Project Proposal: Super Efficient Neural Light Field

Austin Peng
Georgia Institute of Technology
apeng39@gatech.edu

1. Introduction

1.1. Target Problem & Importance

View synthesis is a challenge in both computer vision and graphics, focusing on generating realistic images of a scene from unobserved camera poses, given a limited set of real input images. Current approaches address this by optimizing representations that capture the scene's appearance and geometry.

Neural radiance fields (NeRF) set the state-of-the-art (SOTA) in rendering quality for view synthesis by mapping a 5D (x, y, z, θ, ϕ) input to 4D (R, G, B, σ) output [10]. However, NeRFs have a notable drawback: the multilayer perceptron (MLP) is sampled hundreds of times per pixel for accurate volume rendering, resulting in slow rendering times for high-resolution images [15].

The target problem is the high training and inference time of NeRF models. The computational cost of integrating the radiance field remains a bottleneck for its efficient and real-time implementation. View synthesis finds applications in AR, VR, and cinematography, but current SOTA models require high-end GPUs (for acceleration or significant storage memory) which may not be readily available on resource-constrained devices (ex. mobile). Devices running view synthesis models may lack the computing power of GPUs used in research and industry. Increasing the efficiency of novel view synthesis models is key to expanding its usability.

1.2. Related Works

Efforts to enhance the efficiency of NeRFs use baking and caching approaches, which result in efficient rendering but at the cost of substantial memory consumption. Recent trends have shifted away from volumetric representations due to computational challenges and various performance disadvantages like difficulty representing view-dependent effects like reflection/refraction, rendering sharpness, and high storage space [1, 6].

Numerous attempts have been made to reduce the computational cost of the radiance integration step by distilling meshes from the radiance field. Among these approaches, only a couple of mesh-based methods have been able to achieve real-time rendering capabilities, but also at the cost of rendering quality and usage of over 200 MB of storage [3, 6, 11, 15].

Neural light field (NeLF) utilizes a light field (integral of radiance) to reduce the hundreds of samples per ray to a single sample per ray. The idea of NeLF has demonstrated comparable performance to NeRF but with a better trade-off between rendering quality, speed, and memory [1, 13].

Recent works explore the possibilities of making NeLF efficient, real-time, and suitable for resource-constrained environments like mobile devices [2, 6]. The initial **r**adiance **to** **l**ight (R2L) approach focused on network architecture optimization through distilling knowledge from a pretrained NeRF to a custom NeLF with a 88-layer DNN. This architecture achieved $26 \sim 35\times$ FLOPs reduction and $28 \sim 31\times$ wall-time acceleration on the Realistic Synthetic $360°$ dataset [10] with significantly improved rendering quality ($1.4 \sim 2.8$ dB average PSNR improvement) [13]. Subsequent works like MobileR2L [2] and LightSpeed [6] further refine the proposed DNN MLP for efficiency gains while maintaining or improving rendering quality.

1.3. Proposed Solution

Existing works show the importance of the MLP network in SOTA models for view synthesis. I propose to apply common compression techniques of pruning, quantization, and others (ex. depthwise separable convolution) on a SOTA view synthesis model for real-time resource-constrained rendering like [2]. The goal is to first individually explore the compression techniques on the MobileR2L model before determining the best combination to apply to the final model. There are many cases in published research where combining pruning and quantization can yield efficiency gains at low cost in performance [4, 7, 9, 12]. This approach makes sense as the MLP network within the MobileR2L architecture seems to play a key role in rendering quality and efficiency. There have been multiple iterations in research proposing novel and better solutions for the MLP [1, 2, 6, 13].

1.3.1 Milestones

- Reproduce results from MobileR2L paper (published in CVPR 2023).
- Explore the effects of depthwise separable convolutions on MobileR2L architecture.
- Explore the effects of unstructured/structured pruning at different ratios on MobileR2L architecture.
- Explore the effects of quantization (and/or CPT, QAT, etc.) on MobileR2L architecture.
- Determine which compression techniques and combinations yield the best-performing model for real-time rendering on resource-constrained devices. Confirm findings with CoreMLTools. Involves latency (ms), storage (MB), and FLOPs.

Desired Deadline	Task
02/26 (Mon.)	Project Proposal Submission
03/04 (Mon.)	Reproduce MobileR2L Results
03/11 (Mon.)	Explore Convolutions/Pruning
03/18 (Mon.)	Explore Quantization
03/25 (Mon.)	Combined Experiments + Results
04/01 (Mon.)	Combined Experiments + Results
04/08 (Mon.)	Write Project Report + Poster
04/17 (Wed.)	Final Project Submission

1.4. Expected Outcomes

The expected outcome of applying compression techniques are improvements in model efficiency in terms of latency (ms), storage (MB), and FLOPs while maintaining similar performance in peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and learned perceptual image patch similarity (LPIPS). The following are the efficiency and performance metrics of MobileR2L. Note, the latency calculations are performed on iPhone 14 with iOS 16 using CoreMLTools package.

Metric \ Dataset	Synthetic 360° [10]	RFF [10]
Storage (MB)	8.3	8.3
Latency (ms)	16.48	22.65
PSNR	26.15	30.90
SSIM	0.966	0.947
LPIPS	0.187	0.062

My version of MobileR2L with applied compression techniques should easily be able to achieve real-time speeds (22 ms $\approx$ 45 FPS) and use minimal storage (<8.3 MB) on resource-constrained devices (ex. iPhone 14).

With pruning, I am hoping that I can perform 30%-50% pruning of the model weights without significantly affecting rendering quality. The specific pruning technique will involve unstructured weight pruning and structured channel-wise pruning. With quantization, I hope to experiment with 4, 8, and 16-bit quantization of weights. A stretch goal is to experiment with different types of quantization (ex. hybrid, selective, and post-training quantization). Improvements with unstructured weight pruning and quantization will be theoretical speedups as they usually require specifically designed software or hardware to achieve actual acceleration [5]. The efficiency improvements should come from channel-wise pruning.

The MobileR2L architecture uses a residual structure similar to ResNet [13], with the addition of super-resolution modules. This makes me believe that with channel-wise pruning (ex. sensitivity, scaling factor), we can hope to see up to $\sim 50\%$ FLOP reduction in ResNet-like models with a minimal decrease in performance (sources found $< 2\%$ decrease on CIFAR-10) [8, 14]. Although multiclass classification and view synthesis are completely different, the idea remains the same in terms of maintaining the more important model weights and the pruning of less important model weights.

1.5. Evaluation

The evaluation will be conducted mainly through latency (ms) measured on a real device (ex. iPhone with CoreMLTools), storage (MB), and FLOPs (M). Performance metrics such as PSNR, SSIM, and LPIPS will be included to ensure the compressed model can maintain similar rendering quality.

I will be evaluating the MobileR2L architecture [2] and pruned/quantized versions of MobileR2L on the Real Forward-Facing (RFF) dataset and Realistic Synthetic 360° [10].

References

[1] Benjamin Attal, Jia-Bin Huang, Michael Zollhoefer, Johannes Kopf, and Changil Kim. Learning neural light fields with ray-space embedding networks, 2022. 1

[2] Junli Cao, Huan Wang, Pavlo Chemerys, Vladislav Shakhrai, Ju Hu, Yun Fu, Denys Makoviichuk, Sergey Tulyakov, and Jian Ren. Real-time neural light field on mobile devices, 2023. 1, 2

[3] Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures, 2023. 1

[4] Elias Frantar and Dan Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning, 2022. 1

[5] Shangqian Gao, Feihu Huang, Weidong Cai, and Heng Huang. Network pruning via performance maximization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9270–9280, 2021. 2

[6] Aarush Gupta, Junli Cao, Chaoyang Wang, Ju Hu, Sergey Tulyakov, Jian Ren, and László A Jeni. Lightspeed: Light and fast neural light fields on mobile devices, 2023. 1

[7] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding, 2016. 1

[8] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks, 2017. 2

[9] Peng Hu, Xi Peng, Hongyuan Zhu, Mohamed M Sabry Aly, and Jie Lin. Opq: Compressing deep neural networks with one-shot pruning-quantization, 2021. 1

[10] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020. 1, 2

[11] Sara Rojas, Jesus Zarzar, Juan Camilo Perez, Artsiom Sanakoyeu, Ali Thabet, Albert Pumarola, and Bernard Ghanem. Re-rend: Real-time rendering of nerfs across devices, 2023. 1

[12] Frederick Tung and Greg Mori. Deep neural network compression by in-parallel pruning-quantization. *IEEE transactions on pattern analysis and machine intelligence*, 42(3): 568–579, 2018. 1

[13] Huan Wang, Jian Ren, Zeng Huang, Kyle Olszewski, Menglei Chai, Yun Fu, and Sergey Tulyakov. R2l: Distilling neural radiance field to neural light field for efficient novel view synthesis, 2022. 1, 2

[14] Chenbin Yang and Huiyi Liu. Channel pruning based on convolutional neural network sensitivity. *Neurocomputing*, 507:97–106, 2022. 2

[15] Lior Yariv, Peter Hedman, Christian Reiser, Dor Verbin, Pratul P. Srinivasan, Richard Szeliski, Jonathan T. Barron, and Ben Mildenhall. Bakedsdf: Meshing neural sdfs for real-time view synthesis, 2023. 1