Introducing LoRA: A faster way to fine-tune Stable Diffusion

Posted February 7, 2023 by @cloneofsimo, @andreasjansson, @anotherjesse, and @zeke

Last year, DreamBooth was released. It was a way to train Stable Diffusion on your own objects or styles.

A few short months later, Simo Ryu has created a new image generation model that applies a technique called LoRA to Stable Diffusion. Similar to DreamBooth, LoRA lets you train Stable Diffusion using just a few images, and it generates new output images with those objects or styles. Unlike DreamBooth, LoRA is *fast*: While DreamBooth takes around twenty minutes to run and produces models that are several gigabytes, LoRA trains in as little as eight minutes and produces models that are around 5MB.

LoRA stands for Low-Rank Adaptation, a mathematical technique to reduce the number of parameters that are trained. You can think of it like creating a diff of the model, instead of saving the whole thing. LoRA was developed by researchers at Microsoft, and Simo has applied it to Stable Diffusion. Check out the README for Simo's inference model on GitHub and the paper on arXiv to learn more about how it works.

We've been collaborating with Simo to get LoRA up on Replicate. You can now train LoRA models in the cloud with a single API call. Unlike DreamBooth where you had to wait for a model to push and boot up, LoRA predictions run instantly with no cold boots.

What’s unique about LoRA?

LoRA has a few differences from DreamBooth that make it especially appealing as an alternative:

- **Faster training**: Training a new concept with LoRA takes just a few minutes.
- **Smaller outputs**: Trained LoRA outputs are much smaller than DreamBooth outputs. This makes them easier to share, store, and re-use.
- **Multiple concepts**: You can combine multiple trained concepts in a single image. (This feature is still experimental, but we're working on improving it. 🧪)
- **Faster image generation**: When you train your own DreamBooth model on Replicate, the model only stays warm when you're actively using it. With LoRA, you're not running your own model, but rather running the one cloneofsimo/lora model, which is always on and ready to serve predictions.
- **Better at styles, worse at faces.** Based on our experimentation, LoRA seems to do a better job at styles than DreamBooth, but faces aren't as good. They are stuck in uncanny valley, rather than looking precisely like the person. Your results might be better than ours though, so let us know how you get on.

How to use LoRA

> 🐴 To get an idea of what's possible, check out the LoRA examples page, where you can play around with some of our pretrained concepts like Bob Ross, Pokemon, South Park, Caravaggio, and more.

To train your own reusable LoRA concept, you'll do the following:

1. Gather training images in a zip file.
2. Upload your training images to a publicly accessible URL.
3. Use one of LoRA's training models to **train your concept**.
4. Save the URL of your trained output.
5. Use LoRA's prediction model to **generate new images** with your trained concept.

Step 1: Gather training images

To train a new LoRA concept, create a zip file with a few images of the same face, object, or style. 5-10 images are enough, but for styles you may get better results if you have 20-100 examples. Many of the recommendations for training DreamBooth also apply to LoRA. The training images can be JPGs or PNGs.

💡 Give your zip file a meaningful name, as it will be included as part of the filename of the trained output. This will make it easier to identify and differentiate from other training outputs later.

Step 2: Upload training images

LoRA's training model expects your images to be accessible over HTTP at a public URL. You can use a service like Google Drive, Amazon S3, or GitHub Pages to host your zip file.

You can upload files to Replicate if you don't have a cloud bucket to store the zip file. Here's a Python script that uploads a file to Replicate and returns a URL that you can use in the training model:

```
import os
import requests

zip_path = "/path/to/my-training-images.zip"
zip_filename = zip_path.split("/")[-1]

# Upload inputs to cloud storage.
# You can skip this step if your zip file is already on the internet and accessible over HTTP
upload_response = requests.post(
    "https://dreambooth-api-experimental.replicate.com/v1/upload/" + zip_filename,
    headers={"Authorization": "Token " + os.environ["REPLICATE_API_TOKEN"]},
).json()

with open(zip_path, "rb") as f:
    requests.put(upload_response["upload_url"], data=f)
zip_url = upload_response["serving_url"]
```

Step 3: Train your concept

There are two LoRA training models on Replicate:

- **replicate/lora-training** has preset options for face/object/style that we've found to be optimal for those use cases.
- **replicate/lora-advanced-training** lets you set the options yourself if you want full control of the model.

Start by using the lora-training model to train your concept. Here's an example Python script that uses the training model to train a new concept:

```
import replicate

# Zip file containing input images, hosted somewhere on the internet
zip_url = "https://my-storage/my-input.zip"

# Train the model
training_model = replicate.models.get("replicate/lora-training")
```

```
version = training_model.versions.get("b2a308762e36ac48d16bfadc03a65493fe6e799f429f7941639a6acec5b276cc")
lora_url = version.predict(instance_data=zip_url, task="style")
```

Step 4: Save the URL of your trained output

The output of each training run is a single `.safetensors` file at an HTTPS URL that we host indefinitely.

For example, `https://replicate.delivery/pbxt/S8wVStOvXr5mEFDjP5XkmMPjLPCaDmv1Rw6AzRMDEhoFqqGE/tmp_fs4evyhbob-ross.safetensors`

Copy the URL of that trained concept file from your prediction response so you can use it as an input to LoRA's prediction model.

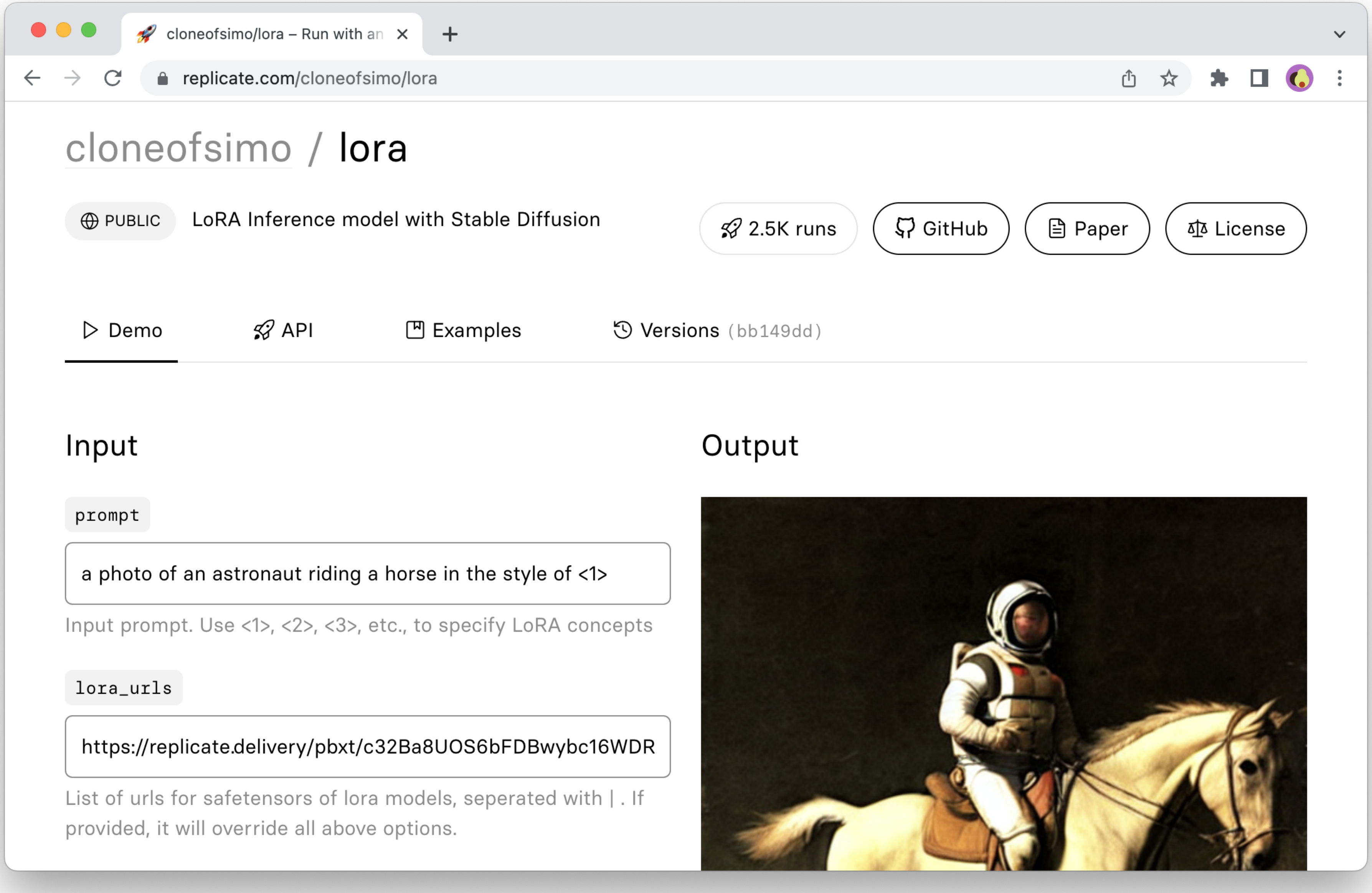
Step 5: Generate images

Now that you've got a trained concept, it's time to generate some new images! You can generate an image based on a single trained concept, or use multiple trained concepts together.

The prediction model replicate/lora requires two inputs:

- `prompt` : A prompt that contains the string `<1>` where the trained concept should be, e.g. `an astronaut riding a horse in the style of <1>` . Use `<2>` , `<3>` if you're passing multiple URLs to the `lora_urls` input.
- `lora_urls` : The URL or URLs of your trained LoRA concept(s) you copied in the previous step. You can pass a single URL, or a list of URLs separated by a pipe character `|` . Passing multiple URLs allows you to combine multiple concepts into a single image.

You can run LoRA's prediction model from your browser:


cloneofsimo/lora – Run with an
replicate.com/cloneofsimo/lora

cloneofsimo / lora

PUBLIC LoRA Inference model with Stable Diffusion

2.5K runs
GitHub
Paper
License
Demo
API
Examples
Versions (bb149dd)

Input

prompt

a photo of an astronaut riding a horse in the style of <1>

Input prompt. Use <1>, <2>, <3>, etc., to specify LoRA concepts

lora_urls

https://replicate.delivery/pbxt/c32Ba8UOS6bFDBwybc16WDR

List of urls for safetensors of lora models, seperated with | . If provided, it will override all above options.

Output

You can also run LoRA's prediction model using Replicate's API. Here's an example Python script that uses the API to generate a new image:

```
import replicate

# Run a prediction
predict_model = replicate.models.get("replicate/lora")
predict_model_version = predict_model.versions.get("97ec1b97e5e6a6476e45ba7211d368509bbf39c30a927e39637f3cb98b36ac91")
lora_url = "https://replicate.delivery/pbxt/S8wVStOvXr5mEFDjP5XkmMPjLPCaDmv1Rw6AzRMDEhoFqqGE/tmp_fs4evyhbob-ross.safetensors"
output_url = predict_model_version.predict(prompt="a painting of dinosaur in the style of <1>", lora_urls=lora_url)
```

Next steps

In the next couple of weeks we'll add support for training LoRA on Stable Diffusion 2.1, inpainting, and other cool things. Let us know your ideas!

If you want to share your LoRA models with the community or see what others come up with, join the **#lora channel** in our Discord.