NeRFing the boundaries: 3D semantic segmentation using 2D supervision

Vaibhav Jade [1] **Rajjeshwar Ganguly** [1] **Keval Pipalia** [1] **Mina Beiramy** [1]

Abstract

Semantic segmentation of 3D objects in scenes is key to a wide variety of problems in fields such as robotics, medical imaging, augmented reality, and architectural visualization. Despite recent advances in 3D segmentation, traditional approaches to carry out segmentation on 3D scenes often require expensive 3D supervision in the form of voxel-based representations, or handcrafted feature design. In contrast, our proposed approach aims to develop an unsupervised, geometry-aware 3D volumetric segmentation method using Neural Radiance Fields (NeRFs). We explore different NeRF architectures to come up with an efficient segmentation model. We primarily focus our efforts on being able to correctly segment indoor scenes.

1. Introduction

Semantic segmentation of objects in 3D for a given scene is key to a wide variety of problems in fields such as robotics, augmented reality, and architectural visualization. 3D segmentation tasks in the past have relied on a joint training method where the depth estimation and segmentation tasks are either treated as two distinct problems that are solved individually in a cascade multi-learning method or they are solved parallelly through a joint training method. Current approaches for segmentation tasks on 3D scenes often require expensive 3D supervision in the form of point-based or voxel-based representations, or handcrafted feature design. However, these new methods rely on gathering 3D representations of data which is often time-consuming and expensive in terms of the work required to generate them and also in terms of computation. Advancements in computing hardware and new data structures like hash encodings have increased the efficiency of storing these 3D representations in memory. However, they are still too expensive in terms of memory and computing requirements which makes their use in fields such as robotics, AR, indoor scene understanding, etc impossible in their current state. Recently there has been a lot of interest in utilizing Neural Radiance Fields for the purpose of 3D generative tasks. However, their application in discriminatory tasks such as segmentation remains scant. Due to recent advances in NeRF models such as InstaNGP (Müller et al., 2022), they have not only become orders of magnitude faster for both training and inference but advances such as GRAF (Schwarz et al., 2021) also use a latent space to learn a manifold of representations observed in the train data. These attributes help this class of models perform well in real-time and on devices such as cell phones due to their efficiency.

In this paper, we propose a novel method of using NeRF models to generalize segmentation masks learned on the data used to train the NeRF model onto a scene. Our approach falls under the paradigm of parallelly learning a class label for each pixel value when the model learns a 3D reconstruction from multiview RGB images. We use COLMAP (Schönberger & Frahm, 2016) to generate camera parameters for our training data and any off-the-shelf image segmentation model suited to the data to generate the segmentation masks on our training set. We show how our modified NeRF model is capable of generalizing these segmentation masks learned on the train set to novel views of the same scene generated by the NeRF model. For our experiments, we compare the original NeRF model as suggested in (Mildenhall et al., 2021) with our own implementations. However, our implementation is architecture agnostic and can work with any other implementations of NeRF as well.

2. Related Work

3D semantic segmentation using different deep learning techniques has been proposed in literature over the years. The particular archetype for these approaches can be broadly classified based on the type of data (RGB-D, voxels, point cloud or other representations) they use or on the type of neural network (MLP, CNN, GCN) used to train and generate an understanding of the scene.

Camera (RGB or Stereo) based methods. When using

[1]Département d'informatique et de recherche opérationnelle, Université de Montréal, Québec, Canada. Correspondence to: Vaibhav Jade <vaibhav.jade@umontreal.ca>.

RGB-D data, an intrinsic correlation between depth and semantic class can be exploited to train depth estimation as proposed by (Liu et al., 2016) and semantic segmentation as a multitask learning problem. An object will have less depth variation within itself than between itself and another object. Based on this (Cao et al., 2017) proposed a deep convolutional neural field framework for depth estimation which is then fed into a two-channel FCN alongside RGB images for the segmentation task. In multitask cascade networks like these the depth estimation task does not get any added supervision from the semantic segmentation task. A parallel network where both these tasks could have some form of parameter sharing to influence each other was therefore proposed by (Wang et al., 2015) through the use of Joint Global CNN. This had the added benefit of exploiting pixel-level depth values and the appropriate semantic label to have a joint training advantage through mutual supervision. (Mousavian et al., 2016) presented a novel approach to explore the depth and semantic mapping at different scales using a multi-scale FCN. Despite some progress in the field, the multi-task learning pipeline was gradually abandoned due to poor quality depth map estimation from RGB images, since RGB-D images were more expensive to produce and train.

LIDAR (point-cloud) based methods. The trajectory of the field post abandonment of multi-task learning paradigms shifted toward the use of 3D supervision using LiDAR-based methods. These methods use point clouds or alternative representations of point cloud data to represent the geometry in a scene. **1) Point-based** methods such as in PointNet (Qi et al., 2017) used a per-point MLP to learn and approximate a permutation invariant set function as a representation of the scene. Later work such as PointConv (Wu et al., 2020) and (Liu et al., 2019) used the same basic principle but with adaptive weights. (Thomas et al., 2019) and (Hua et al., 2018) proposed pseudo grid methods to generalize over local features upon which further work was done by PointASNL (Yan et al., 2020), Point Transformer ((Zhao et al., 2021) and (Engel et al., 2021)) to also learn long-distance dependency by exploiting non-local operators. **2)** Due to the general sampling and grouping inefficiency of point-based methods, **projection-based** methods for LiDAR inputs were adopted. (Lawin et al., 2017), (Boulch et al., 2017) and (Tatarchenko et al., 2018) projected scanned by a rotating LiDAR onto a 2D image by planar projection or like SqueezeSeg (Wu et al., 2017) and SqueezeSegV2 (Wu et al., 2018) where they were projected spherically. These projection-based approaches generally suffered from a loss of information resulting in poor segmentation accuracy. **3)** Using **voxel-based** representations of the input data by passing point data through a 3DCNN as in VoxNet (Maturana & Scherer, 2015) or sparse convolution networks such as in SparseConv (Graham et al., 2017) alleviated this concern to some extent but resulted in sparse representations which had a large memory footprint.

With the inefficiency of camera-based methods and resource costs (both in terms of equipment and computational costs) of LIDAR-based methods, such methods are not feasible for all use cases. We propose a novel segmentation method, with 3D representation learning using:

1. An implicit continuous neural radiance field function learning novel view synthesis using NeRF architecture
2. A novel segmentation map rendering which is architecture agnostic, making it possible to integrate into efficient successors of NeRF, for real-time rendering
3. Utilizing existing image segmentation algorithms as supervision for 3D segmentation understanding

3. Methodology

The complete task of generating a segmentation map using NeRF can be explained as a pipeline of different modules that build upon previous modules and generates a 3D segmentation map. The complete pipeline is visualized in Figure 2. The working of each block is explained later in the section.

3.1. Local Light Field Fusion Dataset

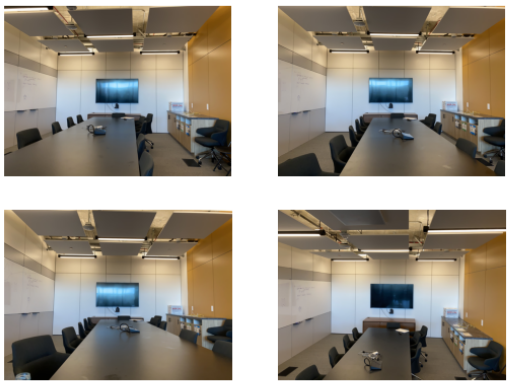


Figure 1. An example of office from downscaled LLFF dataset. The dataset contains 43 images like this from different angles.

The Local Light Field Fusion (LLFF) dataset comprises a collection of light field images used for benchmarking and evaluating LLFF algorithms. It contains both synthetic and real-world datasets that provide challenging lighting and occlusion scenarios. While synthetic datasets are generated

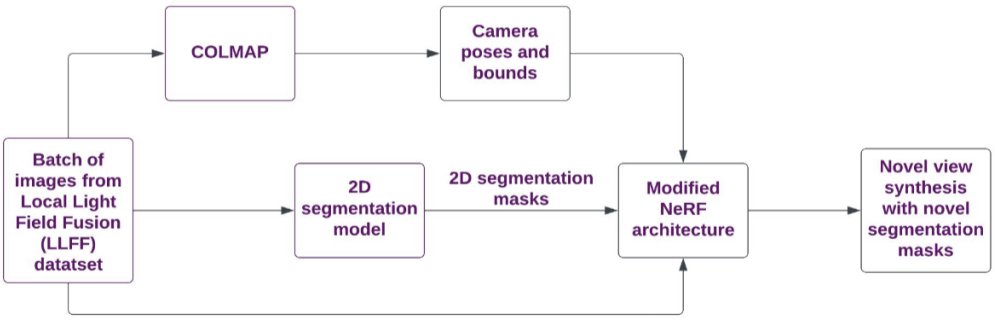


Figure 2. Visualization of segmentation pipeline. Here each block refers to a module that builds upon the output of the previous module.

using computer graphics techniques, real-world datasets are captured using commercial light field cameras and contain scenes with varying levels of complexity. For the purposes of this research, we will utilize samples from the original NeRF LLFF dataset, where each example includes 40-50 image samples. To overcome the limited computing resources we have downscaled these images for our experimentation. These examples contain scenes in indoor settings. An example of the dataset is presented in Figure 1.

3.2. 2D segmentation

Semantic segmentation is the process of assigning each pixel in an image to a specific class label. We implemented fully convolutional neural network (FCN), Feature Pyramid Network (FPN), PSPNet (Pyramid Scene Parsing Network) and DeepLabv3+ models for segmentation task.

The ADE20k dataset (Zhou et al., 2019) is a large-scale scene parsing dataset for semantic segmentation, containing over 20,000 images with pixel-level annotations of over 1,000 object categories. The dataset covers a wide range of indoor and outdoor scenes, and the annotations include not only object classes but also scene attributes and object parts. While the Pascal Context dataset contains over 10,000 images with pixel-level annotations of 59 object classes and their context. It is an extension of the Pascal VOC dataset and includes more detailed annotations of object classes and their surrounding context, making it suitable for research in scene understanding and contextual reasoning.

The fully convolutional network (FCN) approach, as proposed by (Long et al., 2014), preserves spatial information and can be trained end-to-end using backpropagation. This method replaces traditional approaches that rely on hand-crafted features. Feature Pyramid Network (FPN) architecture, developed by (Lin et al., 2017b), creates a pyramid of multi-scale feature maps by reusing the features learned from a backbone network, such as ResNet. At each level of the pyramid, lateral connections from higher resolution feature maps to lower resolution feature maps are added, followed by a 1x1 convolution to fuse the information. The Pyramid Scene Parsing Network (PSPNet) proposed by (Zhao et al., 2017), comprises a convolutional neural network (CNN) with four pyramid pooling modules, a fully connected layer, and a softmax layer. The pyramid pooling module captures the global context information of the image at multiple scales, thus improving the segmentation accuracy. DeepLabv3+ (Chen et al., 2018) combines the strengths of atrous spatial pyramid pooling (ASPP) and encoder-decoder structure to achieve high accuracy in semantic segmentation tasks. The encoder-decoder structure uses a series of down-sampling and up-sampling operations to capture both global and local contextual information in the image.

3.3. COLMAP

To provide the Nerf model with camera poses and image bounds of the room, we integrated a script from LLFF's GitHub (Mildenhall et al., 2019). This script utilizes COLMAP (Schönberger & Frahm, 2016) to generate camera poses and a sparse point cloud for the room images. The script uses an incremental algorithm to compute the camera poses and sparse point cloud for the room images. The results are then refined using global bundle adjustment and Poisson surface reconstruction. The output is represented by a shape of (41, 17), indicating the number of images in the room set and 17 camera parameters. We will reshape this output into three parts.

- (41, 3, 4) representing transformation matrix, to transfer the input into cam2world coordinates.
- (41, 3, 1) representing (H, W, focal length)
- (41, 2) representing (near, far)

These values are used by the NeRF pipeline for ray-marching to generate 5D coordinates which is in turn used to predict color and occupancy by the MLP, rendering color and assigning semantic labels.

3.4. NeRF for 3D segmentation

3.4.1. NeRF-based volume rendering

In NeRF, the 5D neural radiance field represents a scene as the volume density and directional emitted radiance at any point in space. The color of any ray passing through the scene is rendered using principles from classical volume rendering. The volume density $\sigma(x) : (-\infty, \infty) \rightarrow [0, \infty)$, can be interpreted as the differential probability of a ray terminating at an infinitesimal particle at location x. The expected color $C(r)$ of camera ray $r(t) = o + td$ with near and far bounds t_n and t_f is, using transmittance at point t as $T(t)$:

$$T(t) = \exp\left(-\int_{t_n}^{t} \sigma(r(s))ds\right)$$

$$C(r) = \int_{t_n}^{t_f} T(t)\sigma(r(t))c(r(t), d)dt \tag{1}$$

We numerically estimate this continuous integral using quadrature, taking discrete samples along the ray, where $\delta_i = t_{i+1} - t_i$ is the distance between adjacent samples:

$$T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right)$$

$$\hat{C}(r) = \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i\delta_i))c_i \tag{2}$$

3.4.2. Modified NeRF for segmentation rendering

Instead of having the model generate a single-value occupancy $\sigma(x)$ for any given region of the scene, we propose to generate a vector containing the occupancy σ_j. This entails that we not only generate the presence or absence of an object in the given scene but also the class to which it belongs, i.e., a representation of the various semantic class probabilities. We use the same color rendering principles (ray-marching) to generate the 2D segmentation masks.The semantic class $Y(r)$ is given by:

$$Y(r) = softmax\left(\sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i\delta_i))\right) \tag{3}$$

To train the modified function, we utilized the cross-entropy loss between the predicted 2D segmentation map and the ground truth 2D segmentation map. The model was trained on all the images in the training set to ensure its ability to generate 2D segmentation maps from any novel viewpoint. In addition, we introduced a novel volume rendering equation to compare the rendering results with and without color information.

For color rendering, using some function $f_j : \mathbb{R}^C \rightarrow \mathbb{R}$, to project occupancy vector into scalar value,

$$\hat{C}(r) = \sum_{i=1}^{N} T_i \left(1 - \exp(-f_j(\sigma_{[i,j]}) * \delta_i)\right) * c_i \tag{4}$$

We have 2 different pipelines for segmentation map rendering and color rendering, with individual losses. Using this, We experiment with 2 ways of ray marching for color rendering using the projection functions:

1. Taking sum of σ across classes, $\sum_{j=1}^{C} \sigma_j$
2. With color rendering, taking the maximum of σ across classes C, $\max_{j \in C} \sigma_j$

We also experiment with predicting class probabilities at each sample point and then use them for segmentation rendering. We find this can be done in 2 ways:

1. Using the σ value as a direct predictor of class at any given point,

$$y_{[i,j]} = softmax_j\ \sigma_{[i,j]} \tag{5}$$

2. Using a separate linear layer for class prediction in the MLP,

$$y_{[i,j]} = softmax_j\ \hat{y}_{[i,j]} \tag{6}$$

Using the above class predictions, we can get render the semantic labels in the same way as we do for color rendering,

$$\hat{Y}(r)_j = softmax_j\left(\sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i\delta_i))y_{[i,j]}\right) \tag{7}$$

Which is the final predicted pixel-wise class label in the semantic segmentation map.

4. Results and discussion

4.1. 2D segmentation

The output from the segmentation model is represented as an array of size image-height x image-width, where each point in the array represents a class number from the segmentation map. We evaluated the performance of DeepLabv3+, PSPnet, FCN, and FPN models for semantic segmentation on the ADE20k and Context-59 datasets. The models were fine-tuned on these datasets, and their performance was assessed through a quantitative assessment based on the calculation of mIOU scores, which is described in the appendix section.

Table 1. Comparison of segmentation models after fine-tuning on ADE20K and Pascal Context 59 datasets. Numbers here represent % of mIoU score calculated on given separate test set with the datasets.

Model Architecture Name	ADE20K	Context 59
FCN	39.61	51.38
FPN	39.35	N/A
DeeplabV3+	**44.60**	**53.2**
pspnet	43.57	**52.47**

Upon conducting experiments with the ADE20K and Pascal Context 59 datasets, we observed that the Context 59 dataset was better suited for indoor scene labeling. Although Deeplabv3+ produced higher mIoU scores, the semantic maps generated by the PSPNet model were more lucid and accurately aligned with indoor scenes. After conducting multiple experiments on various scenes, we selected the PSPNet model as the default model for our pipeline. Experimentation with the FPN model was discontinued owing to its unsatisfactory performance on the ADE20K dataset. An example of a generated semantic map from the PSPNet model is presented below in Figure 3.

Figure 3. A segmentation map of an image of a room from LLFF dataset generated with pspnet model finetuned on context 59 dataset.

In order to provide a quantitative explanation for the performance of the model, we are engaged in the process of generating a customized validation set from the LLFF dataset. The primary challenge in this endeavor lies in the generation of authentic semantic maps, which require the classification of each pixel. Presently, we are developing a framework that facilitates the direct evaluation of the model's performance on the LLFF dataset. In this regard, we have utilized the Dataloop platform to manually label a few examples. The primary objective of this dataset is to furnish us with quantitative evidence to substantiate our choice of model, rather than to train the model further. Figure 4 illustrates

Figure 4. Segmentation map generated with dataloop platform.

the true segmentation maps generated using the Dataloop platform.

Following our experimentation with the IterNet dataset (Yang et al., 2020), we discovered that the segmentation annotations did not match our initial expectations. The IterNet RGB-D dataset, includes high-resolution RGB images, precise depth maps, and pixel-level instance labels for numerous intricate layouts. However, our findings from experimenting with the dataset revealed that the labels were generated exclusively for data accompanied by depth maps.

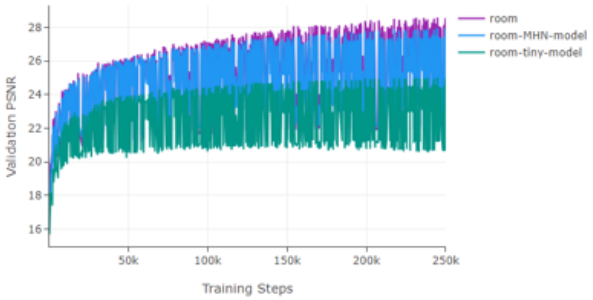
Figure 5. An example of a segmentation map generated from the Iternet dataset is shown below. The labels assigned to each chair are different, indicating instances that are not relevant to our segmentation task.(Yang et al., 2020)

The illustrated image in figure 5 demonstrates distinct segmentation classes for each chair. Consequently, we deduced that to effectively execute the segmentation pipeline with IterNet data, precise depth maps, and pixel-level semantic labels are required, which unfortunately are not available in our LLFF dataset.

4.2. NeRF Architecture

Training the original NeRF implementation requires a long training time ($\sim$ 13 hrs on RTX 8000) due to its ray marching per pixel and hierarchical sampling along the ray for volume rendering as well as its need to overfit on the multi-view image dataset. We experimented with simplified versions of the proposed architecture and perform a comparative ablation study on training time vs performance. We proposed the following architectures:

1. NeRF paper implementation: 8-layer MLP with separate prediction heads for occupancy and color, with residual connections and stochastic depth
2. Simplified Multi-headed NeRF: 6-layer MLP with separate prediction heads for occupancy and color
3. Tiny version of NeRF: Simple 3-layer MLP



(a) Validation PSNR

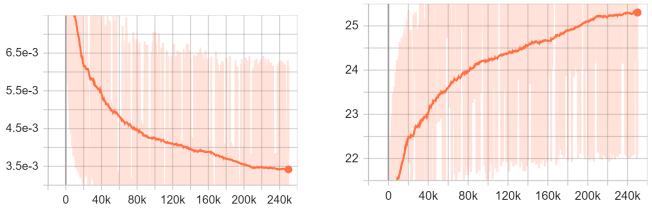
Figure 6. Comparison between (purple) NeRF paper implementation (blue) multi-headed NeRF (green) Simple 3-layer NeRF model

Model architecture	Val loss	Val PSNR	Train time
NeRF (Paper)	0.0038	24.2	$\sim$13 hrs
Multi-headed NeRF	0.0046	23.36	$\sim$12 hrs
Tiny NeRF	0.0086	20.65	$\sim$7 hrs

Table 2. Comparative study between NeRF architectures

As evident from Table (2), we decided to use the original paper implementation of NeRF as the performance and computation cost trade-off did not work in favor of other architectures.

The following plots represent the training and validation metrics of training the original NeRF MLP model (with 7 layers) on the room LLFF dataset.



(a) Validation MSE loss (b) Validation PSNR

Figure 7. Validation MSE loss and psnr.

Our dataset comprised of RGB images with a resolution (4031x3024). We trained the model by downsampling the images by a factor of 8 (effective resolution 504x378) and generated 120 novel views of the scene. We compared the structural similarity of the rendered images with the hold-out test set using the Structural Similarity Index SSIM (Wang et al., 2004) score.

As the rendered poses do not align with test set poses, we randomly permute the render images and calculate the SSIM score. As the rendered poses lie in the same hemispherical camera pose region, we hypothesize that the images won't vary that much semantically. The mean SSIM score was 0.55.

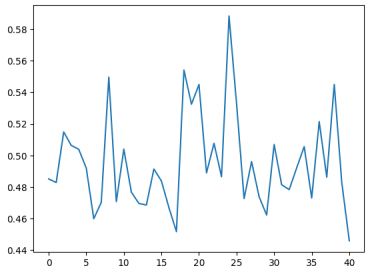


Figure 8. SSIM similarity score on the test set

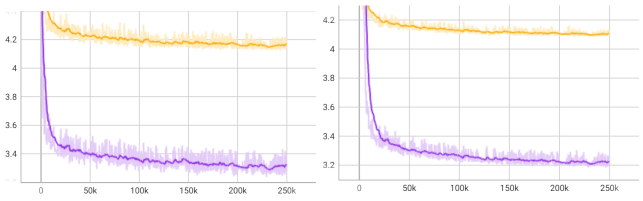
4.3. Modified NeRF for 3D segmentation

We have proposed multiple methods for different components of modified volume rendering for segmentation, analogous to hyperparameters for our pipeline. For choosing the best configuration, we conduct an iterative ablation study, greedily choosing the best configuration for subsequent hyperparameters.

4.3.1. HYPERPARAMETER TUNING FOR NOVEL VOLUME RENDERING

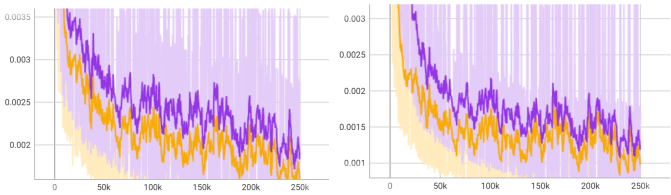
As listed in the methodology section, we experimented with 2 ways of color rendering, using summation or calculating maximum value as the projection function f_j. We found out that both methods do not result in a significant difference in performance. We believe this is caused due conforming the individual occupancy values accordingly due to the final softmax over the class labels. Similarly, we also found out that the 2 ways of predicting class probability per sample point along the ray, using occupancy values directly (equation 5) or with a separate prediction layer (equation 6), resulted in similar performance.

Following this, We experimented with the 2 methods of semantic map rendering, one without adding class probabilities as per equation (3) and one with added class probabilities as explained in equations (4),(6), and (7). The following curves denote the total loss, which is sum of cross-entropy loss (segmentation) and MSE loss (color).



(a) Total train coarse model loss (b) Total train fine model loss

Figure 9. (Yellow) Without using per sample class probabilities (Purple) With using per-sample class probabilities for semantic segmentation rendering.



(a) MSE coarse Validation loss (b) MSE fine Validation loss

Figure 10. (Yellow) Without using per sample class probabilities (Purple) With using per-sample class probabilities for semantic segmentation rendering

As from the training plot, we can see that adding class prediction per ray sample increased the performance, and we achieved a much better convergence point (loss reduced to $\sim$3.3 from $\sim$4.2).

We also observe that the training loss of our modified NeRF model reduced to an approximately constant value of 6.4, after $\sim$ 50k steps. This can be attributed to a major class imbalance in our semantic class labels, which might be causing our cross-entropy loss to converge quickly. We experimented with methods to handle this class imbalance, specifically focal loss.

Furthermore, we observe that the coarse model performs slightly worse than the fine model, which aligns with the ray sampling used for respective models. This can also be observed from artifacting effect in the rendered images.

4.3.2. CLASS IMBALANCE AND FOCAL LOSS

We analyzed the class distribution in the room dataset, for the segmentation maps generated from the PSPNet. The following graph highlights the class imbalance in the segmentation class labels:

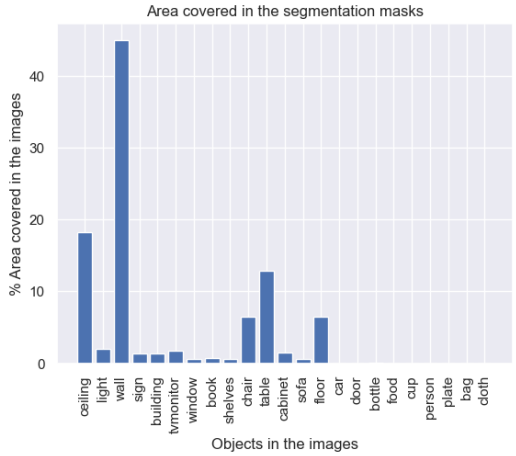
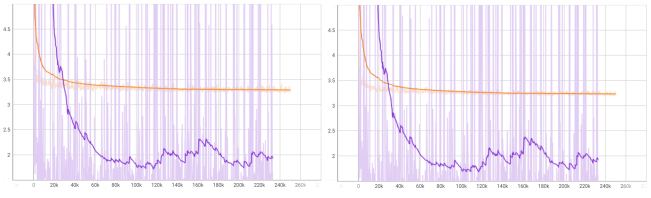


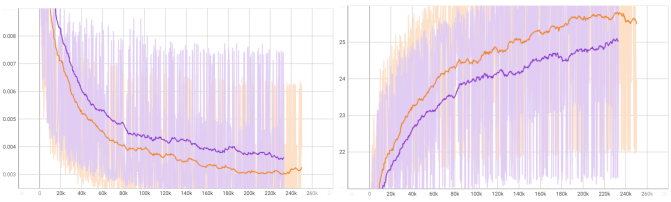
Figure 11. Semantic class labels distribution in LLFF room dataset

We experimented with Focal loss (Lin et al., 2017a), which compensates for class imbalance with class weighting and higher order penalty for bad classifications. Here, we use summation for projection function f_j, per ray sample class prediction using occupancy vector σ. We also removed the excess classes predicted by the 2D segmentation pipeline (59 classes), reducing to 19 classes pertinent to our room LLFF dataset. The following plots summarize the results:



(a) Total train coarse model loss (b) Total train fine model loss

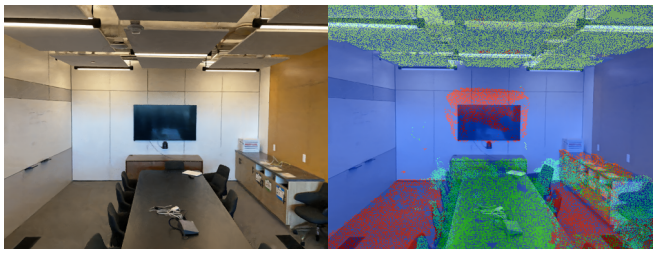
Figure 12. (Yellow) Using Cross Entropy loss for segmentation (Purple) using Focal loss for segmentation rendering.



(a) MSE coarse Validation loss (b) MSE fine Validation loss

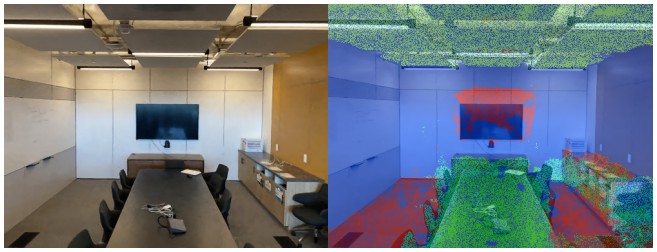
Figure 13. (Yellow) Using Cross Entropy loss for segmentation (Purple) using Focal loss for segmentation rendering.

The following rendered images demonstrate novel views generated by the NeRF model, accompanied by their semantic segmentation maps which the NeRF model generalized to novel views only from having the segmentation maps of the training data.



(a) Rendered image (b) Overlapped segmentation mask

Figure 14. Rendering from a novel viewpoint.



(a) Rendered image (b) Overlapped segmentation mask

Figure 15. Rendering from a novel viewpoint.

5. Conclusion

In our work, we presented an efficient way of learning a 3D segmentation field for a given scene. We showed with our experiments that NeRF models are capable of learning more than just the radiance field for object geometry in any given scene. Using 2D semantic supervision it is feasible for a NeRF model to represent the class of objects in a scene with the same intrinsic continuous neural radiance field function as the one used for representing object geometry. This allows us to generate novel views of a scene with a learned segmentation mask for the synthesized view without explicit supervision. The generated masks are noisy and imprecise owing to the quality of the semantic masks of our training data. However, this can be reduced with post-processing using a Gaussian smoothing filter (through a convolution) method.

Due to the inherent efficiency of only requiring a handful of multiview images to generate a depthwise understanding of the semantic layout of a scene, this approach can have uses for autonomous navigation in robotics or for uses in AR such as interior design.

6. Source Code

The source code for the project is available publicly and can be accessed with this URL: `https://github.com/jadevaibhav/NeRF4Seg`

References

Boulch, A., Saux, B. L., and Audebert, N. Unstructured Point Cloud Semantic Labeling Using Deep Segmentation Networks. In Pratikakis, I., Dupont, F., and Ovsjanikov, M. (eds.), *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2017. ISBN 978-3-03868-030-7. doi: 10.2312/3dor.20171047.

Cao, Y., Shen, C., and Shen, H. T. Exploiting depth from single monocular images for object detection and semantic segmentation. *IEEE Transactions on Image Processing*, 26(2):836–846, 2017. doi: 10.1109/TIP.2016.2621673.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018.

Engel, N., Belagiannis, V., and Dietmayer, K. Point transformer. *IEEE Access*, 9:134826–134840, 2021. doi: 10.1109/ACCESS.2021.3116304.

Graham, B., Engelcke, M., and van der Maaten, L. 3d semantic segmentation with submanifold sparse convolutional networks, 2017.

Hua, B.-S., Tran, M.-K., and Yeung, S.-K. Pointwise convolutional neural networks, 2018.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lawin, F. J., Danelljan, M., Tosteberg, P., Bhat, G., Khan, F. S., and Felsberg, M. Deep projective 3d semantic segmentation, 2017.

Lin, T., Goyal, P., Girshick, R. B., He, K., and Dollár, P. Focal loss for dense object detection. *CoRR*, abs/1708.02002, 2017a. URL http://arxiv.org/abs/1708.02002.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. Feature pyramid networks for object detection, 2017b.

Liu, F., Shen, C., Lin, G., and Reid, I. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2024–2039, oct 2016. doi: 10.1109/tpami.2015.2505283. URL https://doi.org/10.1109%2Ftpami.2015.2505283.

Liu, Y., Fan, B., Xiang, S., and Pan, C. Relation-shape convolutional neural network for point cloud analysis, 2019.

Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation, 2014.

Maturana, D. and Scherer, S. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 922–928, 2015. doi: 10.1109/IROS.2015.7353481.

Mildenhall, B., Srinivasan, P. P., Ortiz-Cayon, R., Kalantari, N. K., Ramamoorthi, R., Ng, R., and Kar, A. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Mousavian, A., Pirsiavash, H., and Kosecka, J. Joint semantic segmentation and depth estimation with deep convolutional networks, 2016.

Müller, T., Evans, A., Schied, C., and Keller, A. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. doi: 10.1145/3528223.3530127. URL https://doi.org/10.1145/3528223.3530127.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2017.

Schönberger, J. L. and Frahm, J.-M. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Schwarz, K., Liao, Y., Niemeyer, M., and Geiger, A. Graf: Generative radiance fields for 3d-aware image synthesis, 2021.

Tatarchenko, M., Park, J., Koltun, V., and Zhou, Q.-Y. Tangent convolutions for dense prediction in 3d, 2018.

Thomas, H., Qi, C. R., Deschaud, J.-E., Marcotegui, B., Goulette, F., and Guibas, L. Kpconv: Flexible and deformable convolution for point clouds. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6410–6419, 2019. doi: 10.1109/ICCV.2019.00651.

Wang, P., Shen, X., Lin, Z., Cohen, S., Price, B., and Yuille, A. Towards unified depth and semantic prediction from a single image. pp. 2800–2809, 06 2015. doi: 10.1109/CVPR.2015.7298897.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. URL http://dblp.uni-trier.de/db/journals/tip/tip13.html#WangBSS04.

Wu, B., Wan, A., Yue, X., and Keutzer, K. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud, 2017.

Wu, B., Zhou, X., Zhao, S., Yue, X., and Keutzer, K. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud, 2018.

Wu, W., Qi, Z., and Fuxin, L. Pointconv: Deep convolutional networks on 3d point clouds, 2020.

Yan, X., Zheng, C., Li, Z., Wang, S., and Cui, S. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling, 2020.

Yang, J., Xu, J., Li, K., Lai, Y.-K., Yue, H., Lu, J., Wu, H., and Liu, Y. Learning to reconstruct and understand indoor scenes from sparse views. *IEEE Transactions on Image Processing*, 29(1):5753–5766, 2020.

Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. Pyramid scene parsing network, 2017.

Zhao, H., Jiang, L., Jia, J., Torr, P., and Koltun, V. Point transformer, 2021.

Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., and Torralba, A. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019.

A. MIoU: Mean Intersection over Union

MIoU stands for Mean Intersection over Union, and it is a widely used evaluation metric in image segmentation tasks. It measures the overlap between the predicted segmentation map and the ground truth segmentation map, by calculating the intersection of the two maps divided by their union for each class.

To compute the MIoU score, the pixel-wise IoU score for each class is first calculated, and then the average of these scores is taken across all classes. The MIoU score ranges between 0 and 1, where a score of 1 indicates perfect segmentation, and a score of 0 indicates no overlap between the predicted and ground truth segmentation maps.

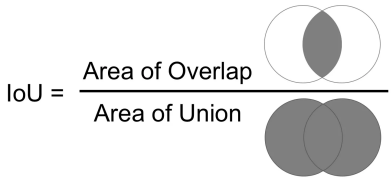


Figure 16. Formulae for calcultation of IoU

MIoU is a useful metric for comparing the performance of different segmentation models, as it takes into account both the accuracy of the predicted labels and their spatial coherence. Higher MIoU scores indicate better segmentation performance.