On The Robustness of Diffusion-Based Text-to-Image Generation

Team 14
Members: Liang Chen, Zhe Yang, Zheng Li

Abstract

In this report we introduce our exploration on the robustness of diffusion-based text2image model. Starting from analysing the challenge, we conclude two stochastic properties that would influence the robustness of image generation. We further propose our robustness metrics and data augmentation methods to increase the robustness. At last, we analyze how different data augmentation methods influence the robustness of diffusion model and discuss the limitation and future work.

Work Allocation is at the end of the report.

1 Introduction

1.1 Diffusion Model

Diffusion Models are generative models, meaning that they are used to generate data similar to the data on which they are trained. Fundamentally, Diffusion Models work by destroying image through the successive addition of Gaussian noise, and then learning to recover the image by reversing this noising process. After training, we can use the diffusion model to generate image by simply passing randomly sampled noise through the learned denoising process.

1.2 High Computational Cost of Diffusion Model

However, Diffusion model is extremely computationally demanding, since training and evaluating such a model requires repeated computations in the high-dimensional pixel space of RGB images. As an example, training the most powerful Diffusion Models often takes hundreds of GPU days and the model may need 1000 iterations to generate an image when evaluating. High computational complexity makes it hard to train or evaluate diffusion model with poor computational resources. To deal with this problem, we adopt some existing methods to accelerate sampling speed and start from pre-trained diffusion model to do our experiments.

1.3 Robustness Problem

Another problem for Diffusion model is that the random Gaussian noise may make the model not robust. For example, for text2img diffusion model, we want the model to generate similar and faithful images when given texts with similar meanings. However, text2img diffusion does not always work as we excepted, which is to say sometimes when we transform the text' syntax structure without changing its semantic meaning, the text2img diffusion model may not fully understand the text that has changed in syntax structure and can hardly generate faithful relevant images. This problem is what we are trying to solve.

Preprint. Under review.

2 Challenge and Method

2.1 Two Stochastic Properties of Diffusion Text2Image Model

We think there are two kinds of stochastic properties that will influence the robustness of a diffusion based text2image model.

The first one is the inner stochastic nature of diffusion model. Since the diffusion model restores the image from random gaussian noise, it might result in very different output when different seeds are given as shown in the slide, This sometimes can be beneficial since it increases the diversity of the output. But it can also be problematic since we can't control the output image. As shown in Figure 1, by changing different seeds we can get very different image outputs based on the same text prompt.

A man and a women posing next to one another in front of a table.

Figure 1: Picture generated with different random seeds.

The second stochastic property is from the text signal. Similar text inputs might lead to images with very different quality. This is a big problem since it suggests that the model is not stable to a variant of text input. As shown in Figure 2, the three captions all have cat in the sentences, and their meanings are similar, however only the first one has a nice cat. The other cats are very strange.

Generated with similar prompts

A cat lying a top on the desk on a laptop.

A cat is laying on top of a laptop computer

A cat sprawled out over the top of a laptop computer keyboard.

Ground Truth

Figure 2: Picture generated with similar prompts.

2.2 Measurement of Robustness

The measurement of robustness is a hard problem since there is no commonsense or public benchmark defining the degree of robustness. However, without automatic evaluation metric it would be much harder for human to judge the robustness of image generation. So we define following the metrics for us to understand the robustness of text2image models:

1. Image Similarity among random seeds
2. Image Similarity among similar text inputs

We adopt CLIP [1] model to compute the Image similarity. Details can be found in Section 4. Above from the robustness scores, we also compute the traditional FID[1] [2] score to evaluate the quality of generated pictures. We also agree that there might be better metrics or evaluation methods that are worth future explorations.

[1] https://github.com/mseitzer/pytorch-fid

2.3 Goal and Methods

Towards a more robust text2image model, we want to increase the density of annotation to support more robust training. As shown in Figure 3, one image might correspond to different points in the text space, and our goal is to augment the points in the text space to stabilize the training.

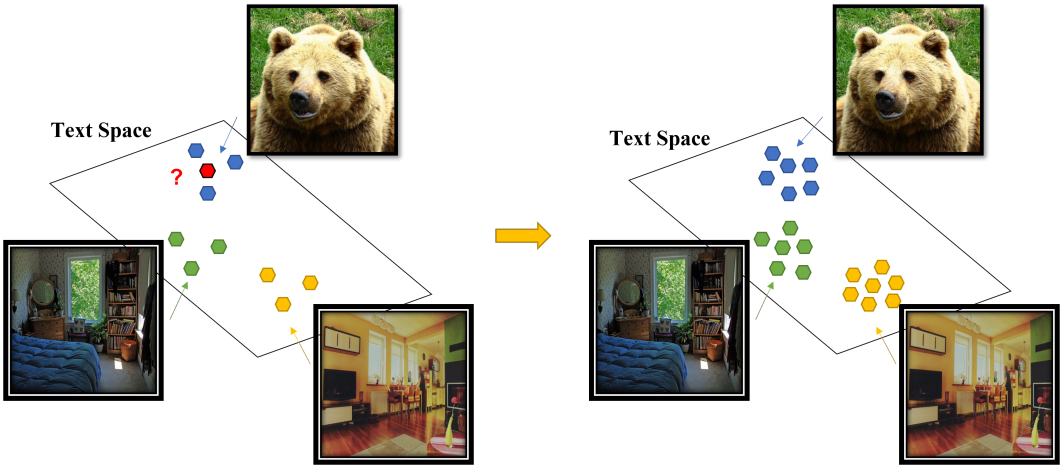

Figure 3: Increasing the density of the annotations could lead to more robust training.

To achieve this, we explore three different kinds of data augmentation methods to assist the diffusion-based text-to-image generation.

The first one is **word manipulating** method where we use random deleting methods that random deleting words of the sentence to generate similar sentence example. **Back-translation** is another powerful text augment method where we first translate the English caption to German, than translate back to English with trained machine translation model. The third one is **hidden states interpolation** method. We design a pipeline for diffusion based model as shown in Figure 4. To conduct hidden states interpolation, we first random select two texts from captions corresponding to one same image. Then we get the two hidden states after going through the clip text encoder. And we sample θ from a given **beta distribution**, and use θ to conduct linear interpolation on the two hidden states. At last, the hidden states will be sent to the latent diffusion model as text guidance signal.

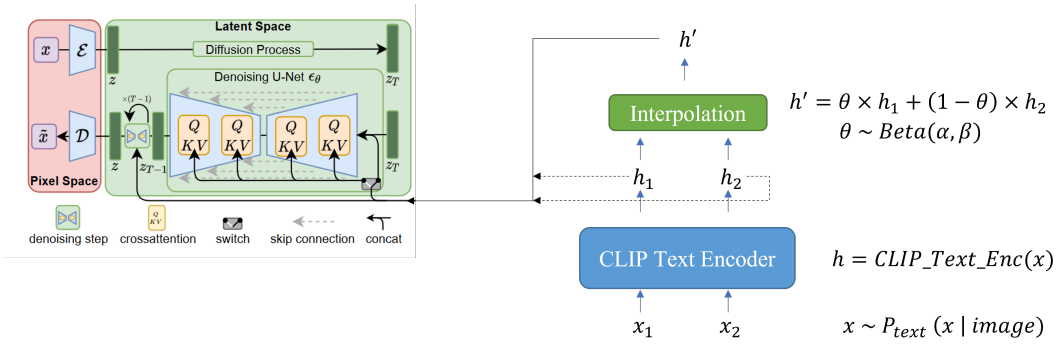

Figure 4: The pipeline of Hidden States Interpolation method in diffusion model.

3 Experiments

3.1 Diffusion Model's Structure

For researchers with low computational resources, doing experiments with diffusion model is extremely hard because of the notoriously high computational cost of reversing diffusion process. To use the diffusion model within acceptable running time, we follow this scheduler[3], by which we

can attain images with only 50 iterations, instead of 1000 iterations, without losing much quality of generated images.

Doing the reversing diffusion process in the pixel space is also computationally expensive, so we follow paper's[4] approach to learn a latent space which is equivalent to the original pixel space but with lower dimensions and do the diffusion process in the latent space. Besides, training a text2image diffusion model from scratch is extremely computationally expensive, so we start from a pre-trained diffusion model called *stable-diffusion-v1-4* .[2]

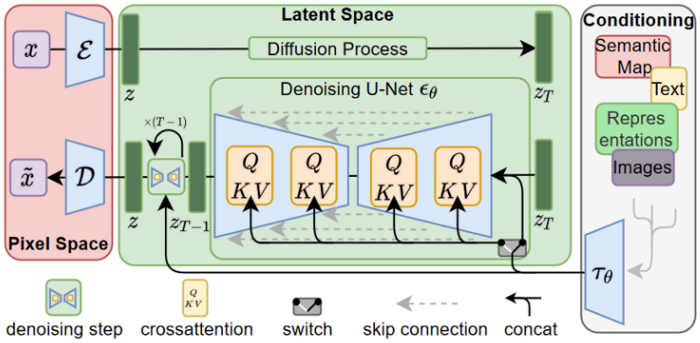

Figure 5: Model structure of text2image diffusion model, this figure is from [4]

The model architecture is shown in Figure 5. Firstly, the original image is projected from pixel space to the latent space by auto-encoder, and the diffusion process is finished in the latent space by U-Net. The text-conditioning information encoded by Clip is passed to the U-net by cross attention. Finally the auto-encoder will transform the image from latent space to pixel space to get a human-readable image.

3.2 Experiment Settings

We use a subset of MSCOCO 2017 as the dataset for both model training and evaluation. All experiments (training and inference) are conducted on a single NVIDIA A40 GPU. The hyper-parameters are shown in Table 1. We use open-source FID[3] metric to evaluate the overall generation quality.

Table 1: Hyper-parameters

Pretrained model	CompVis/stable-diffusion-v1-4
resolution	512*512
training batch size	8
number of training epochs	2
learning rate	1e-5
training seed	1
inference seed	0,1,2,3,4
training set size	50k
test set size	5k

3.3 Main Results

For the FID metric's result as shown in Table 2, we can see that First , finetuning on the coco dataset can decrease the FID scores. Second, Word manipulation and back-translation methods don't work leading to sub-optimal result. Last, our random hidden states interpolation method can further decrease the FID score suggesting a better generation quality.

[2]This trained diffusion model is available at: `https://huggingface.co/CompVis/stable-diffusion`
[3]https://github.com/mseitzer/pytorch-fid

Table 2: Result of FID scores for different data augmentation methods.

Model	FID ↓
Stable-Diffusion v1-4	49.9
Finetuned on COCO	47.3
+ Random Deleting	47.5
+ Back-Translation	47.7
+ Hidden States Interpolation ($\alpha = \beta = 1$)	**47.1**

3.4 Exploration On the Parameter for Beta Distribution

Since the hidden states interpolation method is the only method that can improve the generation quality, we conduct futher experiment its beta distribution. We further explore the hyper-parameter of the interpolation method. We choose beta distribution because it has a nice property that we can control how similar the augmented text is to the original text. As shown in the Figure 6 , bigger alpha and beta will lead to theta more close to 0.5, it will lead to a more different augmentation. A small alpha and beta will lead to a smaller theta , and generate more similar augmented data. The result in Table 3 shows that more similar augmentation is better.

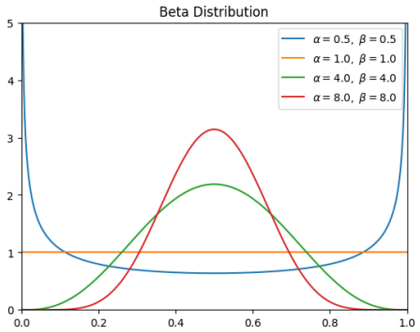

Figure 6: An example of beta distribution.

Table 3: Result of FID scores for different hyper-parameters of the interpolation method.

Model	FID ↓
Stable-Diffusion v1-4	49.9
Finetuned on COCO	47.3
+ Hidden States Interpolation ($\alpha = \beta = 1$)	**47.1**
+ Hidden States Interpolation ($\alpha = \beta = 4$)	47.4
+ Hidden States Interpolation ($\alpha = \beta = 8$)	47.6

4 Analysis on the Robustness of Diffusion Based Text2Image Model

This chapter introduces three self-created similarities based on the CLIP model, these self-created similarities can better match samples in the corpus, reflect the similarity between the image generated by the model and the original image or original text, and thus, to a certain extent, it can better reflect the stability of different models to generate images.

4.1 CLIP as Feature Extractor

We use CLIP as feature extractor for image and text. CLIP (Contrastive Language-Image Pre-training) is a pre-trained model for natural language processing and image recognition developed by OpenAI. It can learn language and image representations from large amounts of text and image data on the internet without the need for human-labeled data.

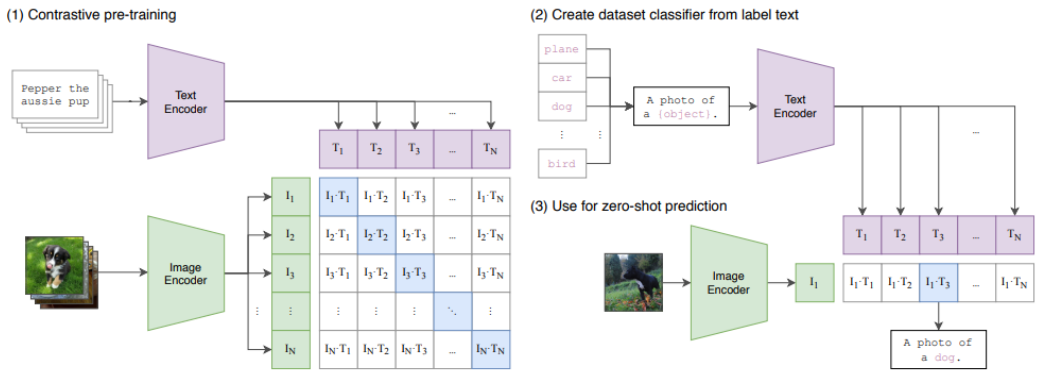

Figure 7: CLIP Model

As the Figure 7 shown, the core idea of CLIP is to use contrastive training to learn language and image representations. Contrastive training is a method of training a model by comparing the similarity between different samples, and CLIP uses this method to learn the relationship between language and images.

During the pre-training phase, CLIP learns language and image representations from a large amount of text and image data, and then these representations can be transferred to different tasks. In natural language processing tasks, CLIP can be used for tasks such as semantic analysis and text generation; in image recognition tasks, CLIP can be used for tasks such as image classification and object detection.

As a model pre-trained on 4,000,000,000 image-text pairs, it can learn a more robust and generalizable representation of language and visual information, and can be applied to downstream tasks without fine-tuning. Thus, we can say that the model can map an image or text to a hidden vector space, and the mapping may contain the content information of the original image or text.

So, in our project, we use the CLIP model to encode the images, including one group of original images, and other groups of images generated by different models. And then, we calculate the similarity between the groups of images generated by different models and the original images. The similarity is adapted from the cosine similarity.

4.2 Similarity1 : Image Similarity among Random Seeds

The first similarity is the image similarity among random seeds. Fixing texts, we compare the pairwise similarity between the images generated under five random seeds for each model.

Table 4: The Robustness to Random Seeds

Model	CLIP Similarity among 5 seeds↑
Stable-Diffusion v1-4	0.55
Finetuned on COCO 10k Images, 2 epochs	0.55
+Random Deleting	0.53
+Back-Translation	**0.58**
+Random Hidden States Interpolation ($\alpha = \beta = 1$)	0.55

And the results is in the Table 4. We can see that, the original coco model performs the same as stable-diffusion model, but after using back-translation, it has a 3 percent increase. So the back-translation can promote the model's robustness to random seeds.

4.3 Similarity2 : within Similar Texts

The second similarity is similarity within similar texts.

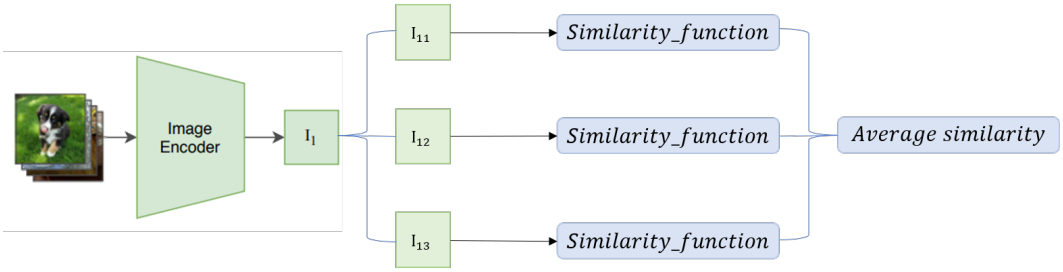

Figure 8: Similarity2 : within Similar Texts

As shown in Figure 8, in this calculation, we first encode each image by CLIP, after that, we calculate the similarities in each small group classified by texts (a small group contains 5 images generated by 5 similar texts), and for each two images in this small group we calculate the average similarity. And then we calculate the average similarities for each big group classified by model.

Table 5: The Robustness to Similar Texts

Model	Average Similarity	Max Similarity	Min Similarity	Std of Sim
Stable-Diffusion v1-4	**86.23**	95.31	56.2	6.68
Finetuned on COCO 10k Images, 2 epochs	85.5	95.61	58.59	6.71
+Random Deleting	85.25	95.36	59.91	6.42
+Back-Translation	85.94	**95.90**	58.89	6.48
+Random Hidden States Interpolation ($\alpha = \beta = 1$)	85.64	94.92	**60.11**	**6.20**

And the results is in the Table 5. We can see that, the adjustment on the coco model all have improved compared with coco model, but the fine tune on coco leads a performance reduction compared with baseline model.

We can also see that in general, the value of Similarity 2 is much higher than that of Similarity 1, indicating that the influence brought by random seeds is greater than that brought by different similar texts.

4.4 Similarity3 : Faithfulness between Image and Text

The third similarity is similarity between image and text, and we think this can reflect the faithfulness for the generating models to the original texts.

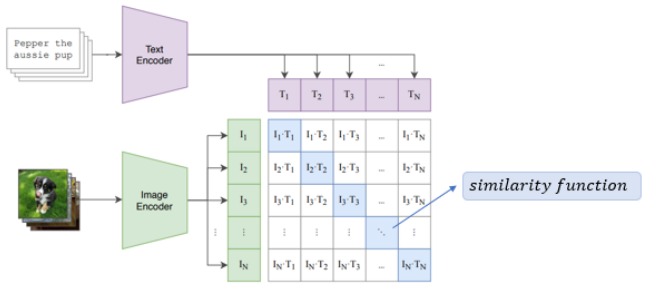

Figure 9: Similarity3 : Faithfulness between Image and Text

As shown in Figure 9, we first encode the images and texts by CLIP model, and then we calculate the average similarity for each group of images and the texts. Cause the images is generated by model giving the texts, so this similarity can reflect the faithfulness of the generated image to the original text.

And the results is in the Table 6. We can see that still the coco model with back translation performs best.

Table 6: Faithfulness between Image and Text

Model	Similarity	Relative Similarity
Original Images	15.15	100
Stable-Diffusion v1-4	14.15	93.4
Finetuned on COCO 10k Images, 2 epochs	14.66	96.77
+Back-Translation	**14.70**	**97.03**
+Random Detection	14.69	96.96
+Random Hidden States Interpolation ($\alpha = \beta = 1$)	14.66	96.77

5 Conclusion and Future Work

In conclusion, our work and contributions are as follows:

1. We found that Hidden States Interpolation Method leads to improvement to the overall quality of text2image model. While the other traditional text augmentation methods fail on that.
2. The randomness from diffusion process has a bigger effect than that from similar texts.
3. Back-Translation can best improve the robustness and faithfulness of the model.

Actually, the difference between images generated by different texts with similar meanings may come from two reasons:

1. The diversity of diffusion model itself: similar semantic meanings not necessary lead to the same image, and images with great difference may also fit the same semantic meaning.
2. The diffusion model is not robust to different texts: for some texts the model can perform well, but for other texts the model may generate unfaithful images.

One limitation of this work is that the evaluation of robustness might raise disagreement that how can we differentiate between diversity and robustness. Whether the low CLIP similarity comes from the diversity or unfaithfulness is worth more exploration. Inspired by the GAN model, Using a discriminator to identify the unfaithfulness or training the diffusion model and discriminator jointly may be a possible approach.

The text augmentation methods we use is relatively naive, more effective augmentation can be explore to improve the robustness of text2image diffusion model.

6 Work Allocation

Liang Chen Challenge analysis, implementation of text data augmentation methods and ablation on hidden states interpolation method.

Zhe Yang Training our text2image diffusion model with augmented texts and generating images with our model.

Zheng Li Proposing different image and text coding and similarity calculation methods to measure the robustness of model generation.

References

[1] 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. In *International Conference on Machine Learning*, 2021.

[2] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS*, 2017.

[3] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.

[4] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.