Conditional GAN is all you need for MR2CT

Xia Li[1,2][0000-0003-2284-1700] and Ye Zhang[1][0000-0003-1608-4467]

Center for Proton Therapy, Paul Scherrer Institut, Villigen, Switzerland
{xia.li, ye.zhang}@psi.ch
Department of Computer Science, ETH Zurich, Zurich, Switzerland
{xia.li}@inf.ethz.ch

Abstract. Within this note, we describe the framework we used for the SynthRAD data challenge. With the kernel of a 3D nnUNet [1], it proves the power of convolutional networks even when Transformers are the most popular. The main contribution of this framework is an efficient 3D augmentation strategy, which takes the majority of the performance gain over the baseline. Besides, we did minor modifications to the module design and verified their effectiveness.

Keywords: MR-based CT Synthesis · Conditional GAN · Pix2pix.

1 Methods

In this section, the proposed framework's intricacies are presented, which comprise two main components: data augmentation and network structures.

1.1 Augmentation

It is widely accepted that data quality supersedes algorithmic complexity in real-world applications. In this challenge, given that no additional dataset is permissible, optimizing the use of the provided dataset becomes paramount. Traditional data augmentation primarily focuses on 2D images, encompassing techniques like random flipping, resizing, and rotating. Directly transferring these techniques to 3D images can be challenging. While one could argue that extracting 2D slices or 3D patches from 3D volumes for augmentation is straightforward, this approach doesn't harness the full potential of the 3D structure. Another method involves augmenting the entire volume before extracting slices to feed into the networks. This latter approach, however, compromises either randomness (by feeding sequentially) or efficiency (by feeding only once).

Our proposed solution deviates from these conventional methods. We suggest augmenting the sampling plane, a method that minimizes computation while allowing slices/patches from various volumes to be combined into a mini-batch. Initially, a standard sampling plane is constructed with center coordinates at $(0, 0, 0)$ and dimensions of $[4, 128, 128]$. This plane undergoes random resizing between $[0.7, 1.3]$, random rotations between $[-30, 30]$ degrees, and random flipping. After determining random offsets to the center—ensuring sampling remains within the mask's Volume of Interest—all sampling plane coordinates are adjusted by these offsets. Subsequently, patches are sampled from the current volume. MRs and CTs are sampled using trilinear interpolation, while masks use the nearest neighbor method.

1.2 Network Structure

Despite the emergence of attention-based networks in medical image processing, UNet remains the prevalent architecture. Notably, nnUNet, a mature CNN-based framework, has demonstrated its efficacy on several segmentation benchmarks. We leveraged the 3D variant for our challenge, accommodating input and output dimensions of $[4, 128, 128]$. Beginning with a $3\times3\times3$ convolution, the architecture employs four pairs of downsampling and upsampling CNN blocks, with channels sized $[128, 256, 512, 512]$. Originally, each block contained two successive $3 \times 3 \times 3$ convolutional layers. However, these made the network sluggish and prone to overfitting. To counteract this, the layers were substituted with three sequential convolutional layers having kernel dimensions of $1 \times 3 \times 3$, $3 \times 1 \times 1$, and $1 \times 3 \times 3$, reducing the parameter count to approximately $\frac{7}{18}$ of the initial configuration. Moreover, the MR patch was fused with the associated mask as network input, and the predicted CT was post-processed by mask multiplication.

2 Experiments

2.1 Implementations

Pytorch [2] was employed to develop the entire framework, with network training executed on our clusters. Each experimental run utilized four RTX-2080-TI GPUs. To expedite training, we incorporated the accelerate library [4] from Hugging Face in conjunction with Microsoft's DeepSpeed [3] tool. This setup enabled the “fp16” option, diminishing memory usage and training duration. Additionally, the ”stage2” option from DeepSpeed was activated to further optimize memory allocation.

2.2 Training Details

Owing to significant distribution differences between regions, separate networks were designated for each region. This strategy resulted in training sets comprising 150 images per region. To facilitate local evaluation, 30 images from each region were designated as the ”trainval” set, with the remaining 120 images reserved for training. It should be noted that the final model submission underwent training on the complete dataset.

For every experiment, a cosine learning schedule was employed, initiating with a learning rate of 2e-4. The chosen batch size was 16. Routine experiments were capped at 100 epochs, while the final submission extended to 200 epochs.

3 Conclusion

An efficient and potent framework was developed for this data challenge. Stripped of superfluous complexities, our streamlined framework delivered commendable outcomes.

References

1. Isensee, F., Jaeger, P. F., Kohl, S. A., Petersen, J., & Maier-Hein, K. H. (2021). nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. Nature methods, 18(2), 203-211.
2. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems, 32.
3. Rasley, J., Rajbhandari, S., Ruwase, O., & He, Y. (2020, August). Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 3505-3506).
4. Sylvain Gugger, B.. (2022). Accelerate: Training and inference at scale made simple, efficient and adaptable.