Nithyashree Manohar
Stable Diffusion - Thumbs Up Gesture

1. Data Preparation

For the training step, we need class images and instance images. Class images should contain images of the broader class, in our case different people showing a thumbs up. Instance images are specific to the image we want to generate, i.e. a specific person showing a thumbs up. The images in the dropbox link are used for the class images. For instance images, multiple images of me showing the thumbs up gesture are captured in different settings. A total of 121 class images and 116 instance images are used.

Example instance images

2. Training approach

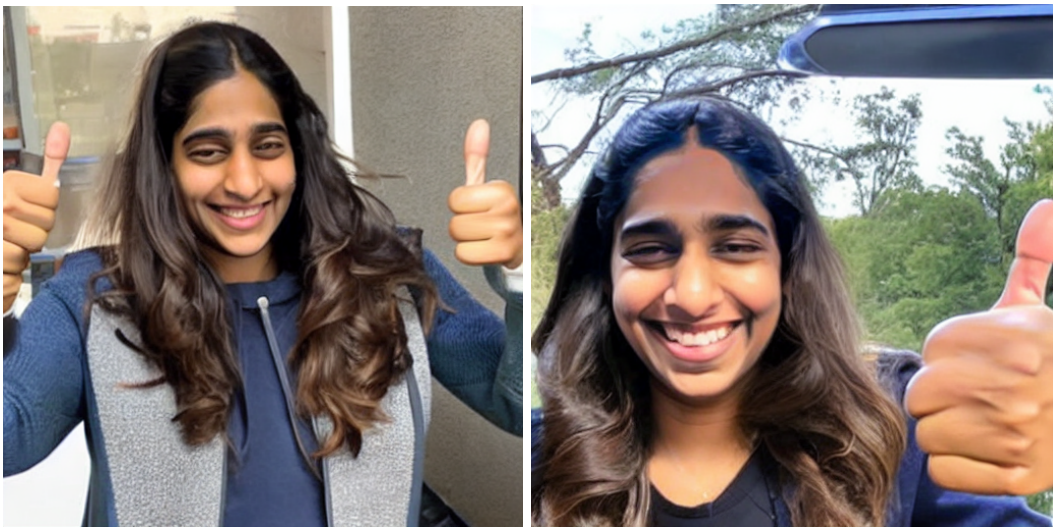
- The Dreambooth technique is used to teach new concepts to Stable-Diffusion using a specialized form of fine-tuning.
- For the purpose of fine-tuning, Stable Diffusion 1.5 (runwayml/stable-diffusion-v1-5) is used.
- Several experiments with different parameterizations were conducted to analyze the effect of different settings in Dreambooth. The experiments were conducted in a Google Colab notebook with a GPU runtime.
- To get good-quality images, I had to find a balance between the number of training steps and the learning rate until the results were satisfactory. Experiments with a low learning rate (2e-6) resulted in poor performance with the images not resembling the target person. Higher learning rates (1e-5) gave a satisfactory result while not being the optimal. Increasing the learning rate improved performance.
- The number of class images were increased to 200 by generating additional images using the generative model itself. This step was a part of the training script.
- Empirically, **800-1000** steps worked well when using a batch size of **1** (with a single GPU) and learning rate of **5e-5**.

- Prior preservation is important to avoid overfitting when training on faces. This was provided through the class images.

3. Results

We generated and displayed 20 images for the purpose of evaluation. Visually, all images resemble the target person and display the thumbs up gesture. The prompt used was, "*Alexa with a thumbs up gesture*" along with a negative prompt, "*low quality, blurry, unfinished, moustache*". Without mentioning "moustache" in the negative prompt the model generated male-looking faces with facial hair, which speaks to the inherent biases in these models.

Two of the generated images are shown below. The images are particularly good at capturing hair and eyebrows, and all contain the thumbs up.



4. Evaluation

To automate result evaluation, we used a pretrained ResNet model and further fine-tuned it to identify the thumbs up gesture. The model was trained on thumbs-up images using the previously used class and instance images, contrasted with general images of humans without any gestures from Kaggle. 20% of the images were held out for validation. The trained model achieved a 90% validation accuracy.

This model classified 18 out of the 20 generated images to contain the thumbs-up gesture. Using the automated evaluation 90% of the generated images match the criteria, which satisfies the expected criterion of 75%.

5. Deployment

- Training and evaluating diffusion model - Providing better training data to the model to see if the model can perform better
- Choosing an appropriate deployment environment - AWS, Google Cloud, or Azure
- Containerizing the model: Packaging the diffusion model along with its dependencies into a container. Containerization makes it easier to deploy and manage the model across different environments. Eg. Docker
- Monitoring and maintaining model: Regularly monitor the performance of the deployed diffusion model to detect any issues or deviations from expected behavior.
- Continuous integration and deployment (CI/CD): If we plan to make updates or improvements to the diffusion model over time, we should consider implementing a CI/CD pipeline. This allows us to automate the process of building, testing, and deploying new versions of the model, ensuring a smooth and efficient update process.
- Version control: Maintaining version control for the deployed models and associated code. This allows us to track changes, rollback to previous versions if necessary, and collaborate with a team.
- Regularly updating and retraining the model: Diffusion models, like other machine learning models, may benefit from periodic updates and retraining. As new data becomes available or as we identify areas for improvement, retraining the model using updated data and deploying the new version.