusion model towards grounded generation, 2023. 1, 2, 3, 4, 5, 6, 8

[23] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 1, 3, 5

[24] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *CoRR*, abs/2103.14030, 2021. 3, 5, 12

[25] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1

[26] Timo Lüddecke and Alexander Ecker. Image segmentation using text and image prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7086–7096, June 2022. 2

[27] 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. *CoRR*, abs/2103.00020, 2021. 2, 3, 5

[28] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020. 2

[29] 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. 2

[30] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *CoRR*, abs/2112.10752, 2021. 1, 2, 3, 5

[31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted*

*Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015. 1, 3

[32] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022. 2

[33] Amirreza Shaban, Shray Bansal, Zhen Liu, Irfan Essa, and Byron Boots. One-shot learning for semantic segmentation. *arXiv preprint arXiv:1709.03410*, 2017. 1

[34] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 2

[35] Carole H. Sudre, Wenqi Li, Tom Vercauteren, Sébastien Ourselin, and M. Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. *CoRR*, abs/1707.03237, 2017. 6

[36] Hoang Thanh-Tung and Truyen Tran. Catastrophic forgetting and mode collapse in gans. In *2020 international joint conference on neural networks (ijcnn)*, pages 1–10. IEEE, 2020. 2

[37] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016. 2

[38] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022. 1, 3, 5

[39] Jianyi Wang, Kelvin C. K. Chan, and Chen Change Loy. Exploring clip for assessing the look and feel of images, 2022. 3, 5, 7, 8, 11, 13, 14, 15, 16

[40] Zhaoqing Wang, Yu Lu, Qiang Li, Xunqiang Tao, Yandong Guo, Mingming Gong, and Tongliang Liu. Cris: Clip-driven referring image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11686–11695, June 2022. 2

[41] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1

[42] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019. 1

# Appendix

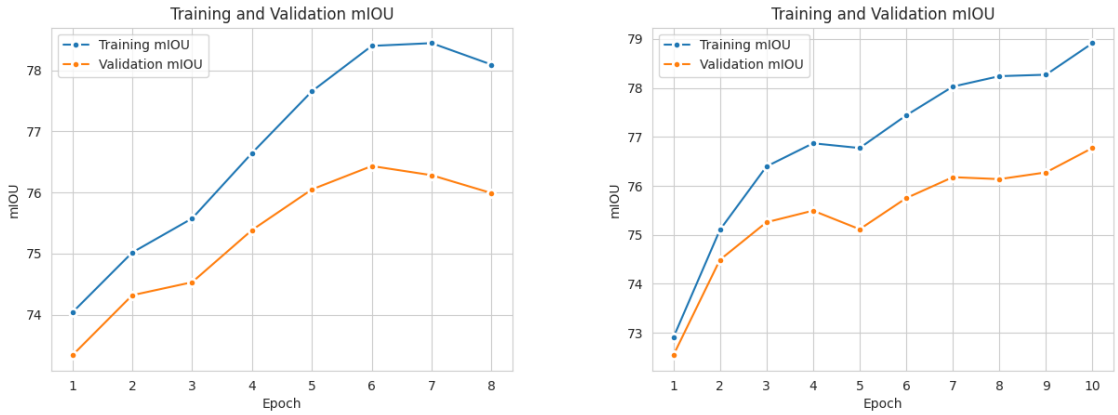
## A. DINO Feature Maps

DINO has demonstrated emergent object decomposition abilities [1]. Thus, we experimented with concatenating its saliency map to the output of the transformer decoder. The concatenated vector is then fed to the MLP, which produces the mask embeddings. This has resulted in worse performances than our improved model, except for the seen setup by a negligible margin, as shown in Figure 9. While these results are not encouraging, other approaches such as LERF [17] have shown that DINO can help regularize grounding models; thus, further research is recommended.

| Model | Dropout | Scoring | Seen | Seen+Unseen | Unseen |
|---|---|---|---|---|---|
| Improved model + DINO | 0.1 | No | 75.95 | 61.43 | 46.70 |
| Improved model + DINO | 0.2 | No | 76.94 | 64.89 | 52.75 |
| Improved model + DINO | 0.1 | Yes | 76.69 | 62.80 | 50.90 |
| Improved model + DINO | 0.2 | Yes | 77.74 | 65.52 | 56.16 |
| Improved model | 0.1 | No | 77.20 | 67.50 | 58.43 |
| Improved model | 0.1 | Yes | **78.05** | **68.08** | **61.77** |

Table 9. **Ablation results using DINO.** Integrated DINO saliency map in the segmentation model of our improved model with different dropout rates and scoring using CLIP-IQA+ [39].

We performed epoch-based training as mentioned in Section 4.4.3 for the improved model with DINO saliency maps as shown in Table 9. We provide the training and the validation mIoU plot for the mentioned models, shown in Figure 9. The plot shows that the model underperforms when compared to our improved model, and the one trained with a 0.1 dropout starts to overfit, as evident in Figure 9a.




(a) Training with 0.1 dropout

(b) Training with 0.2 dropout

Figure 9. **DINO architecture training.** Training and Validation mIoU for the architecture using the DINO saliency maps on the training and test datasets with 10k and 1k samples respectively.

## B. Performance Comparison Using CascadePSP

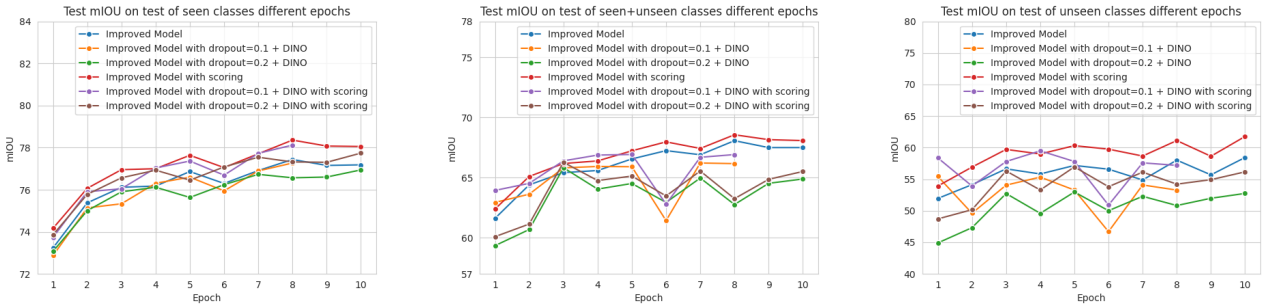
CascadePSP [5] is a deep learning model that refines segmentation masks, producing high-resolution masks from coarse ones. We compared the impact training the grounding model with refined masks has on performance by training it with 1k samples with high-resolution masks and evaluating on the three setups, using 200 samples for each. We also conducted a cross-comparison, in a similar fashion to Section 4.6.2 and, as can be seen in Table 10, this had a minimal impact on performance, improving the mIoU on the seen setup by 1% while decreasing the performance by the same amount on the unseen setup.

| | Standard | | | Refined | | |
|---|---|---|---|---|---|---|
| Training Dataset | Seen | Seen+Unseen | Unseen | Seen | Seen+Unseen | Unseen |
| Standard | 67.96 | 51.61 | 41.90 | 67.14 | 51.08 | 41.65 |
| Refined | 68.51 | 51.35 | 40.90 | 68.14 | 51.20 | 40.77 |

Table 10. **Ablation on generated segmentation masks.** Comparison of models trained using masks produced by Mask R-CNN [24] and masks refined by CascadePSP [5]. All models were trained for 10 epochs with the log-cosh loss using a 0.2 dropout.

## C. Model Performance Across Epochs

We tested the performance of all models mentioned in Table 9 at each epoch using the saved checkpoints for the seen, seen+unseen, and unseen setups, as shown in Figure 10. The plots in the Figure 10 show that our improved model with scoring consistently performs better than the other models.




(a) Testing mIoU on seen classes. (b) Testing mIoU on seen+unseen classes. (c) Testing mIoU on unseen classes.

Figure 10. **Model performance across epochs**: ablation experiment testing the mIoU at different epochs on seen, seen+unseen, and unseen classes for all models mentioned in Table 9.

## D. Funny Results From Stable Diffusion

Figure 11 shows some funny images from our training set generated by Stable Diffusion v1.5, highlighting its limitations in generating images with multiple objects.

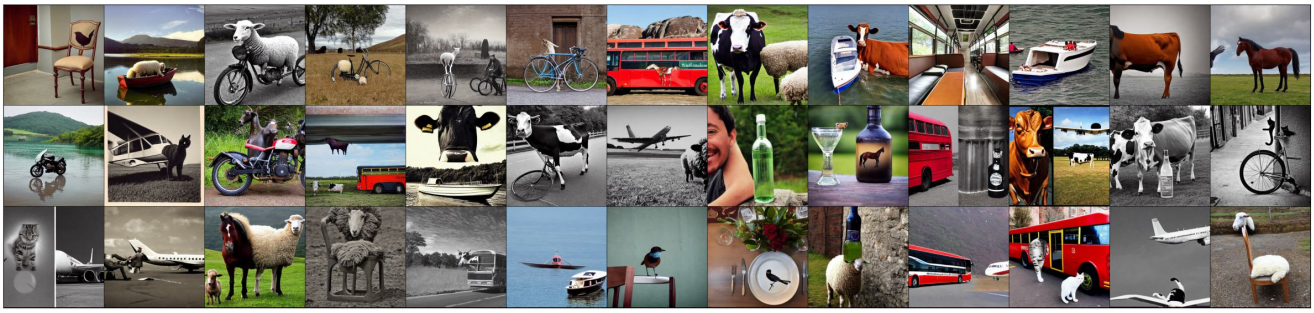


Figure 11. **Funny generations:** a few hand-picked funny images generated by Stable Diffusion from the training set.

## E. Top and Bottom Scored Images From The Training Set

Figures 12 and 13 depict the top 64 and the bottom 64 images on the training set ranked using the CLIP-IQA+ [39] metric.

Figure 12. **Best training images**. Top 64 images from the training set as scored by CLIP-IQA+ [39].

g

Figure 13. **Worst training images**. Bottom 64 images from the training set as scored by CLIP-IQA+ [39].

## F. Top and Bottom Scored Images From The Test Unseen set

Figures 14 and 15 depict the top 64 and the bottom 64 images on the test unseen set ranked using the CLIP-IQA+ [39] metric.

Figure 14. **Best test images**. Top 64 images from the test unseen set as scored by CLIP-IQA+ [39].

Figure 15. **Worst test images**. Bottom 64 images from the test unseen set as scored by CLIP-IQA+ [39].