Aligning Text-to-Image Models using Human Feedback

K Lee, H Liu, M Ryu et al. ICML 2023 (under review)

Hyeonhoon Lee

RI paper study (10th)

2023.03.27.

Aligning Text-to-Image Models using Human Feedback

Kimin Lee [1] **Hao Liu** [2] **Moonkyung Ryu** [1] **Olivia Watkins** [2] **Yuqing Du** [2]
Craig Boutilier [1] **Pieter Abbeel** [2] **Mohammad Ghavamzadeh** [1] **Shixiang Shane Gu** [1]

[1]Google Research [2]University of California, Berkeley. Correspondence to: Kimin Lee <kiminl@google.com>.

Contents

Introduction

Text-to-image models

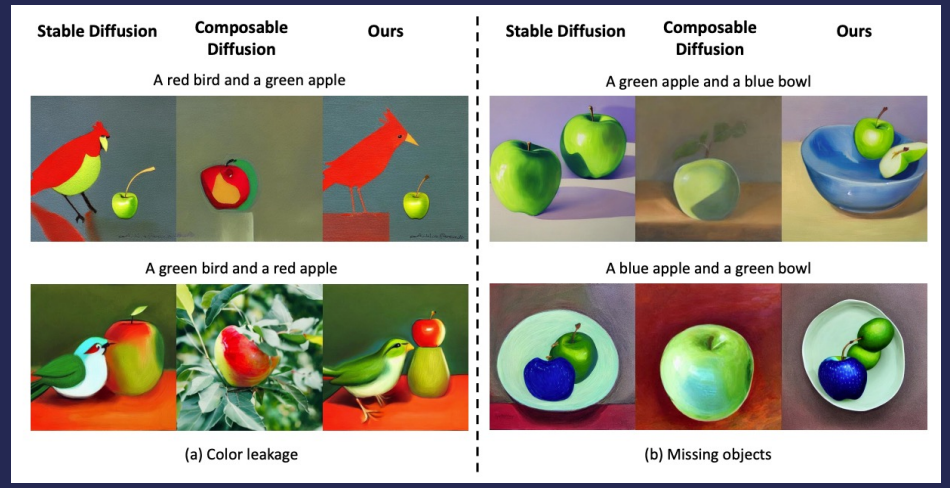
- Advances in image generative models

https://arxiv.org/pdf/2112.10752.pdf
https://arxiv.org/pdf/2212.10562.pdf
https://arxiv.org/pdf/2212.05032.pdf

- Mis-***alignment*** of text and images

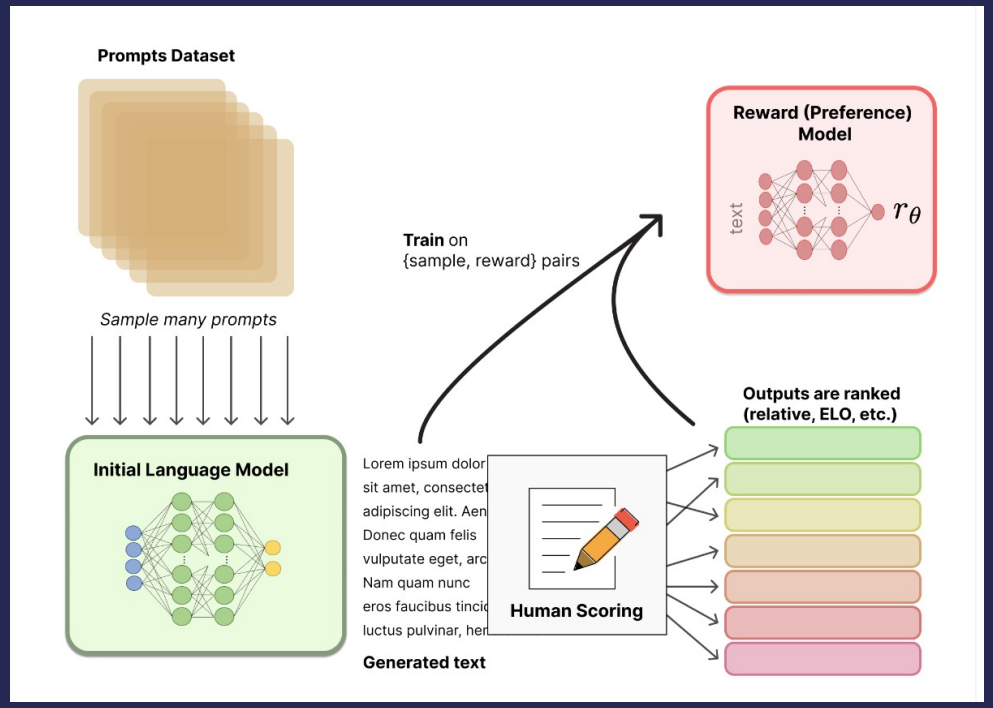
1)Failed to produce reliable visual text.

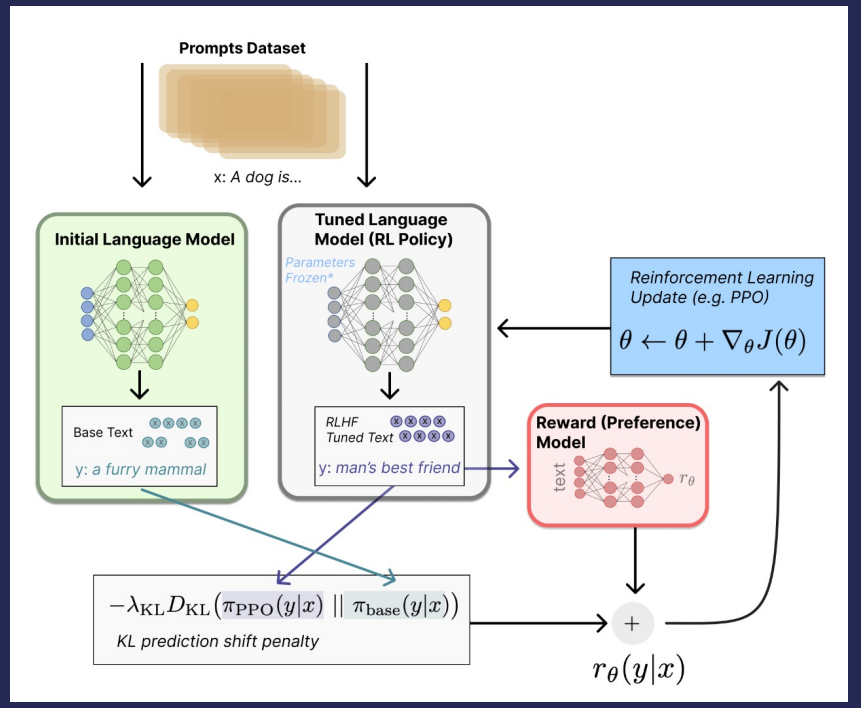
2) Struggling with *compositional* image generation.



Learning with human feedback

- (In language modelling), a powerful solution for aligning model behavior with human feedback
  - 1) Learning a *reward function* intended to reflect humans, using human feedback on model outputs.
  - 2) (Language) model is then optimized using the learned reward function by *RL* algorithm (a.k.a. RLHF).





Human feedback framework

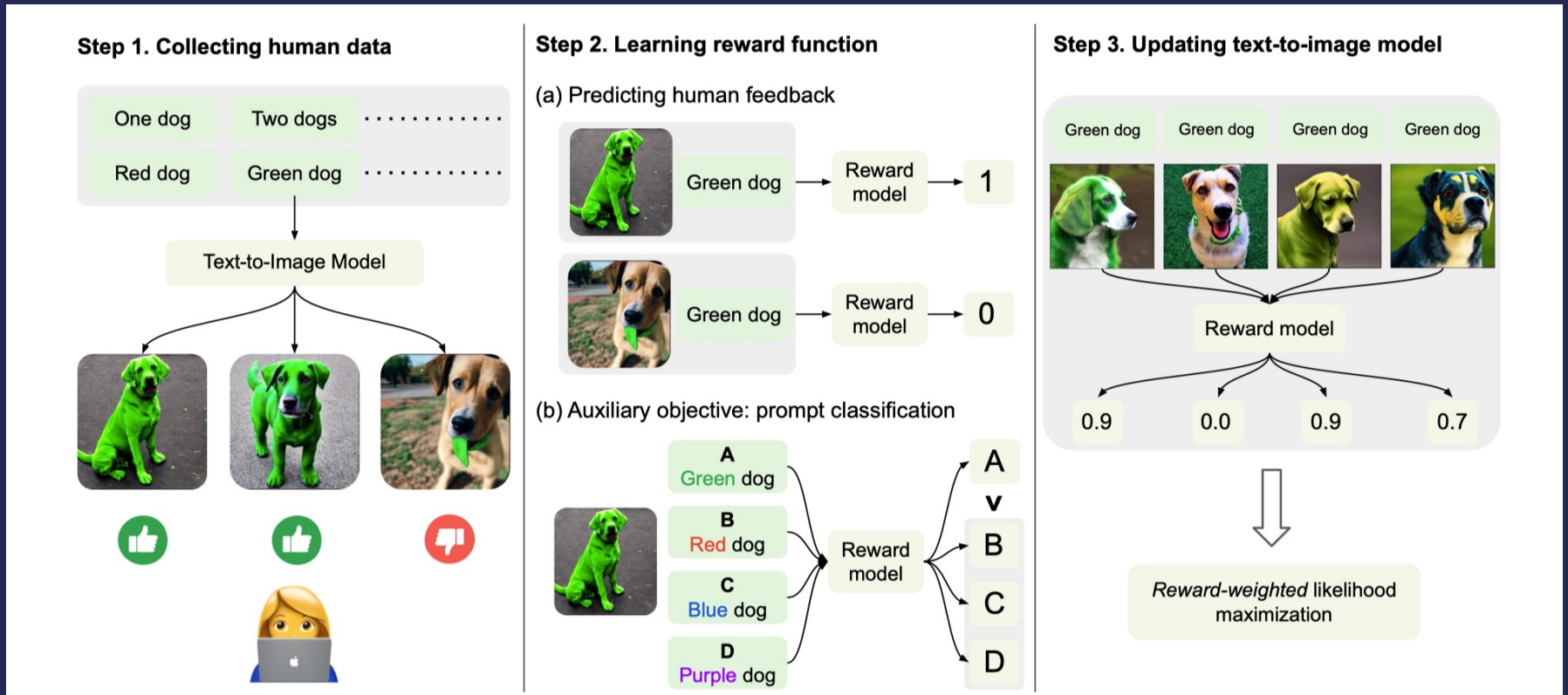


Figure 1. The steps in our fine-tuning method. (1) Multiple images sampled from the text-to-image model using the same text prompt, followed by collection of (binary) human feedback. (2) A reward function is learned from human assessments to predict image-text alignment. We also utilize an auxiliary objective called prompt classification, which identifies the original text prompt within a set of *perturbed* text prompts. (3) We update the text-to-image model via reward-weighted likelihood maximization.

Methods

Human data collection

- Image-text dataset
  - Three categories of text prompts
  - Combination of three categories
  - <60 images per prompt
  - Stable Diffusion v1.5

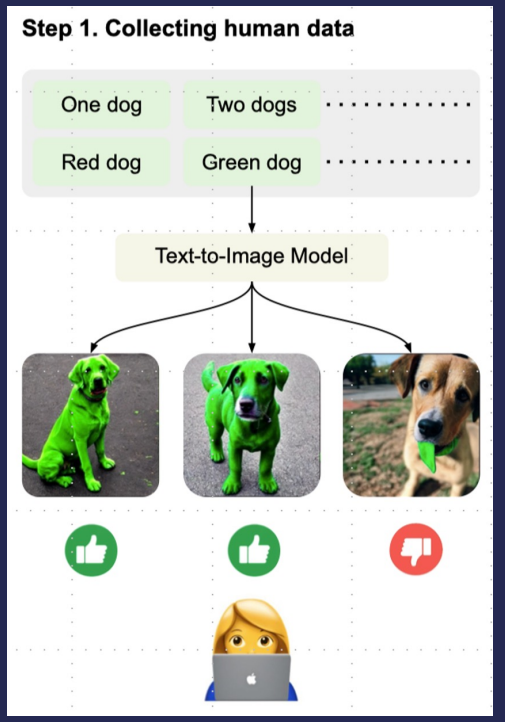
Category	Examples
Count	`One dog; Two dogs; Three dogs; Four dogs; Five dogs;`
Color	`A green colored dog; A red colored dog;`
Background	`A dog in the forest; A dog on the moon;`
Combination	`Two blue dogs in the forest; Five white dogs in the city;`

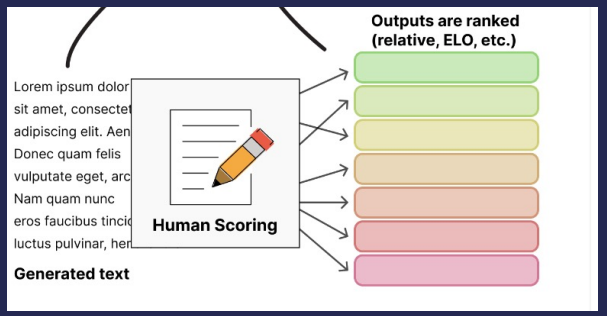
Table 1. Examples of text categories.

- Human feedback
  - Binary option (Good vs. bad)

Category	Total # of images	Human feedback (%)		
		Good	Bad	Skip
Count	6480	34.4	61.0	4.6
Color	3480	70.4	20.8	8.8
Background	2400	66.9	33.1	0.0
Combination	15168	35.8	59.9	4.3
Total	27528	46.5	48.5	5.0

Table 2. Details of image-text datasets and human feedback.





Reward learning

- A reward function: $r_\phi(\mathbf{x}, \mathbf{z})$
  - CLIP embeddings of an image x and text prompt z -> a scalar value
  - Trained to predict human feedback $y \in \{0, 1\}$ (1 = good, 0 = bad).

$$\mathcal{L}^{\text{MSE}}(\phi) = \mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{z},y)\sim\mathcal{D}^{\text{human}}} \left[\left(y - r_\phi(\mathbf{x}, \mathbf{z})\right)^2 \right].$$

- Prompt classification
  - Dataset generation (where y=1) for each image x, and the index i' of the original prompt

$$\mathcal{D}^{\text{txt}} = \{(\mathbf{x}, \{\mathbf{z}_j\}_{j=1}^N, i')\} \text{ with } N \text{ text prompts } \{\mathbf{z}_j\}_{j=1}^N,$$

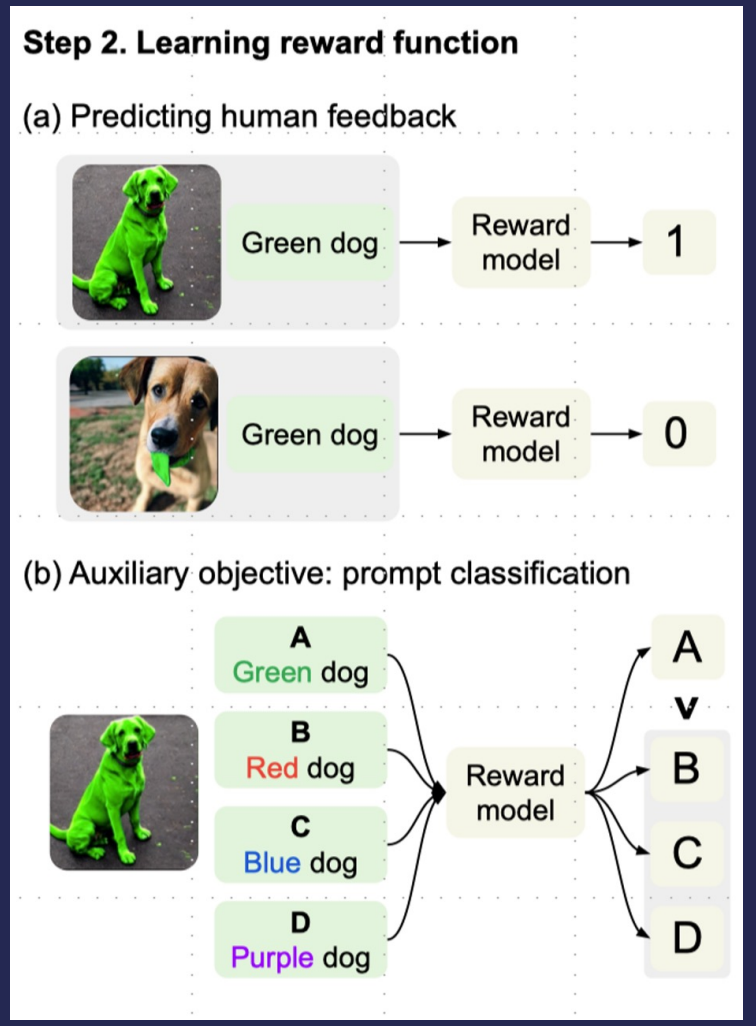
  - Training a prompt classifier (T >0: temperature)

$$P_\phi(i|\mathbf{x}, \{\mathbf{z}_j\}_{j=1}^N) = \frac{\exp(r_\phi(\mathbf{x}, \mathbf{z}_i)/T)}{\sum_j \exp(r_\phi(\mathbf{x}, \mathbf{z}_j)/T)}, \quad \forall i \in [N],$$

$$\mathcal{L}^{\text{pc}}(\phi) = \mathop{\mathbb{E}}_{(\mathbf{x},\{\mathbf{z}_j\}_{j=1}^N,i')\sim\mathcal{D}^{\text{txt}}} \left[\mathcal{L}^{\text{CE}}\left(P_\phi(i|\mathbf{x}, \{\mathbf{z}_j\}_{j=1}^N), i'\right) \right], (1)$$

- Combined loss

$$\mathcal{L}^{\text{reward}}(\phi) = \mathcal{L}^{\text{MSE}}(\phi) + \lambda \mathcal{L}^{\text{pc}}(\phi),$$



Updating the Text-to-image model

- Using the learned reward function, update the text-to-image model p with theta

$$\mathcal{L}(\theta) = \mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{z})\sim\mathcal{D}^{\text{model}}} \left[-r_\phi(\mathbf{x},\mathbf{z}) \log p_\theta(\mathbf{x}|\mathbf{z}) \right] \rightarrow \text{Reward-weighted negative log-likelihood}$$

$$+ \beta \mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{z})\sim\mathcal{D}^{\text{pre}}} \left[-\log p_\theta(\mathbf{x}|\mathbf{z}) \right], \rightarrow \text{Regularization (like PPO-ptx)}$$

  - D^{model} : model-generated dataset (images generated by the text-to-image model on the *tested text prompts*)
  - D^{pre}: pre-training dataset
  - Beta: penalty parameter

Results

Dataset

B. Image-text Dataset

In this section, we describe our image-text dataset. We generate 2774 text prompts by combining a word or phrase from that category with some object. Specifically, we consider 9 colors (`red, yellow, green, blue, black, pink, purple, white, brown`), 6 numbers (`1-6`), 8 backgrounds (`forest, city, moon, field, sea, table, desert, San Franciso`) and 25 objects (`dog, cat, lion, orange, vase, cup, apple, chair, bird, cake, bicycle, tree, donut, box, plate, clock, backpack, car, airplane, bear, horse, tiger, rabbit, rose, wolf`).[15] For each text prompt, we generate 60 or 6 images according to the text category. In total, our image-text dataset consists of 27528 image-text pairs. Labeling for training is done by two human labelers.

For evaluation, we use 120 text prompts listed in Table 4. Given two (anonymized) sets of 4 images, we ask human raters to assess which is better w.r.t. image-text alignment and fidelity (i.e., image quality). Each query is rated by 9 independent human raters in Figure 4 and Figure 6.

Category	Examples
Seen	A red colored dog.; A red colored donut.; A red colored cake.; A red colored vase.; A green colored dog.; A green colored donut.; A green colored cake.; A green colored vase.; A pink colored dog.; A pink colored donut.; A pink colored cake.; A pink colored vase.; A blue colored dog.; A blue colored donut.; A blue colored cake.; A blue colored vase.; A black colored apple.; A green colored apple.; A pink colored apple.; A blue colored apple.; A dog on the moon.; A donut on the moon.; A cake on the moon.; A vase on the moon.; An apple on the moon.; A dog in the sea.; A donut in the sea.; A cake in the sea.; A vase in the sea.; An apple in the sea.; A dog in the city.; A donut in the city.; A cake in the city.; A vase in the city.; An apple in the city.; A dog in the forest.; A donut in the forest.; A cake in the forest.; A vase in the forest.; An apple in the forest.; Two dogs.; Two donuts.; Two cakes.; Two vases.; Two apples.; Three dogs.; Three donuts.; Three cakes.; Three vases.; Three apples.; Four dogs.; Four donuts.; Four cakes.; Four vases.; Four apples.; Five dogs.;
Unseen	A red colored bear.; A red colored wolf.; A red colored tiger.; A red colored rabbit.; A green colored bear.; A green colored wolf.; A green colored tiger.; A green colored rabbit.; A pink colored bear.; A pink colored wolf.; A pink colored tiger.; A pink colored rabbit.; A blue colored bear.; A blue colored wolf.; A blue colored tiger.; A blue colored rabbit.; A black colored rose.; A green colored rose.; A pink colored rose.; A blue colored rose.; A bear on the moon.; A wolf on the moon.; A tiger on the moon.; A rabbit on the moon.; A rose on the moon.; A bear in the sea.; A wolf in the sea.; A tiger in the sea.; A rabbit in the sea.; A rose in the sea.; A bear in the city.; A wolf in the city.; A tiger in the city.; A rabbit in the city.; A rose in the city.; A bear in the forest.; A wolf in the forest.; A tiger in the forest.; A rabbit in the forest.; A rose in the forest.; Two brown bears.; Two wolves.; Two tigers.; Two rabbits.; Two red roses.; Three brown bears.; Three wolves.; Three tigers.; Three rabbits.; Three red roses.; Four brown bears.; Four wolves.; Four tigers.; Four rabbits.;

Table 4. Examples of text prompts for evaluation.

Human evaluation

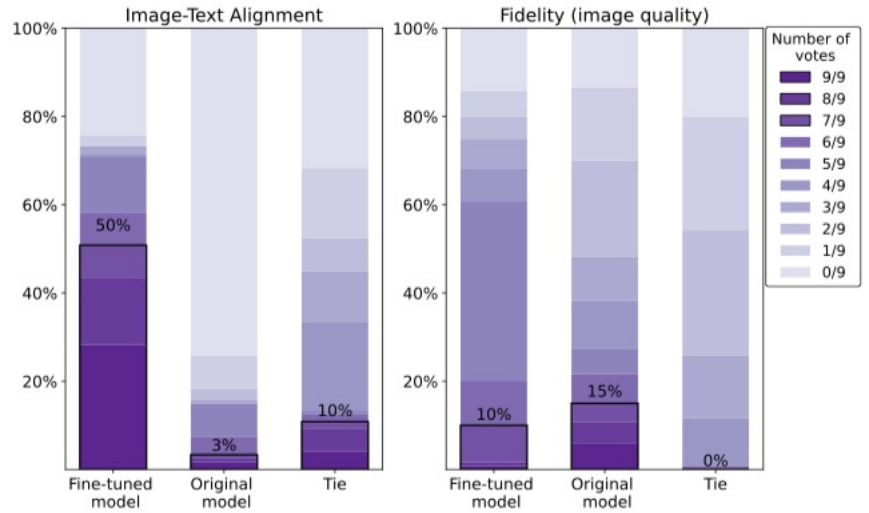


Figure 4. Human evaluation results on 120 text prompts (60 seen text prompts and 60 unseen text prompts). We generate two sets of images (one from our fine-tuned model and one from the original model) with same text prompt. Then, human raters indicate which one is better, or tie (i.e., two sets are similar) in terms of image-text alignment and image fidelity. Each query is evaluated by 9 independent human raters and we report the percentage of queries based on the number of positive votes. We also highlight the percentage of queries with two-thirds vote (7 or more positive votes) in the black box.

Qualitative comparison

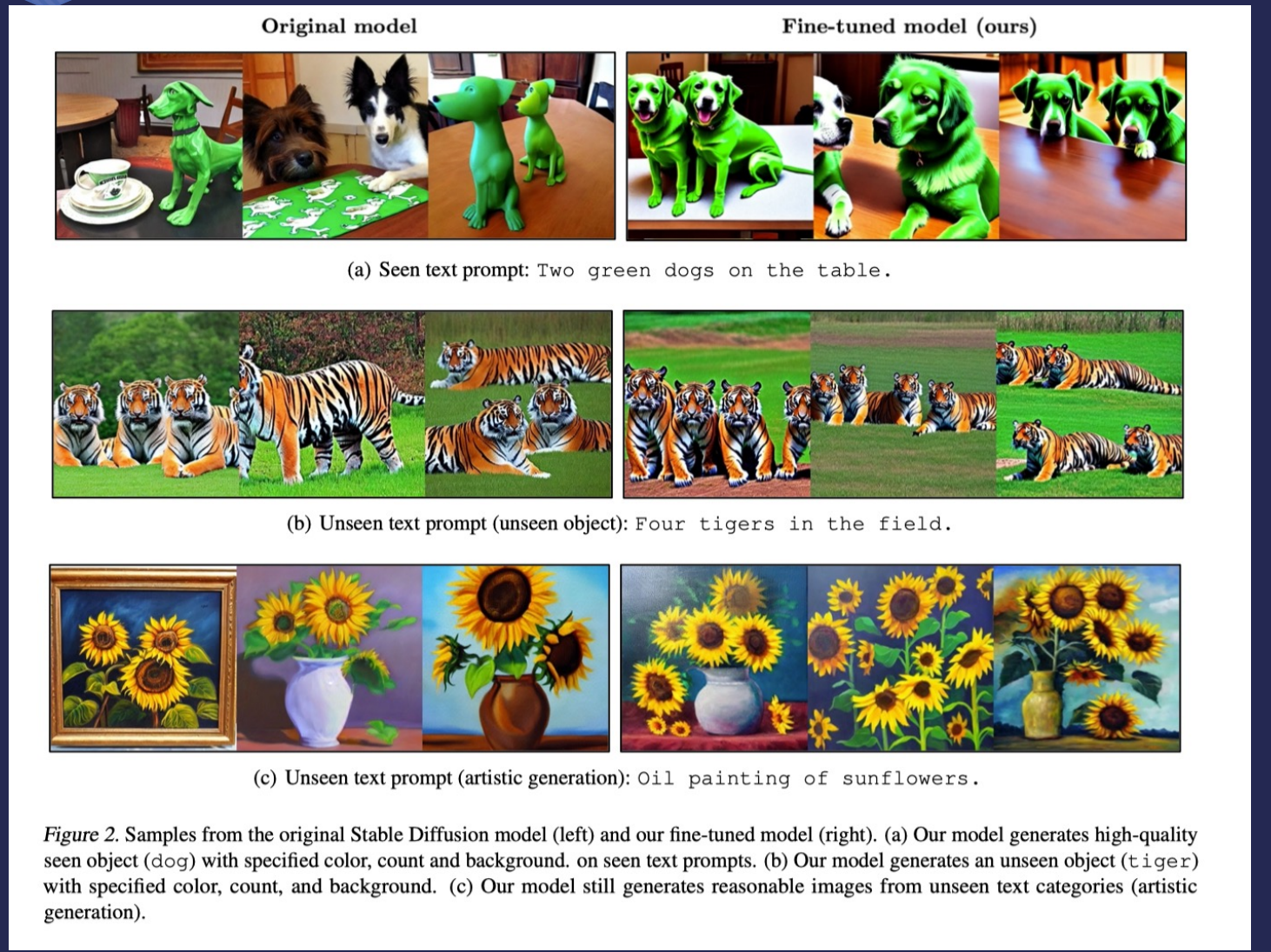
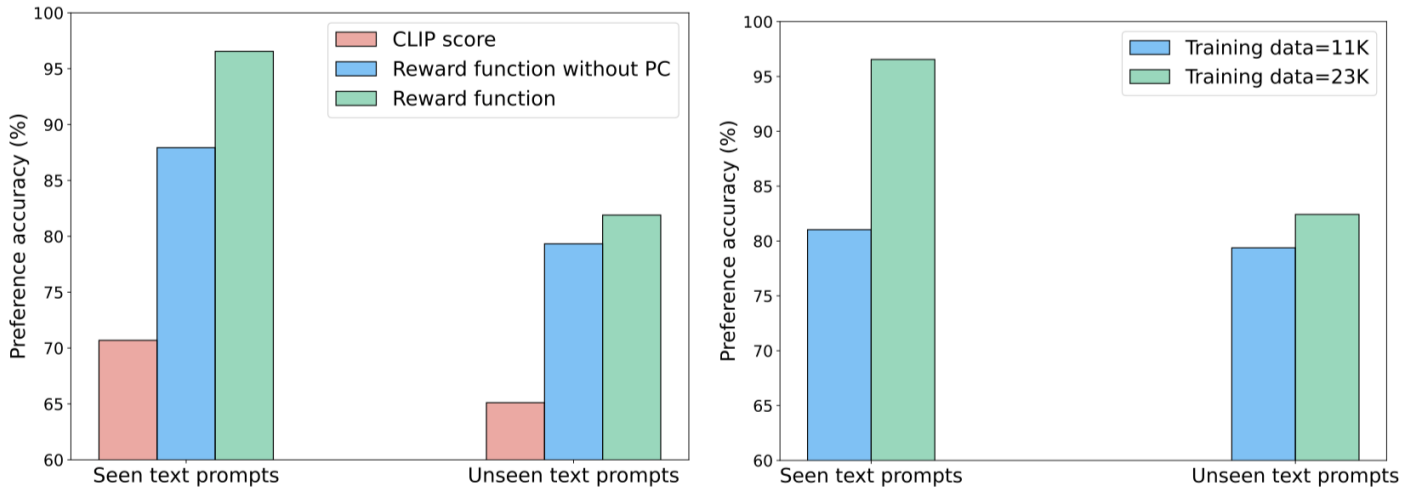


Figure 2. Samples from the original Stable Diffusion model (left) and our fine-tuned model (right). (a) Our model generates high-quality seen object (`dog`) with specified color, count and background. on seen text prompts. (b) Our model generates an unseen object (`tiger`) with specified color, count, and background. (c) Our model still generates reasonable images from unseen text categories (artistic generation).

Results on Reward Learning

- Predicting human preferences
  - vs. CLIP score: a measure of image-text similarity in the CLIP embedding space



(a) Comparison with CLIP score (Radford et al., 2021)

(b) Performance of reward function versus data size

Figure 3. (a) Accuracy of CLIP score (Radford et al., 2021) and our reward functions on predicting the preferences of human labelers. Fo
our method, we consider a variant of our reward function, which is not trained with prompt classification (PC) loss in (1). (b) Performanc
of reward functions with varying the size of the training dataset.

Results on Reward Learning

- Rejection sampling
  - selects the best output w.r.t. the learned reward function.
  - Specifically, we generate 16 images per text prompt from the original stable diffusion model and select the 4 with the highest reward scores.
  - We compare these to 4 randomly sampled images in Figure 6(a).

(a) Original model versus Rejection sampling

(b) Fine-tuned model versus Rejection sampling

Figure 6. Human evaluation on 120 tested text prompts (60 seen text prompts and 60 unseen text prompts). We generate two sets of images with same text prompt. Then, human raters indicate which one is better, or tie (i.e., two sets are similar) in terms for image-text alignment and image fidelity. Each query is evaluated by 9 independent human raters and we report the percentage of queries based on the number of positive votes. We also highlight the percentage of queries with two-thirds vote (7 or more positive votes) in the black box. (a) For rejection sampling, we generate 16 images per text prompt and select best 4 images based on reward score, i.e., more inference-time compute. (b) Comparison between fine-tuned model and original model with rejection sampling.

Ablation studies

- Human dataset size

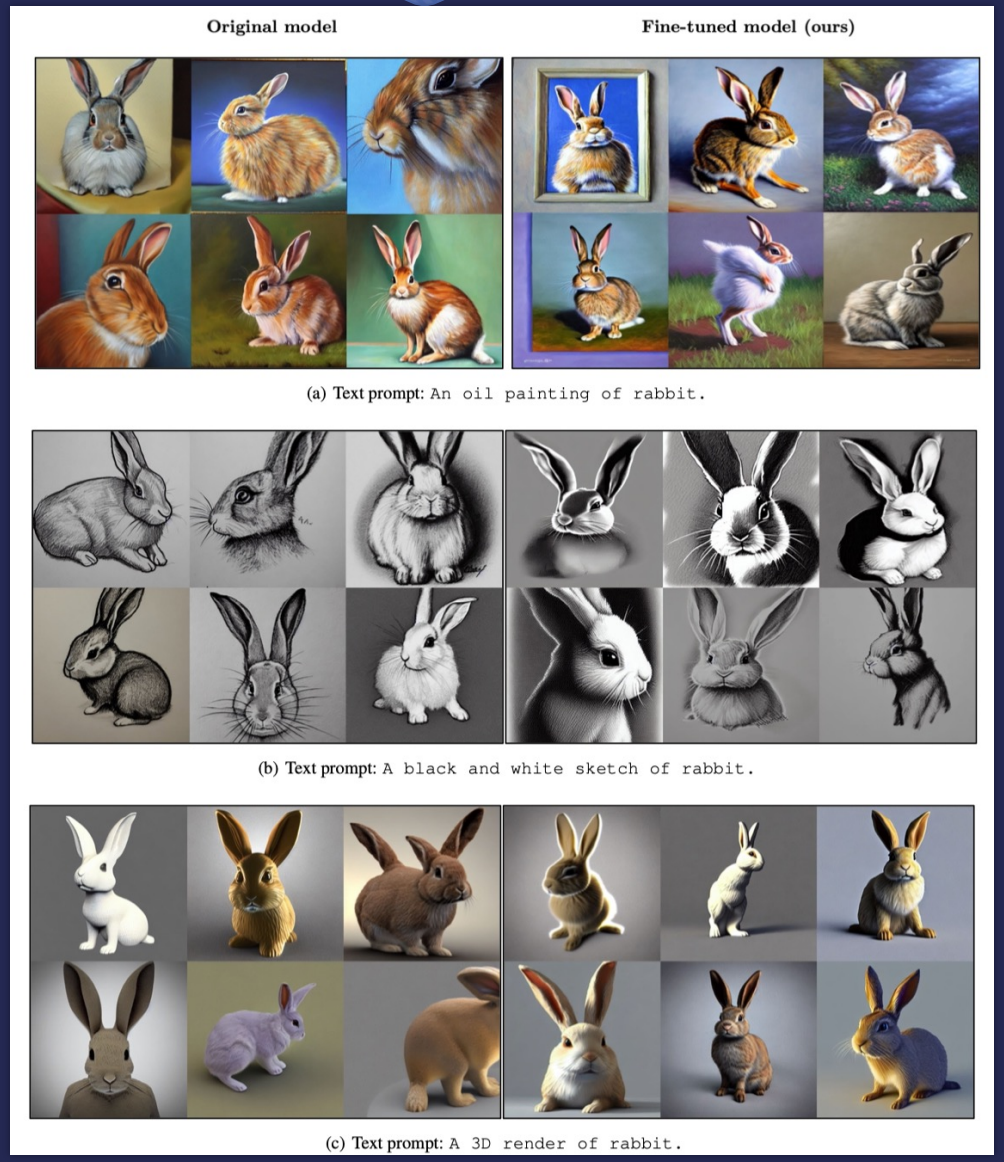
(b) Performance of reward function versus data size

- Diverse datasets

	FID on MS-CoCo (↓)	Average rewards on tested prompts (↑)
Original model	13.97	0.43
Fine-tuned model w.o unlabeled & pre-train	26.59	0.69
Fine-tuned model w.o pre-train	21.02	0.79
Fine-tuned model	16.76	0.79

Table 3. Comparison with the original Stable Diffusion. For evaluating image fidelity, we measure FID scores on the MS-CoCo. For evaluating the image-text alignment, we measure reward scores and CLIP scores on 120 tested text prompts. ↑ (↓) indicates that the higher (lower) number is the better.

Appendix. Qualitative comparison



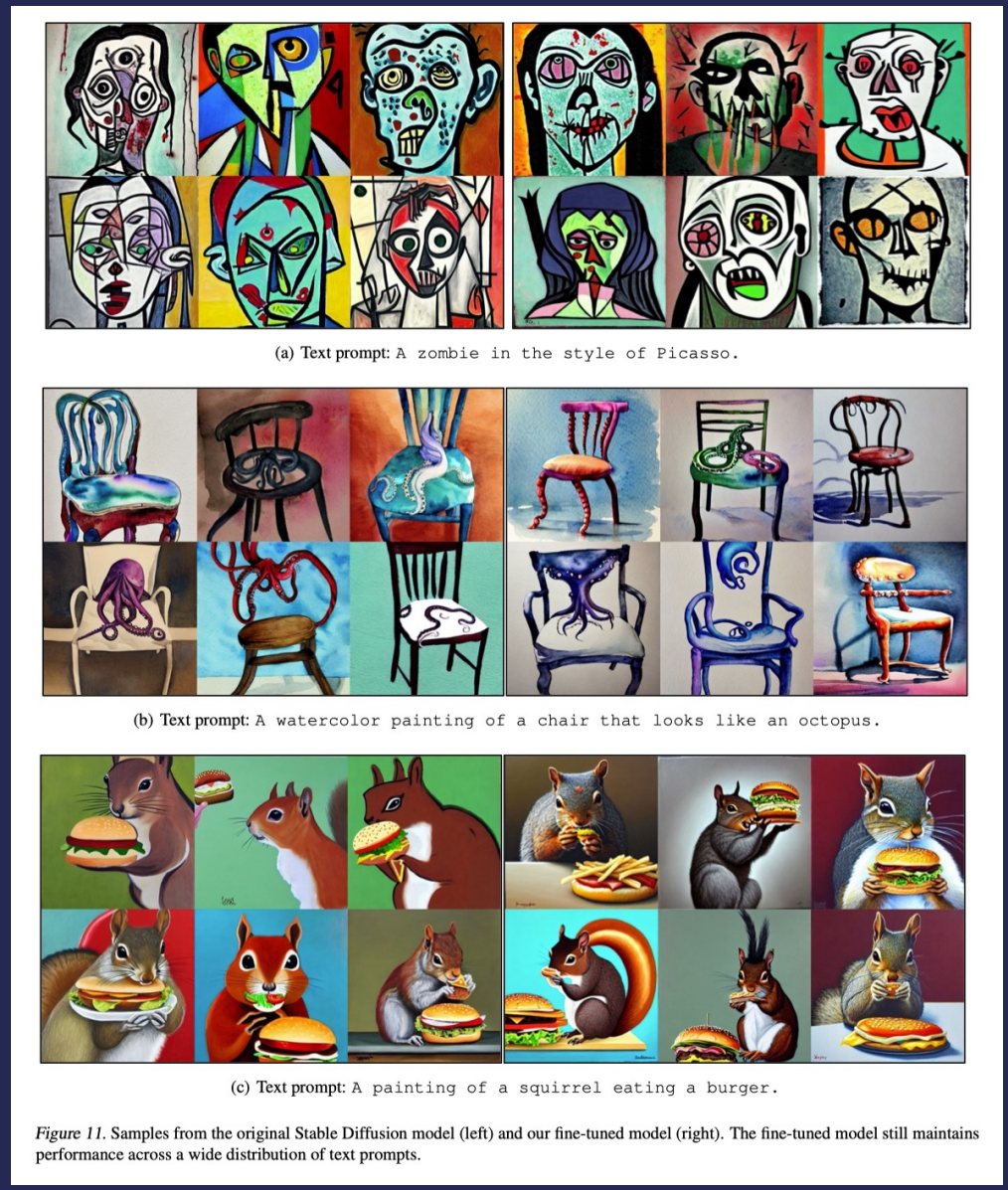


Figure 11. Samples from the original Stable Diffusion model (left) and our fine-tuned model (right). The fine-tuned model still maintains performance across a wide distribution of text prompts.

Appendix. Pseudocode

Algorithm 1 Reward Learning Pseudocode

```
# x, z, y: image, text prompt, human label
# clip: pre-trained CLIP model
# preprocess: Image transform
# pred_r: two-layers MLPs
# Get_perturbated_prompts: function to generate perturbated text prompts
# lambda: penalty parameter
# T: temperature
# N: # of perturbated text prompts

# main model
def RewardFunction(x, z):
    # compute embeddings for tokens
    img_embedding = clip.encode_image(prepocess(x))
    txt_embedding = clip.encode_text(clip.tokenize(z))
    input_embeds = concatenate(img_embedding, txt_embedding)

    # predict score
    return pred_r(input_embeds)

# training loop
for (x, z, y) in dataloader: # dims: (batch_size, dim)
    # MSE loss
    r_preds = RewardFunction(x, z)
    loss = MSELoss(r_preds, y)

    # Prompt classification
    scores = [r_preds]
    for z_neg in Get_perturbated_prompts(z, N):
        scores.append(RewardFunction(x, z_neg))
    scores = scores / T
    labels = [0] * batch_size # origin text is always class 0
    loss += lambda * CrossEntropyLoss(scores, labels)

    # update reward function
    optimizer.zero_grad(); loss.backward(); optimizer.step()
```

Appendix. Pseudocode

Algorithm 2 Perturbated Text Prompts Generation Pseudocode

```
# z: image, text prompt, human label
# N: # of perturbated text prompts

def Get_perturbated_prompts(z, N):
    color_list = [``red'', ``yellow'', ...]
    obj_list = [``dog'', ``cat'', ...]
    count_list = [``One'', ``Two'', ...]
    loc_list = [``in the sea.'', ``in the sky.'', ...]

    output = []
    count = 0
    while (count < N):
        idx = random.randint(0, len(count_list)-1)
        count = count_list[idx]
        idx = random.randint(0, len(color_list)-1)
        color = color_list[idx]
        idx = random.randint(0, len(loc_list)-1)
        loc = loc_list[idx]
        idx = random.randint(0, len(obj_list)-1)
        obj = obj_list[idx]

        if count == ``One'':
            text = ``{} {} {} {}.''.format(count, color, obj, loc)
        else:
            text = ``{} {} {}s {}.''.format(count, color, obj, loc)

        if z != text:
            count += 1
            output.append(text)
    return output
```


Thank you