RBE 474X
Project 4

Colin Balfour
Worcester Polytechnic Institute
Worcester, Massachussetts 01609

Khang Luu
Worcester Polytechnic Institute
Worcester, Massachussetts 01609

Thinh Nguyen
Worcester Polytechnic Institute
Worcester, Massachussetts 01609

I. Part 1 - Diffusion Training

We trained the diffusion model to generate new images from a provided dataset. This training is done over linear betas. Below are the hyperparameters used for training:

- Batch size: 80
- T: 500 (number of steps)
- Channel: 128
- Channel-multiplier: [1, 2, 2, 2]
- Learning rate: 1e-4
- Beta 1: 1e-4
- Scheduling: linear
- Beta T: 0.028
- Loss Function: MSE loss

For training, we are using the CIFAR10 dataset.

Here is the link for the model checkpoint:

https://wpi0-my.sharepoint.com/:u:/g/personal/knluu_wpi_edu/EXPcDEfuu5tPn4DGihGNaMYBHhFu96k89cUHJEek6a-ILQe=zOgmxp

Figure 1 is the loss plot of the training:

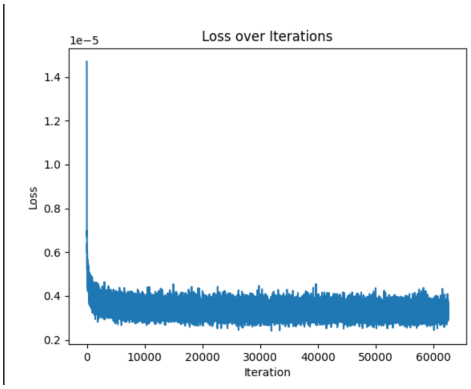


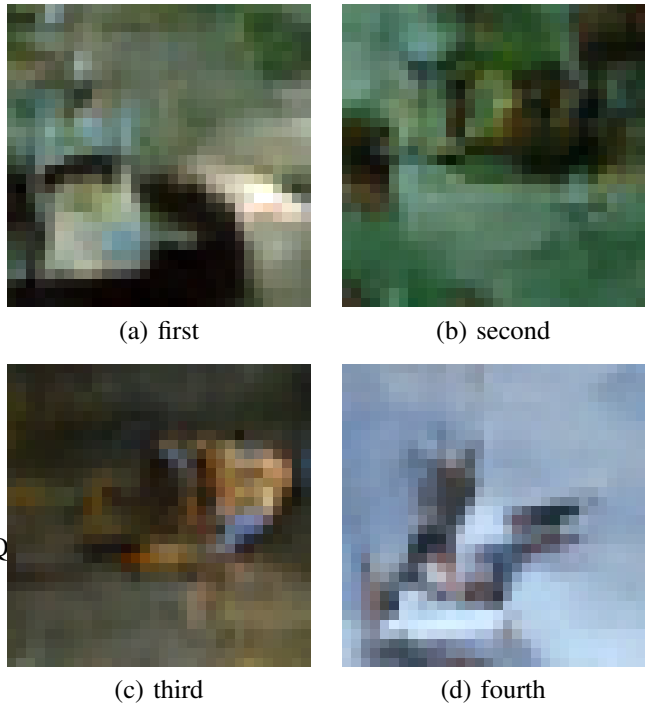
Fig. 1. Plot

Figure 2 to 11 show the images for every class generated by the diffusion model.

II. Part 2 - Data augmentation using diffusion

For this section, we are augmenting the cifar10 dataset with 5000 extra images per class (examples are shown in Figure 2 to 11).

The network used for this is a custom implementation of MLP. The hyperparameters used for training are:



(a) first (b) second

(c) third (d) fourth

Fig. 2. Bird images, generated from the diffusion model

- Learning rate: 1e-4
- Epoch count: 40
- Batch size: 32

A. Results for non-augmented data

The accuracy for the non-augmented data is 44.05%. Figure 12 is the loss plot for the training.

And the confusion matrix for the training is shown in Figure 13.

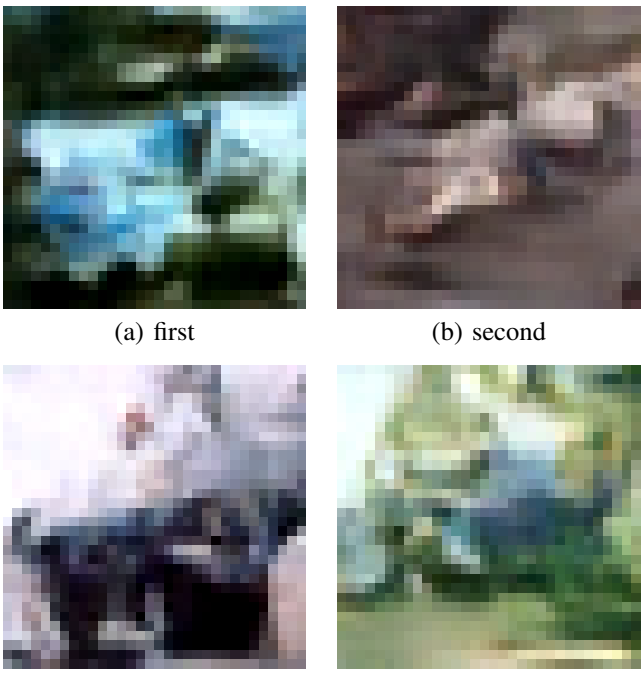
B. Results for augmented data

The accuracy for the augmented data is 41.14%. Figure 14 is the loss plot for the training.

And the confusion matrix for the training is shown in Figure 15.

C. Comparison

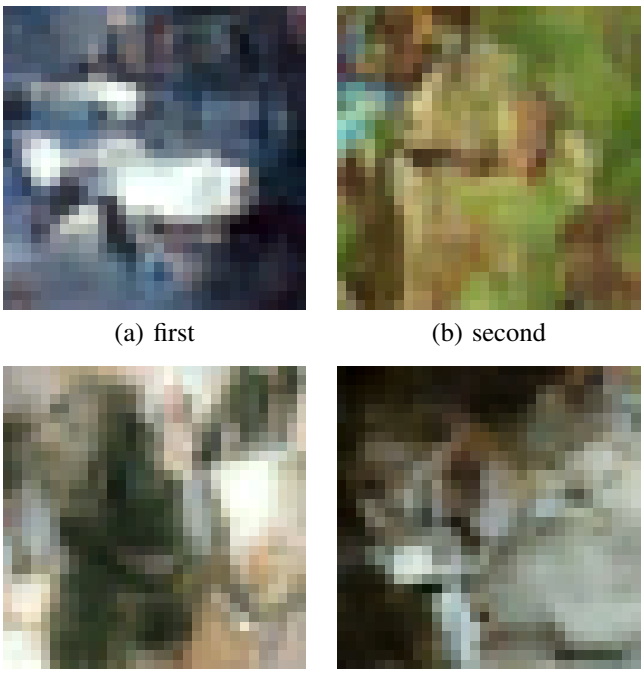
The accuracy for the non-augmented data is 44.05%, while the accuracy for the augmented data is 41.14%. Although the



(a) first (b) second

(c) third (d) fourth

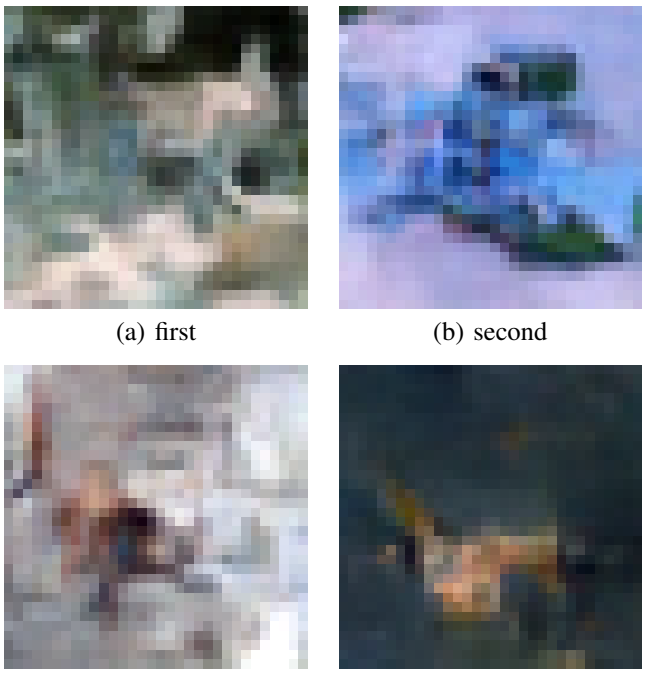
Fig. 3. Car images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

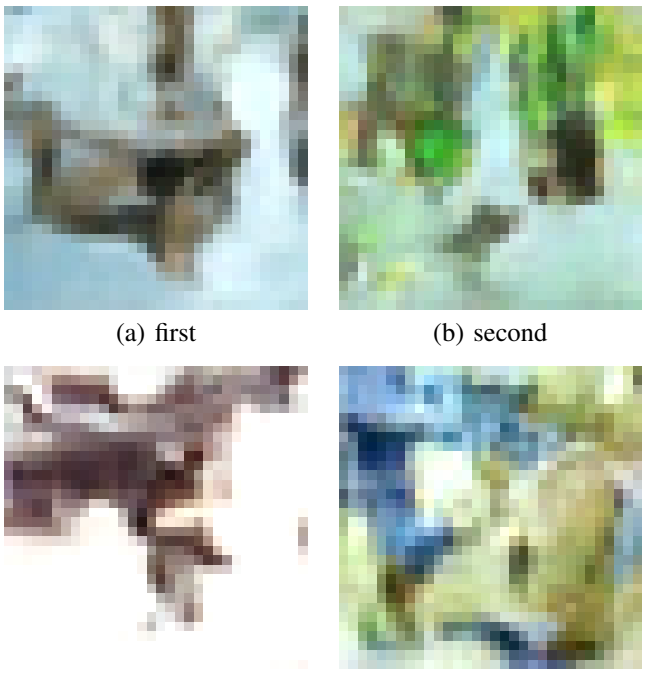
Fig. 4. Cat images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

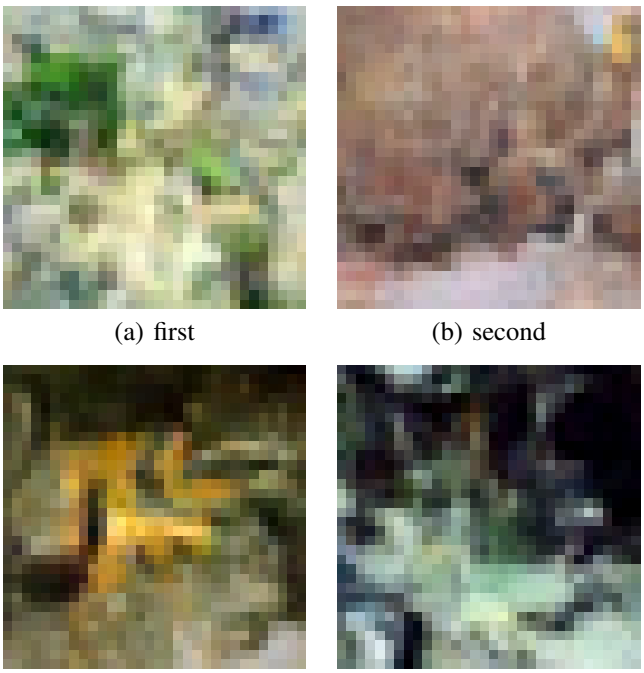
Fig. 5. Deer images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

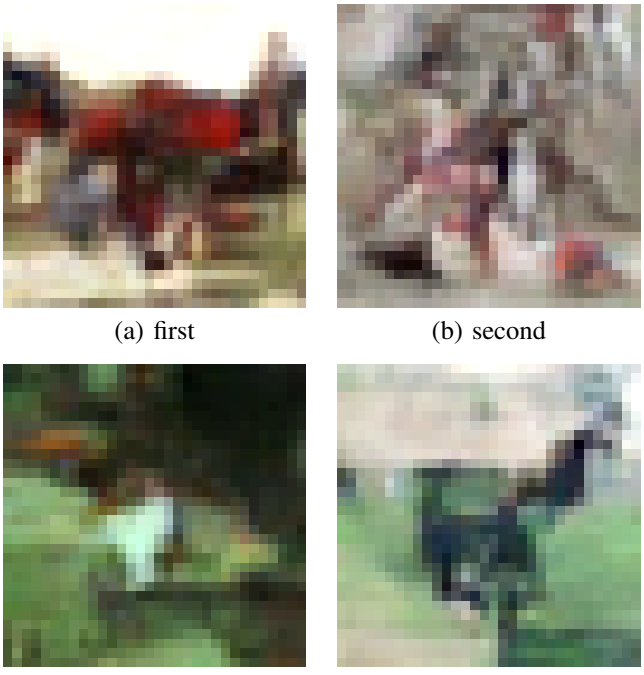
Fig. 6. Dog images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

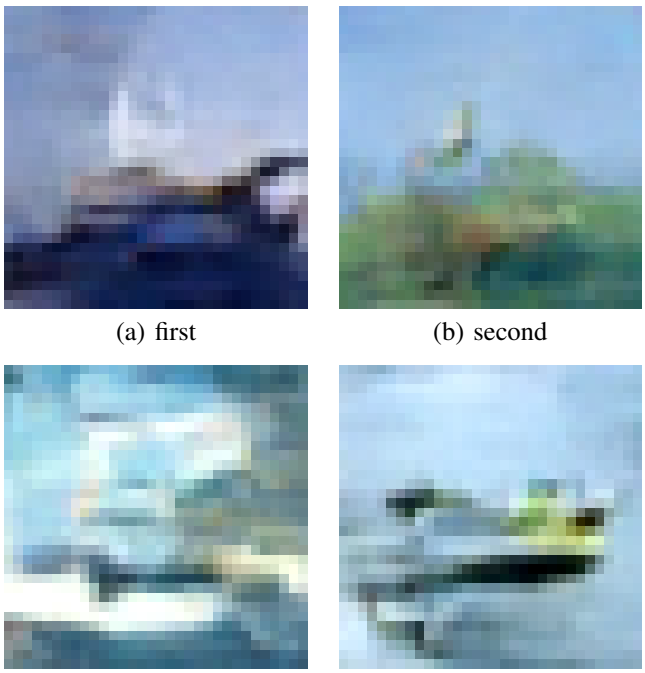
Fig. 7. Frog images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

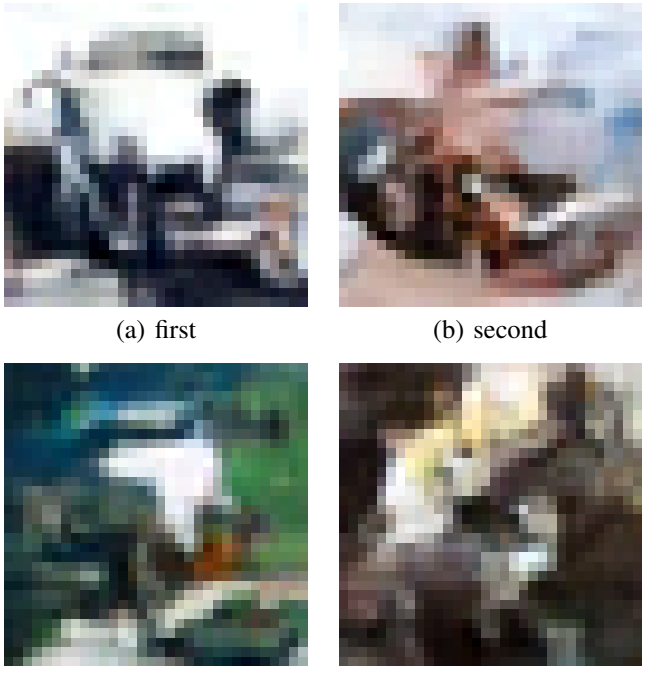
Fig. 8. Horse images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

Fig. 9. Ship images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

Fig. 10. Truck images, generated from the diffusion model

(a) first (b) second

(c) third (d) fourth

Fig. 11. Plane images, generated from the diffusion model

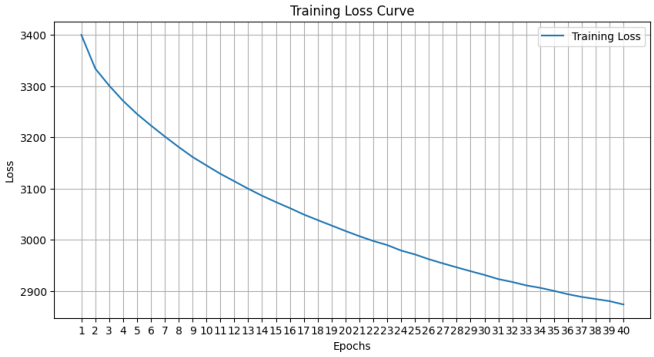


Fig. 12. Non-augmented data, loss plot

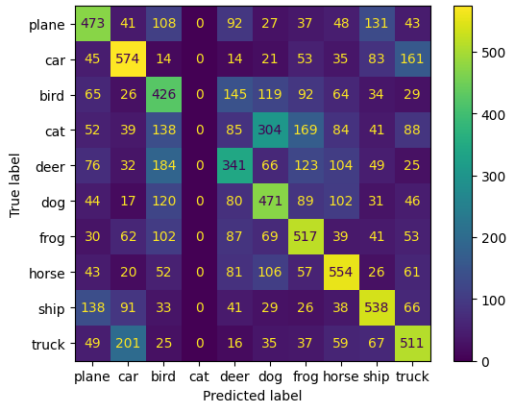


Fig. 13. Non-augmented data, confusion matrix

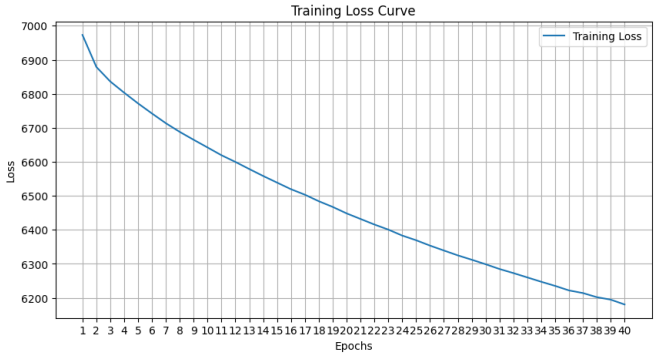


Fig. 14. Augmented data, loss plot

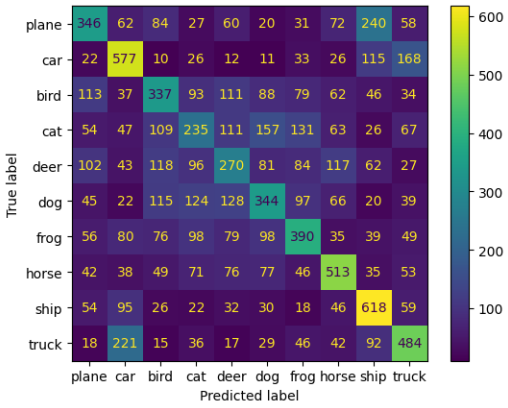


Fig. 15. Augmented data, confusion matrix

augmented data includes additional images generated by the diffusion model, it does not improve the accuracy. This is likely because the generated images are somewhere between random noise and real data, resulting in a training process that is less effective. The network takes longer to train, and the loss is significantly higher for the augmented data, almost double that of the non-augmented data.

This higher loss suggests that the model is not effectively learning from the augmented dataset and may even be forming incorrect patterns, which explains why the accuracy remains similar but doesn't improve with augmentation. In this case, the diffusion model's generated data does not contribute meaningful new information for the model to learn from, and may be decreasing performance due to the network learning to model the noise of the diffusion images.

The model essentially learns to classify the noise generated by the diffusion model, which causes overfitting and a decrease in performance. The model is not able to generalize well to the real data, as the noise generated by the diffusion model is not representative of the real data. This is why the accuracy is lower for the augmented data compared to the non-augmented data.

III. Part 3 - Cosine Scheduling Training

The same training as in Part 1 is done, but with cosine scheduling. Below are the hyperparameters used for training:

- Batch size: 80
- T: 500 (number of steps)
- Channel: 128
- Channel-multiplier: [1, 2, 2, 2]
- Learning rate: 1e-4
- Beta 1: 1e-4
- Scheduling: cosine
- Beta T: 0.028
- Loss Function: MSE loss

Here is the link for the model checkpoint:
https://wpi0-my.sharepoint.com/:u:/g/personal/knluu_wpi_edu/EWTdnweuDa5Et6ImdWz_3zkBO-7_RaapQ23v73cWzs33IQ?e=ew56Vr

Figure 1 is the loss plot of the training:

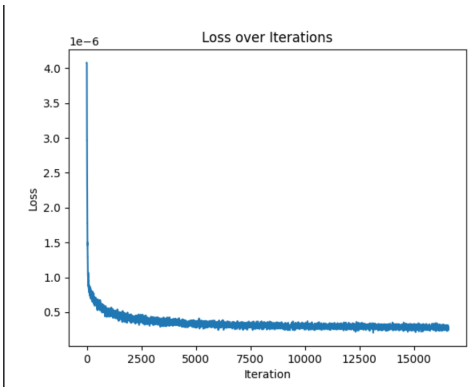


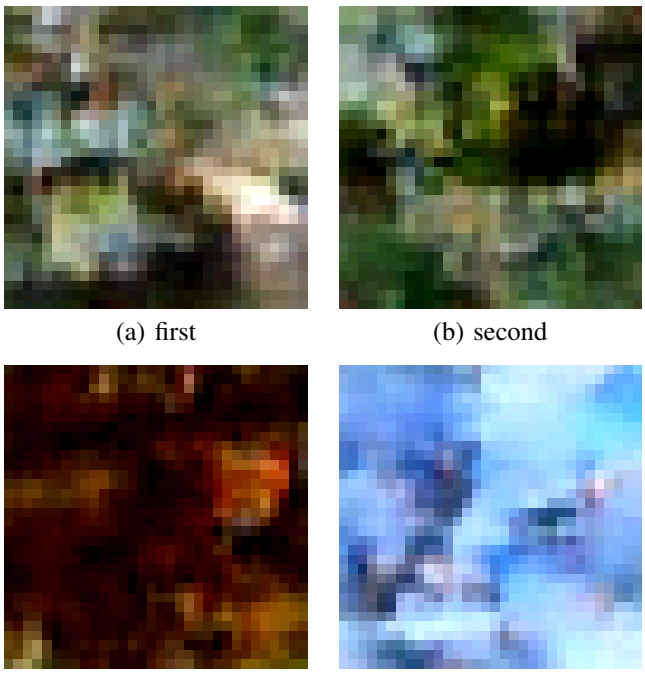
Fig. 16. Plot

Figure 2 to 11 show the images for every class generated by the diffusion model.

The difference between cosine and linear scheduling lies in how the learning rate decays over time. In linear scheduling, the learning rate decays at a constant rate throughout training, whereas in cosine scheduling, the learning rate decreases rapidly at first and then slows down, following a cosine curve. This allows the model to explore more during the early stages of training and then gradually refine its learning.

Loss and Performance Comparison The loss for cosine scheduling follows a similar trend as the linear schedule in Part 1, but with slight variations. The early stages of training using cosine scheduling see a sharper decline in loss, indicating that the model is adjusting more rapidly at the beginning. However, as training progresses, the rate of improvement in loss slows down, eventually reaching values comparable to those achieved with linear scheduling. This behavior is visible in the training loss plot (Figure 16).

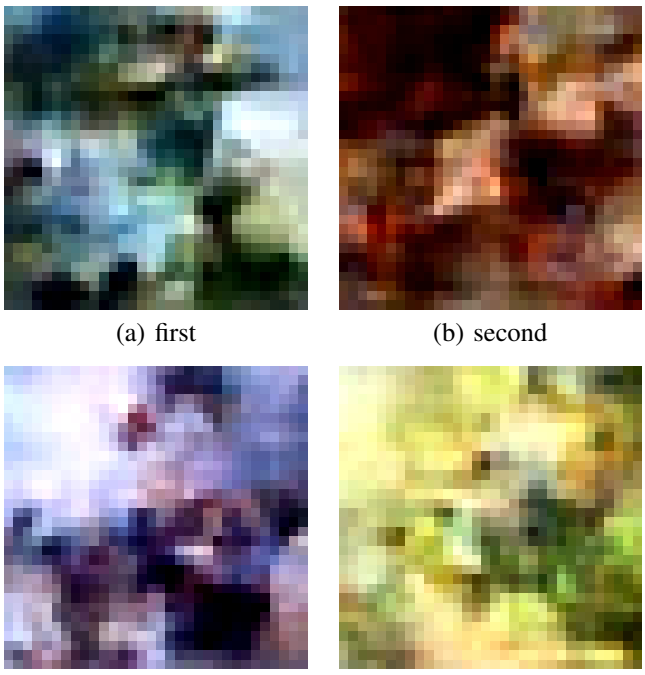
Although the final loss values are similar, the training process with cosine scheduling produces smoother and potentially more stable results in the long run. The cosine decay helps the model avoid overly aggressive updates toward



(a) first (b) second

(c) third (d) fourth

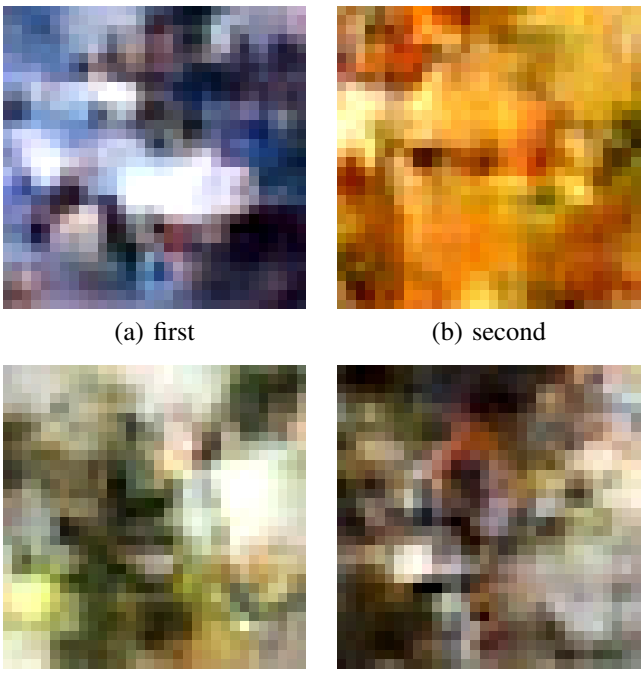
Fig. 17. Bird images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

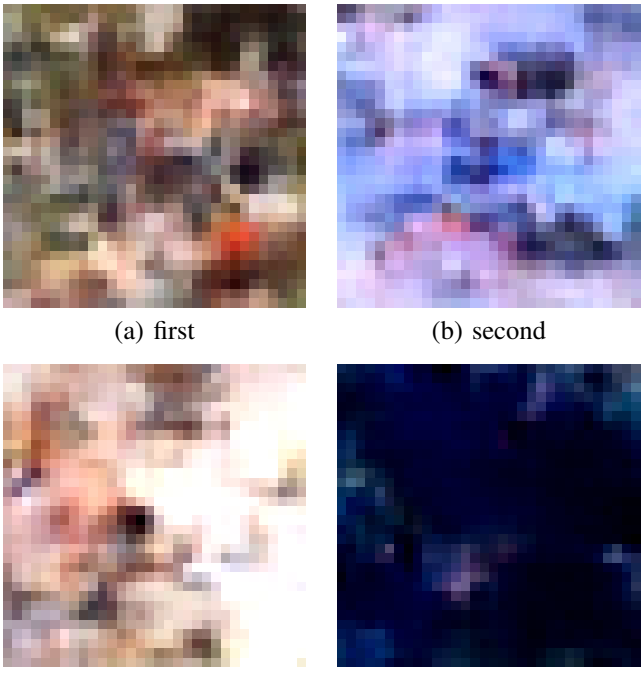
Fig. 18. Car images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

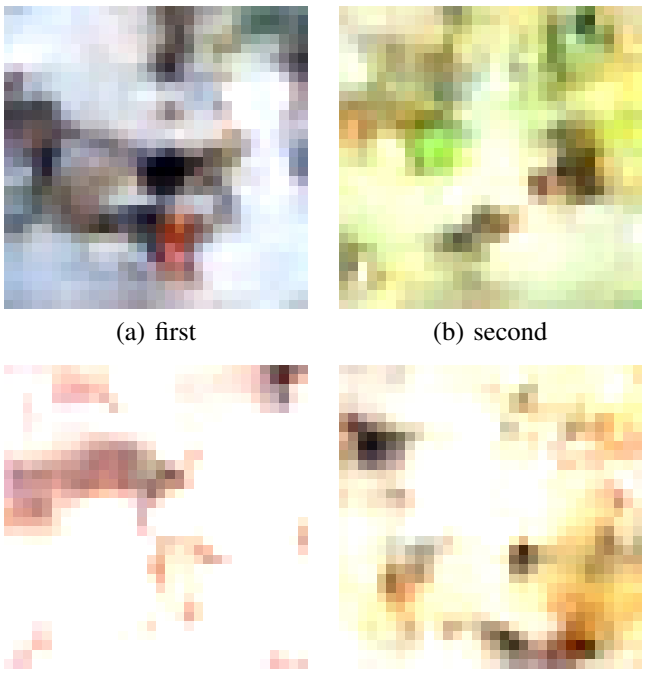
Fig. 19. Cat images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

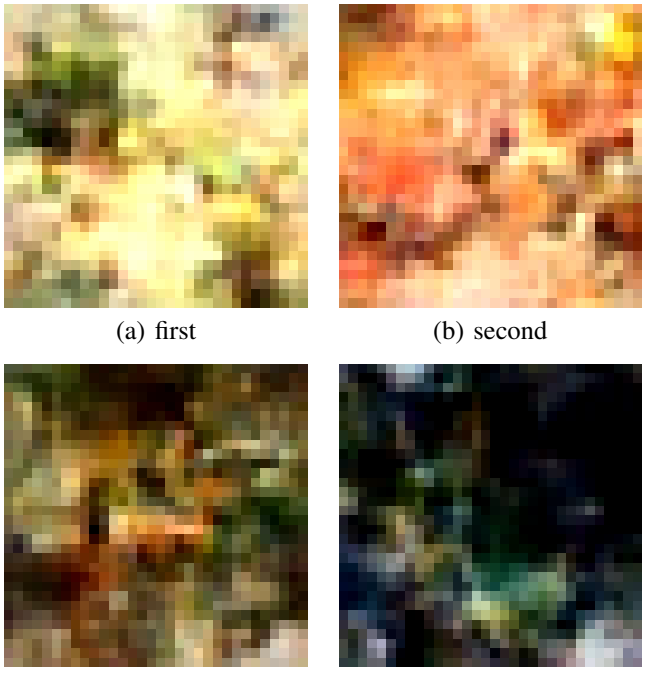
Fig. 20. Deer images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

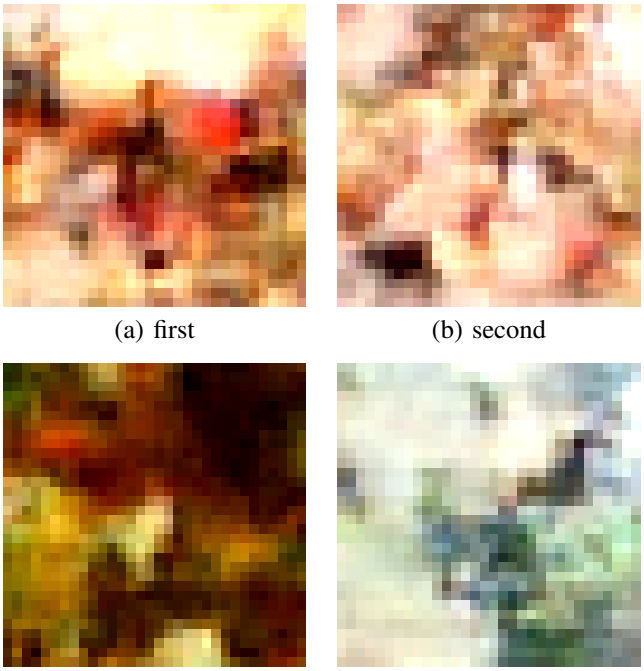
Fig. 21. Dog images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

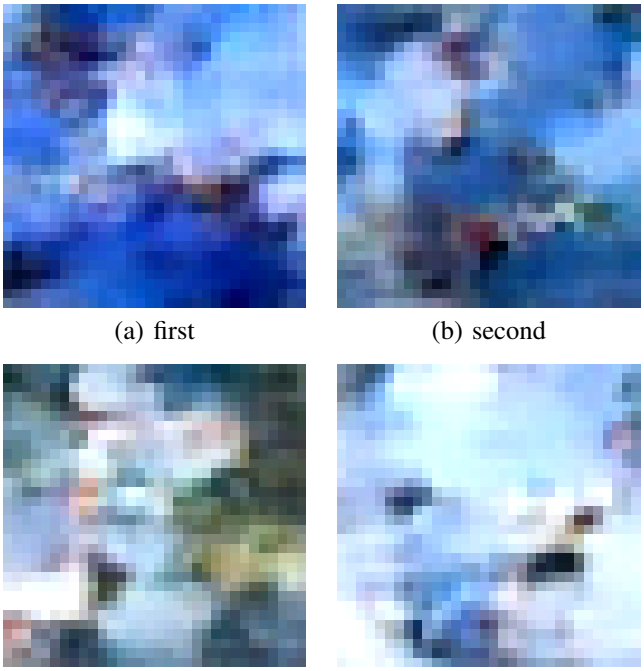
Fig. 22. Frog images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

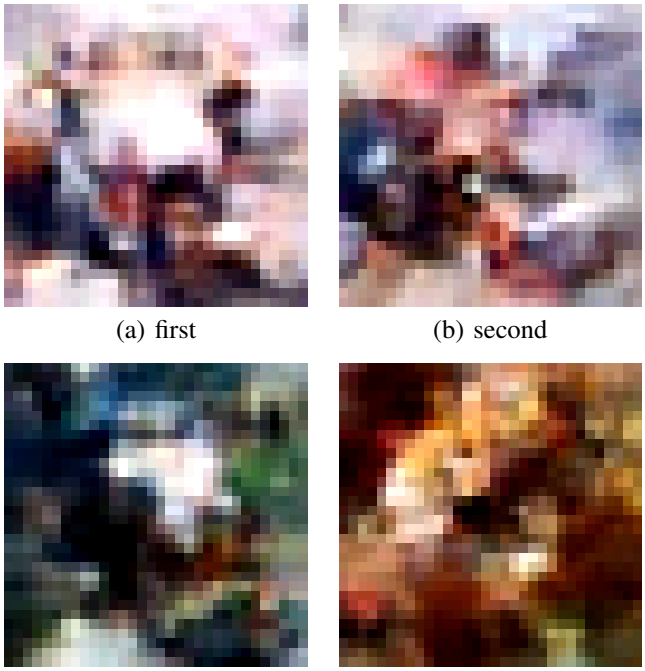
Fig. 23. Horse images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

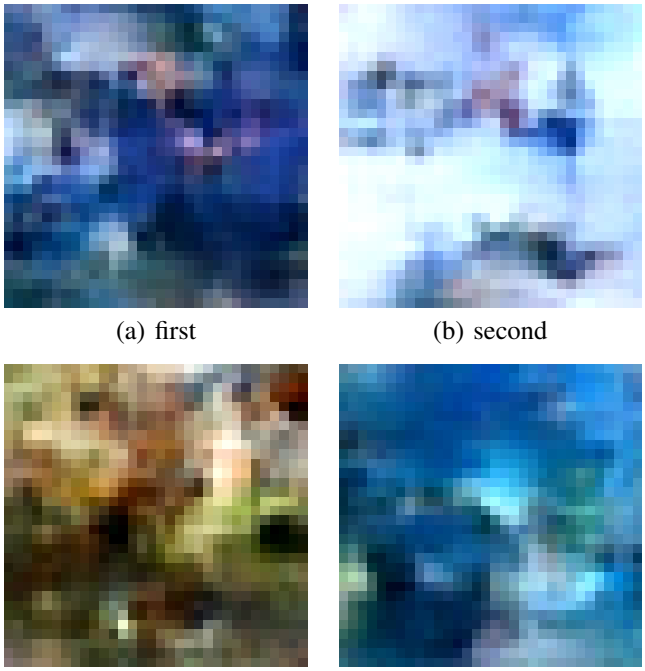
Fig. 24. Ship images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

Fig. 25. Truck images, generated from the diffusion model



(a) first (b) second

(c) third (d) fourth

Fig. 26. Plane images, generated from the diffusion model

the end of training, which can prevent it from getting stuck in local minima or overfitting to noise.

A. Similarity in results

Despite these differences in how the learning rate decays, the overall performance in terms of accuracy and quality of the generated images remains quite similar to that observed with linear scheduling. This is expected because both linear and cosine approaches effectively balance exploration and refinement during training, though they do so at different paces.

The similar results suggest that for this particular dataset and model configuration, the choice between cosine and linear scheduling may not drastically affect the final outcome. However, cosine scheduling offers the advantage of more nuanced control over the learning process, especially during the later stages of training.

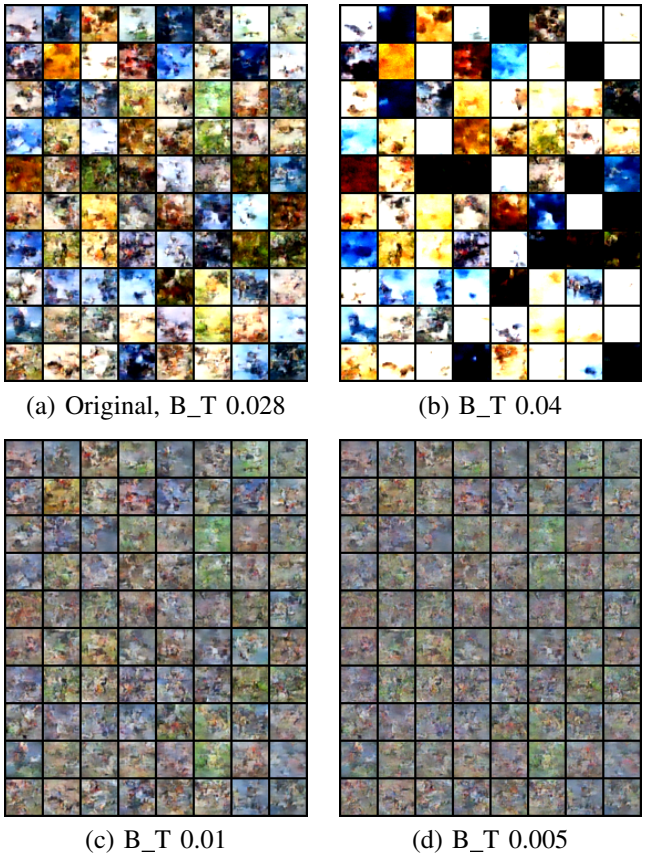
B. Parameter tuning

The most influencial hyperparameters for the diffusion model's evaluation are beta 1, beta T, and img size. Beta 1 and beta T are the parameters that control the noise level of the generated images. The img size parameter is the size of the images generated by the diffusion model.

The img size parameter is important because it determines the resolution of the generated images, and also effect the image generated to be much different than the generation of different resolutions.

If beta 1 is raised too high, or beta T is raised too high, the generated images will be washed out. Lower beta 1 values seem to not affect much in our testing.

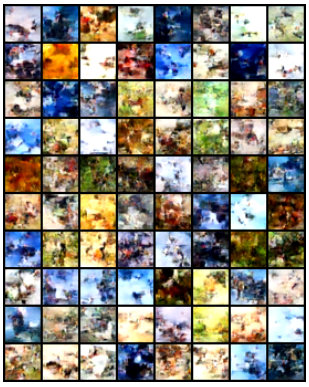
If beta T is lowered to close to beta 1, the generated images will be too noisy.



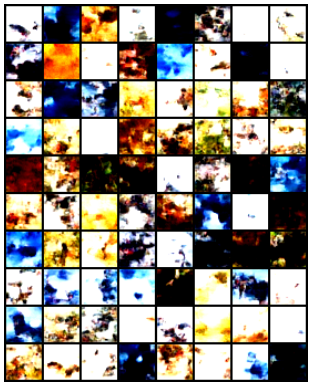
(a) Original, B_T 0.028 (b) B_T 0.04

(c) B_T 0.01 (d) B_T 0.005

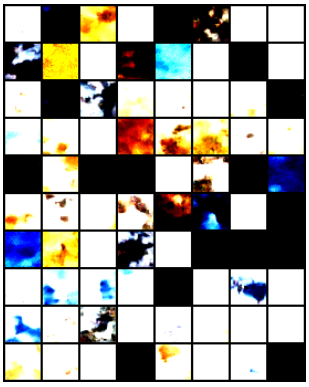
Fig. 27. Diffusion outputs with different beta T values



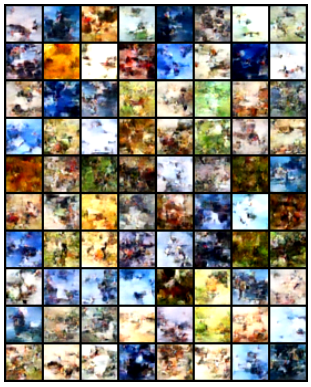
(a) Original, B_1 1e-4



(b) B_T 0.04

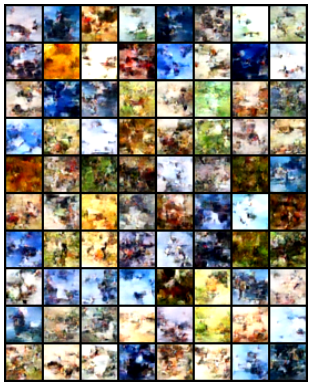


(c) B_1 1e-2, B_T 0.004

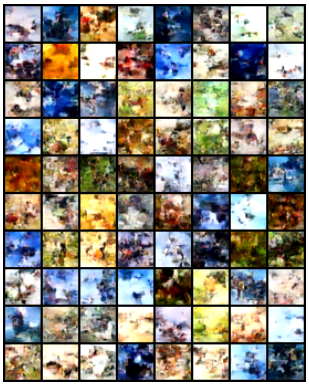


(d) B_1 1e-6

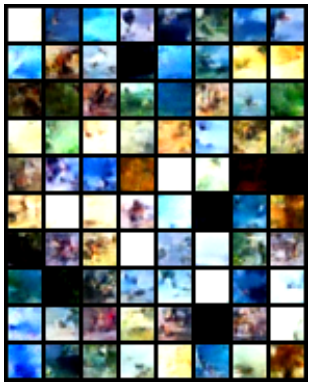
Fig. 28. Diffusion outputs with different beta 1 values



(a) W 0.18



(b) W 18



(c) Image size 16

Fig. 29. Diffusion outputs, other hyperparameters