University of California San Diego

CSE 252D: Advanced Computer Vision
Spring 2023

A Two-Phase Training Approach To Boost NeRF Reconstruction Speed

Author	*Email*
Krish Rewanth Sevuga Perumal	`ksevugaperumal@ucsd.edu`
Manas Sharma	`m7sharma@ucsd.edu`
Ritika Kishore Kumar	`rkishorekumar@ucsd.edu`
Sanidhya Singal	`ssingal@ucsd.edu`

1 Problem Statement

NeRF, or Neural Radiance Fields, have brought a significant breakthrough in the field of 3D reconstruction and novel view synthesis. They achieved state-of-the-art visual quality, producing impressive demonstrations and inspiring many subsequent works. However, the current training process of NeRF is prohibitively slow, hindering its practicality and scalability in real-world applications. This extensive training time required prevents rapid experimentation, model refinement, and deployment in time-sensitive scenarios. Therefore, in this project, we propose a novel method to improve training time of NeRF.

2 Method Description

The training duration for a high-definition NeRF model can stretch across hours. To tackle this obstacle, we present a pioneering strategy inspired by transfer learning. Our approach entails the utilization of a pre-trained NeRF model, trained on one or more generic objects to subsequently train the object of interest. We call the former as *pretext training* and the latter as *downstream training*.

Our approach revolves around the fundamental idea of enabling the pre-trained NeRF to acquire general features and representations pertinent to one or more object categories, thereby diminishing the number of epochs required for training the new object from scratch. As a result, we achieve a significant reduction in both time and computational resources. By capitalizing on the knowledge and acquired features from the pre-trained NeRF model, we expedite the training process and facilitate the generation of a downstream 3D object.

2.1 Datasets and Metrics

We use the ShapeNet dataset, which is a richly annotated, large-scale dataset of 3D shapes. Each data file in this dataset contains 800 views of a given shape. These shapes are divided into several categories, such as vehicles, furnishing, and natural object.

We evaluate our model on the Peak Signal to Noise Ratio (PSNR), a widely used metric which measures the ratio between the maximum possible pixel intensity and the mean squared error. We aim for higher PSNR values, which indicate better image quality and less distortion.

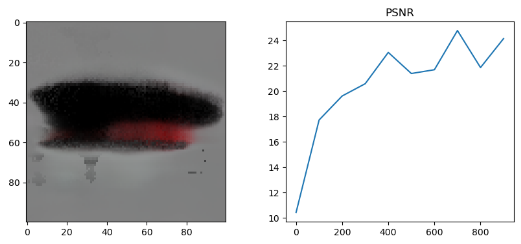
2.2 Implementation Details

The aim of our project is to obtain a transferable weights model (from pretext training) which can be applied to scene-specific downstream training to obtain novel views of an object of interest. Due to computational constraints, we use a Tiny-NeRF model instead of the baseline NeRF model. The first MLP in the Tiny-NeRF model contains layers of sizes 39 (input), 256 (hidden), and 257 (output) respectively. The second MLP also has 3 layers but with sizes of 295 (input), 256 (hidden), and 3 (output) respectively. Only the output layer of second MLP has the Sigmoid activation, all other layers have the ReLU activation function. Overall, the Tiny-NeRF model is much lighter and faster than the vanilla NeRF model. For instance, training this model for 5000 epochs on a single Nvidia GTX 1080 GPU takes about 15 minutes. Here, we train on 800 images of an object with each image of size 100×100.

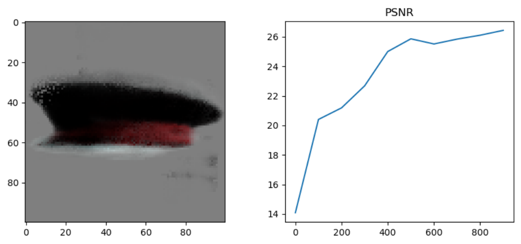
We modify this Tiny-NeRF model for our purpose of training a Two-Phase NeRF. Please refer to our implementation here: `https://github.com/sayhitosandy/Two_Phase_NeRF`.

Essentially, we train the Tiny-NeRF twice, once for pretext training and once for downstream training respectively. First, we train the pretext model using the Adam optimizer and MSE loss. We set up the training data which consists of one or more .npz files. Each .npz file contains 800 images of size 100×100 for an object/shape present in the ShapeNet dataset, along with the (x, y, z, θ, ϕ) values. The images help in the extraction of the (r, g, b, σ) values. We train the model for 5000 epochs, saving the pretext model with its weights after every 1000 epochs.

Next, we train the downstream model for 2000 epochs, again using the Adam optimizer and MSE loss. However, we do not train the model from scratch; instead, we use the saved pretext model to further train on another object/shape from the ShapeNet dataset. Again, after every 100 epochs, we use the downstream model to generate novel views of the object of interest, along with the PSNR values. In the following section, we show that by using this approach, we achieve the same PSNR values with much lower number of training epochs than the baseline NeRF.

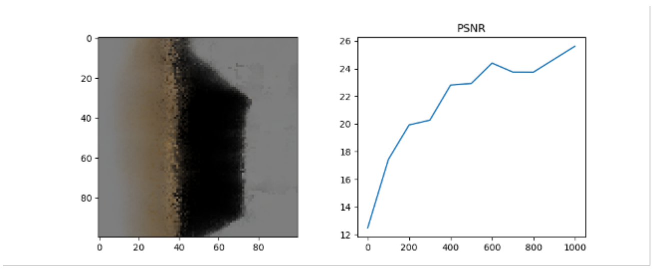


(a) Baseline NeRF

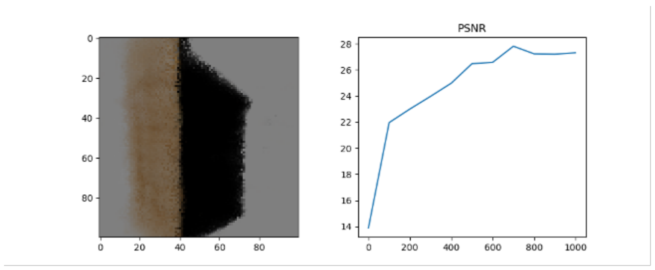


(b) Two-Phase NeRF

Figure 1: Comparison of Image Reconstruction with NeRF trained on objects of same categories



(a) Baseline NeRF



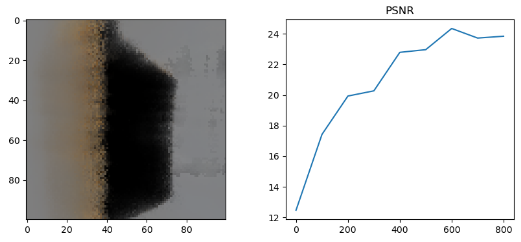
(b) Two-Phase NeRF

Figure 2: Comparison of Image Reconstruction with NeRFs trained on objects of different categories

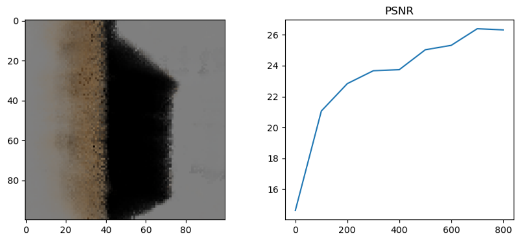
3 Results

We perform several experiments on the ShapeNet dataset to evaluate our proposed model. These experiments are broadly classified into the following categories:

1. **Pretext training on one object and downstream training on one unseen object of the same category**: We trained a pretext model on a type of cap/hat and applied it to generate views of another type of cap/hat. We observed a substantial reduction in the number of epochs required for training. Figure 1 compares the two models.
2. **Pretext training on one object, and downstream training on one unseen object of a different category than the first object**: We trained a pretext model on multiple scenes of a barn, and utilized it as a starting point for generating 3D views of a wooden box, which belong to completely different categories. We observed a reduction in the number of epochs required for training in this scenario as well, as shown in figure 2. Refer to the appendix (§6) for difference in reconstruction.
3. **Pretext training on one or more objects of different categories, and downstream training on one unseen object of a different category**: In order to generalize the pre-trained NeRF model, we experimented with two different approaches. Firstly, we trained the NeRF model on two different objects by randomly sampling images of each object during pretext training. However, this approach resulted in a noisy output because combining scenes of two different objects simultaneously did not make sense. Consequently, the performance of this approach was poorer than Case 1, as the starting scene for downstream training was filled with noise as shown in the appendix (§6). We depict the final results for this approach in figure 3.



(a) Baseline NeRF



(b) Two-Phase NeRF

Figure 3: Comparison of Image Reconstruction with NeRFs trained on objects of different categories with random sampling

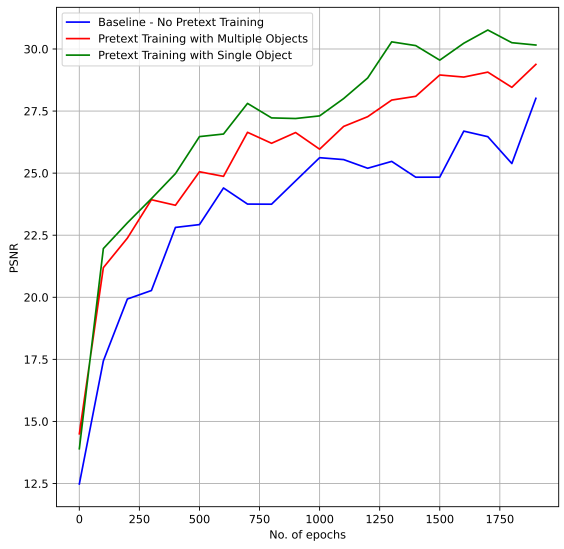


Figure 4: Comparison of PSNR values for the 3 types of experiments

Secondly, we attempted training of the two objects in a sequential order and removed the random sampling. This approach, however, yielded results similar to Case 1 and did not provide any significant advantages. In this case, the second view dominated the training process, causing us to lose most of the information from the first view.

4 Conclusion

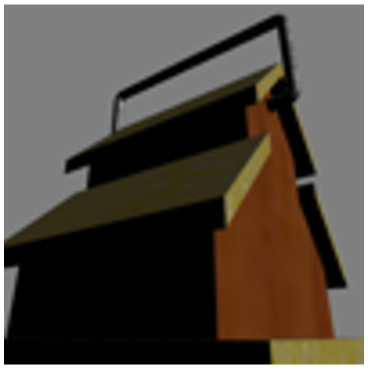
Figure 4 illustrates the relationship between the number of epochs and the PSNR (Peak Signal to Noise Ratio) for the aforementioned scenarios. We observe that utilizing a pretext model trained on different categories significantly reduces the number of epochs required to achieve a good PSNR score. Consequently, this approach saves a significant amount of training time. When comparing the number of iterations required to achieve a PSNR value of 27.5, our model reached that threshold in approximately 700 iterations, whereas the baseline model required around 1900 iterations. This resulted in a significant increase in speed, with our model being approximately **2.7**× faster than the baseline model.

The primary objective of our project was to minimize the training time required to train an object of interest by leveraging the structural knowledge learned from pretext training. If time is not a constraint, training NeRF on a single view without any pretext training would yield a much more accurate representation of objects. Nevertheless, this would necessitate a considerably longer training duration.

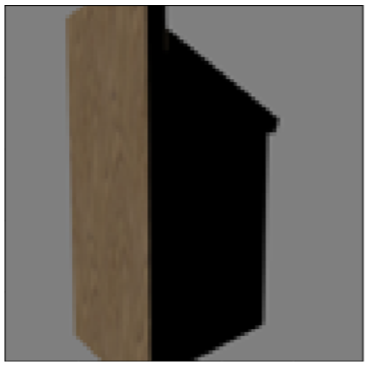
5 Contributions

The project was executed with a strong sense of collaboration and shared responsibility, and each team member actively contributed to every task. The following breakdown outlines the primary responsibilities undertaken by each team member:

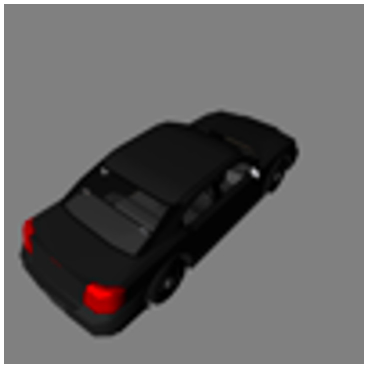
1. **Krish Rewanth Sevuga Perumal**: Created an end-to-end pipeline for training the model, and reviewed the literature.
2. **Manas Sharma**: Added configurations to run different experiments, interpreted the results, reformatted the code, and reviewed the literature.
3. **Ritika Kishore Kumar**: Developed an initial proof-of-concept, modularized the code, and reviewed the literature.
4. **Sanidhya Singal**: Pre-processed the input data, built utility functions, ran different experiments, and reviewed the literature.



(a) Barn



(b) Cardboard Box



(c) Car

Figure 5: Views of objects of different categories

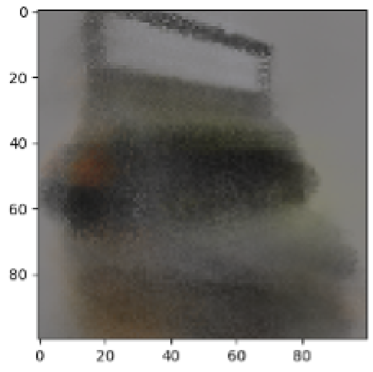


Figure 6: Noise due to random initialization

6 Appendix

Here, we analyze the training process of Baseline NeRF and Two-Phase NeRF iteratively to compare their performance and observe the results of noisy images generated as part of pretext training with random sampling.

The Baseline NeRF model is trained without any prior knowledge of specific views, while the Two-Phase NeRF model undergoes a pre-training phase using a view of a barn (figure 5a). Both NeRF models are then tasked with reconstructing the view of a cardboard box (figure 5b) during the training process.

The key concept behind the faster training of the Two-Phase NeRF approach lies in the fact that the pre-training phase equips the model with knowledge of smooth shapes and an understanding of common geometry. As a result, the Two-Phase NeRF model can quickly adapt to new views as compared to the Baseline NeRF, which lacks prior understanding of different viewpoints and starts the reconstruction process from random initialization, as we can see from figure 7.

Another observation is that using multiple images during pretext training with random sampling generates a highly noisy output, as shown in the figure 6. This output, which serves as the starting point for new training, contains a lot of noise and lacks smoothness. We can see remnants of the barn (figure 5a) and car (figure 5c), on which it was initially trained, in this output. While it outperforms the Baseline NeRF, it does not perform as well as the Two-Phase NeRF trained on a single object, due to significant noise.

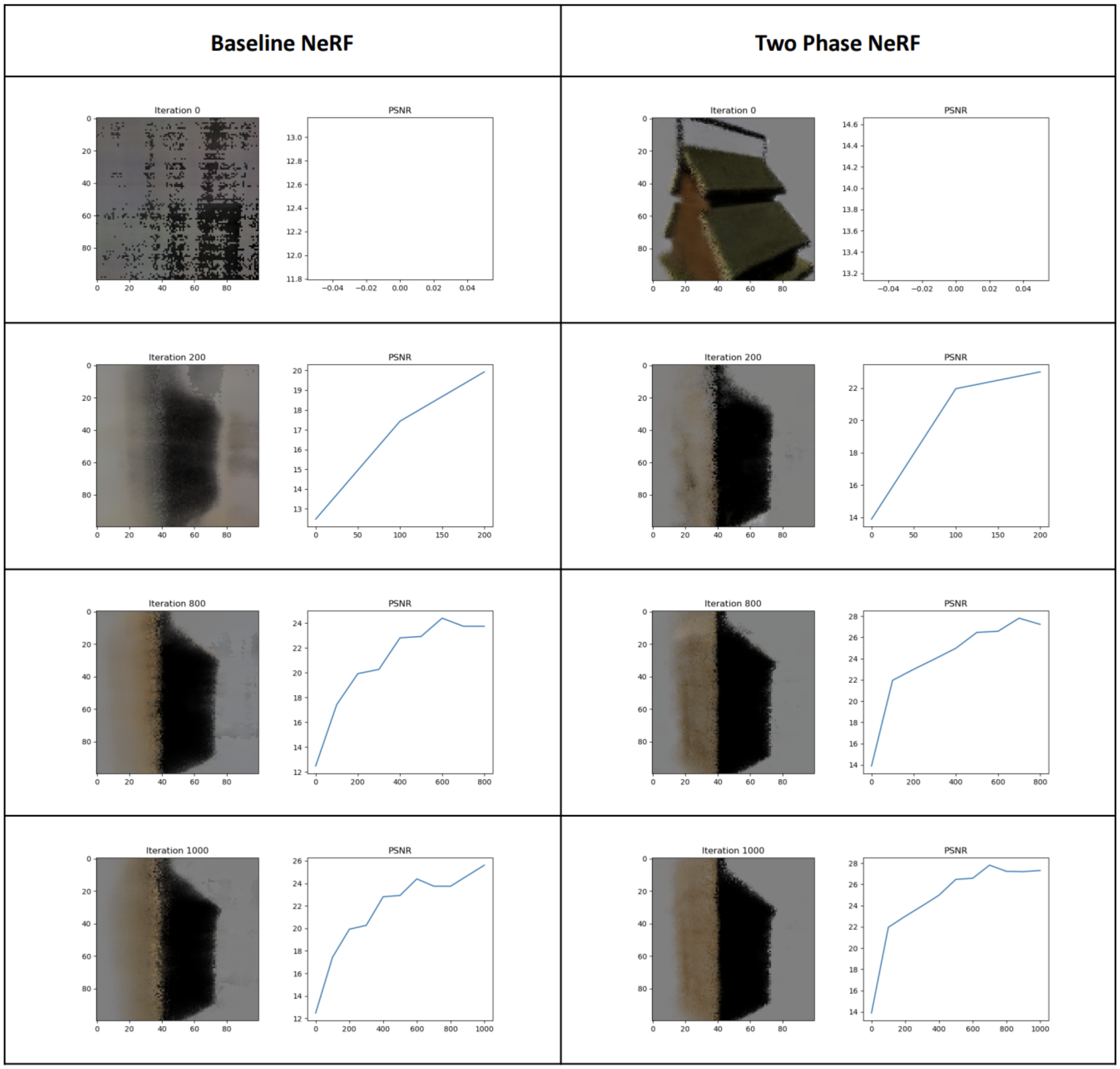


Figure 7: Comparison of Baseline NeRF with Two-Phase NeRF across multiple iterations to reconstruct a view