Distributed Deep Learning for Kidney Tumor Segmentation

Hosein Beheshtifard
Computer Science
University of Calgary
Calgary, Canada
hosein.beheshtifard@ucalgary.ca

ABSTRACT

Kidney cancer is one of the most common malignant tumors in the world. Kidney segmentation is vital for quantitative analysis, monitoring disease progression, and treatment planning, offering essential data on tumor location, size, and shape. Manual segmentation of kidney tumors is time-consuming and subjective, as it relies on the expertise of radiologists and surgeons to interpret the imaging data. Therefore, there is a critical need for a more efficient and precise method for segmenting kidney tumors from CT images. Deep learning models have shown promise in medical image analysis, and can potentially provide objective and accurate segmentation of kidney tumors. While these models show promising results in the segmentation tasks, they require a resource-intensive and time-consuming training process. This becomes particularly pronounced when dealing with CT images in the form of 3D volumetric data. In this project, we developed a 2D U-Net base model with ResNet34 encoders. We trained and tested our model using the KITS23 dataset, which includes 489 patients who underwent partial or radical nephrectomy for suspected renal malignancy. Furthermore, we used parallelization techniques to train our model over multi-GPUs. Our model achieved an average dice score of 0.937 for kidney segmentation and 0.788 for tumor segmentation. Notably, the training process was accelerated by 3.15 times through distributed learning over multiple nodes.

KEYWORDS

Medical Imaging, Kidney Tumors, Semantic Segmentation, Deep Learning, U-Net model, Data Parallelization, Distributed Learning

1 Introduction

Kidney cancer is the 13th most common cancer worldwide, accounting for 2.4% of all cancers, with more than 330,000 new cases diagnosed yearly, and its incidence is still increasing [1]. The most common type of kidney cancer is renal cell carcinoma, which accounts for about 90% of all cases [2]. The treatment of renal cell carcinoma often involves surgical removal of the tumor, which requires accurate preoperative planning to ensure complete removal of the tumor while minimizing damage to the surrounding healthy tissue. Therefore, accurate segmentation of the tumor and surrounding anatomy from CT images is crucial for effective treatment and better patient outcomes.

Manual segmentation of kidney tumors is time-consuming and subjective, as it relies on the expertise of radiologists and surgeons to interpret the imaging data. Automated segmentation methods have the potential to overcome these limitations and provide objective and standardized measurements of tumor size, shape, and appearance. Over the years, different methods have been proposed to improve the accuracy and efficiency of automated kidney tumor segmentation.

One of the earliest approaches to kidney tumor segmentation was region growing, which involves selecting a seed point within the tumor and growing a region around it that satisfies certain criteria, such as intensity homogeneity. While this method is simple and fast, it is highly dependent on the choice of seed point and can lead to oversegmentation or undersegmentation of the tumor [4].

Another commonly used method is thresholding, which involves selecting a threshold value for the image intensity that separates the tumor from the surrounding tissue. This method is fast and easy to implement, but it is highly dependent on the choice of threshold value and can lead to inaccurate segmentation when the tumor and surrounding tissue have similar intensity values [5, 6].

Recent advancements in machine learning, particularly deep learning models like the U-Net model [7], have demonstrated tremendous potential in various medical imaging applications, including kidney tumor segmentation. While these models show promise in the segmentation tasks, a resource-intensive and time-consuming training process is required. This becomes particularly pronounced when dealing with CT images in the form of 3D volumetric data. The intricate nature of such data necessitates large GPU memories to facilitate the training process.

To address this issue, we initially convert 3D CT images to 2D slices and employ a 2D U-Net-based model, which is more lightweight compared to its 3D counterpart. Additionally, we utilize data parallel processing and distributed learning techniques to train our model across multiple GPUs on the Digital Research Alliance of Canada cluster. Following this, we conduct experiments to demonstrate the impact of data parallelism and distributed learning.

2 Related Work

Kidney Tumor Segmentation: The Kidney Tumor Segmentation Challenge (KiTS) series, starting with KiTS19, was initiated in tandem with the 2019 International Conference on Medical Image

Computing and Computer Assisted Intervention (MICCAI). Its primary objective was to address the complexities of automated segmentation and foster advancements in this domain. The KiTS19 competition offered a training set comprising 210 cross-sectional CT images featuring kidney tumors, accompanied by publicly available corresponding semantic segmentation masks. A total of 106 teams from five continents leveraged this dataset to develop automated systems capable of predicting accurate segmentation masks for a private test set of 90 CT images. Evaluation and ranking were based on the average Sørensen-Dice coefficient for the kidney and tumor across all 90 cases. The winning team employed a 3D U-Net model, achieving a Dice coefficient of 0.974 for the kidney and 0.851 for the tumor [3].

Building upon the success of KiTS19, the KiTS21 challenge continued its legacy and was held concurrently with MICCAI. Its primary goal remained the automatic segmentation of kidneys and tumors in CT images. The challenge provided a training set of 300 CT scans, including 210 from KiTS19 and an additional 90 scans with corresponding semantic segmentation masks for the kidney and tumor regions. Participants were tasked with developing deep learning models to accurately predict segmentation masks for a private test set of 100 CT scans. The teams' performance was assessed using the Sørensen-Dice coefficient, a common metric for evaluating segmentation accuracy. The winning team achieved an impressive Dice coefficient of 0.975 for kidney segmentation and 0.86 for tumor segmentation, employing a coarse-to-fine framework based on the nnU-Net [8].

For the third time, KiTS is inviting the greater research community to participate in a competition to develop the best automatic semantic segmentation system for kidney tumors. This year's competition features an expanded training set (489 cases), a fresh never-before-used test set (110 cases), and the addition of cases in the nephrogenic contrast phase, whereas previously all cases were in late arterial. The winning team achieved an average dice score of 0.835 and a tumor dice score of 0.758. The final reports for this challenge are yet to be published.

Data Parallelism and Distributed Learning: In the domain of deep learning, the application of data parallelism on a single node and distributed learning across multiple machines has become instrumental in addressing the computational demands of training large-scale models. Seminal works in the field have provided crucial insights into the effectiveness of these strategies. Notably, the research by Goyal et al. [9] underscores the significance of data parallelism, demonstrating its efficacy in minimizing training time on a single node. This work has become foundational in understanding the advantages of scaling up batch sizes for accelerated model convergence. Concurrently, advancements in distributed learning have been exemplified by the work of Keskar et al. [10]. The study delves into the challenges and benefits of training deep learning models across multiple machines, shedding light on the generalization gap and the optimization landscape in such distributed settings. In this project, our objective is to conduct experiments utilizing the PyTorch framework to explore the effectiveness of data parallelism on a single node and distributed learning across multiple nodes specifically tailored for the task of kidney tumor segmentation.

3 Dataset

The KiTS23 cohort includes patients who underwent partial or radical nephrectomy for suspected renal malignancy between 2010 and 2022 at either an M Health Fairview or Cleveland Clinic medical center. A retrospective review of these cases was conducted to identify all patients who had undergone a contrast-enhanced preoperative CT scan that includes the entirety of all kidneys.

Each case's most recent corticomedullary preoperative scan was independently segmented three times for each instance of Kidney, Tumor, and Cyst.

The annotation process involves placing 3D bounding boxes around each region of interest, annotating axial slices within these bounding boxes with "guidance pin annotations," and reviewing the annotations by experts and trainees. Laypeople then perform "contour annotations" based on the approved guidance, and the annotations are post-processed to generate segmentations [3].

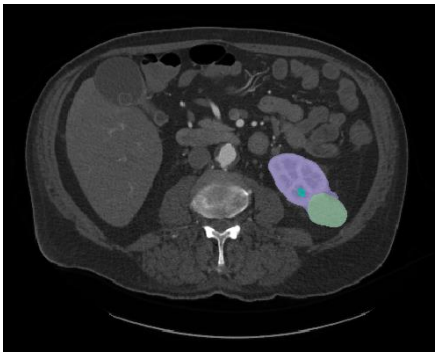

Figure 1: An example of a segmented axial slice from the KiTS23 dataset

The dataset for this project comprises data from 489 patients (publicly available), along with segmentation masks for kidney, tumor, and cyst regions.

4 Methodology

In this work, we employed a 2D model, which is more lightweight compared to 3D models. We utilized a 2D U-Net-based model with ResNet34 encoders using segmentation_models_pytorch library. The project pipeline is illustrated in Figure 3, encompassing preprocessing, train-time random data augmentation, and post-processing.

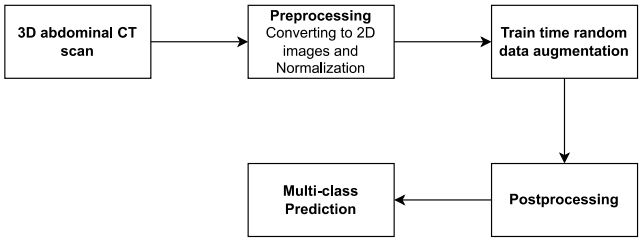

Figure 2: Project pipeline diagram

4.1 Preprocessing

The first step is to convert the 3D images and their related masks into 2D slices.

In the second step, we divided the data into three sets: train, validation, and test, with proportions of 80%, 10%, and 10%, respectively. So we have 391 patient data for training, 49 for validation, and 49 cases for testing that we did not feed the model with during training.

As part of our preprocessing, we changed the intensities based on a range of 0 to 200, which is an approximate range for the kidneys and some soft tissues. Any intensities greater than the maximum value were set to the maximum, and any intensities lower than the minimum value were set to the minimum. This allowed us to disregard other organs with different intensity ranges such as bones or metals. This transformation was only applied during the first phase since we only dealt with kidney segments during the second phase. Additionally, the data was normalized using the training dataset's mean and standard deviation.

Lastly, we added some random augmentations such as random rotation, random flip, random crop, and random Gaussian noise. The aim of these transformations is to increase the variability of the data to avoid overfitting.

4.2 Model Architecture

The chosen model architecture is a U-Net base model. The encoder path is based on a pre-trained ResNet34 that extracts feature maps at different scales. The decoder path generates accurate segmentation masks by combining information from different scales. The U-Net architecture consists of 23 convolutional layers in the encoder and decoder, excluding the final 1x1 convolutional layer, with skip connections between corresponding encoder and decoder blocks. The encoder path gradually reduces the spatial resolution of the input image and increases the number of feature channels. The decoder path gradually upsamples the feature maps to the original input size to produce a segmentation map. Each block in the encoder consists of two convolutional layers followed by a max-pooling layer, while each block in the decoder comprises an upsampling layer and a convolutional layer. The number of feature channels is doubled after each downsampling step and halved after each upsampling step, allowing the network to learn increasingly abstract representations of the input image and gradually refine the segmentation map. The skip connections concatenate feature maps from the corresponding encoder block with the upsampled feature maps in the decoder, which helps the network capture fine-grained details while maintaining information.

We used transfer learning using pre-trained weights on RadImageNet [11] for our final model.

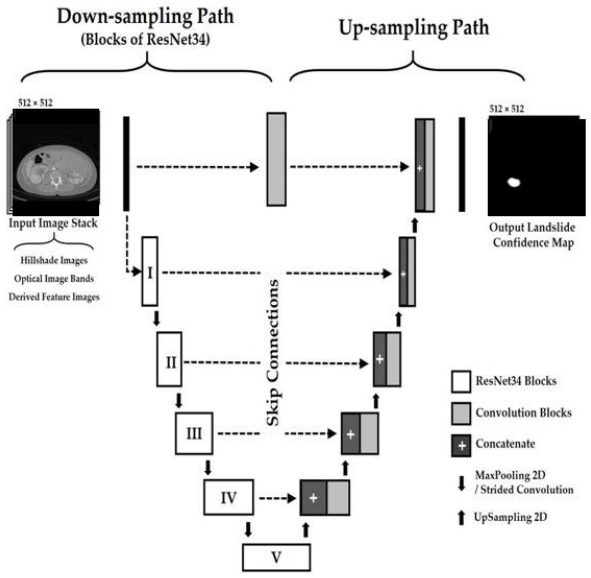

Figure 3: U-net architecture with ResNet34 blocks in the down-sampling path [4].

4.3 Model Training

For the loss function, we used BCEWithLogitsLoss, which is a commonly used loss function in segmentation tasks due to its ability to handle binary classification problems and its ease of use in combination with sigmoid activation functions. Furthermore, we utilized Adam as the optimizer with a learning rate of 0.001.

For training, we created dataloaders with a batch size of 64, and trained our model per 50 epochs. However, we only saved the best model regarding validation loss to avoid overfitting.

4.4 Postprocessing

One postprocessing technique we employed involved disregarding areas of predicted tumors that were outside of the kidneys. However, this approach did not yield a significant improvement in the results.

Finally, we construct a 3D tumour segment by combining the 2D segments.

5 Parallelization

In this project, we experimented by running each batch of data in parallel over multiple GPUs on the same node and across multiple nodes, each with one GPU. We utilized the PyTorch DataParallel library to leverage multiple GPUs on the same node and employed the DistributedDataParallel library with the NVIDIA Collective Communications Library (NCCL) backend for distributed learning across multiple nodes. All experiments were conducted on A100 GPUs within the Digital Research Alliance of Canada cluster.

A pivotal experiment is depicted in Figure 4, illustrating the impact of batch size on parallel processing. The results indicate that a smaller batch size (less than 32 in our case) provides minimal benefits in parallel processing. Conversely, with a larger batch size (32 or higher), we achieved approximately a 40% speedup using 2 GPUs and a 70% speedup using 4 GPUs.

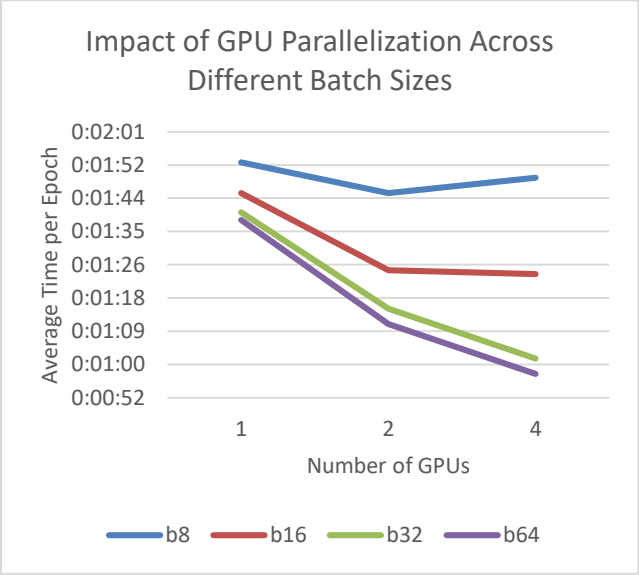

Figure 4: Time comparison for running the training process in parallel across different batch sizes on a small proportion of the dataset.

In the realm of distributed learning across multiple nodes, the choice of communication backend and network bandwidth significantly influences results. Figure y highlights performance variations across nodes, impacting overall efficiency.

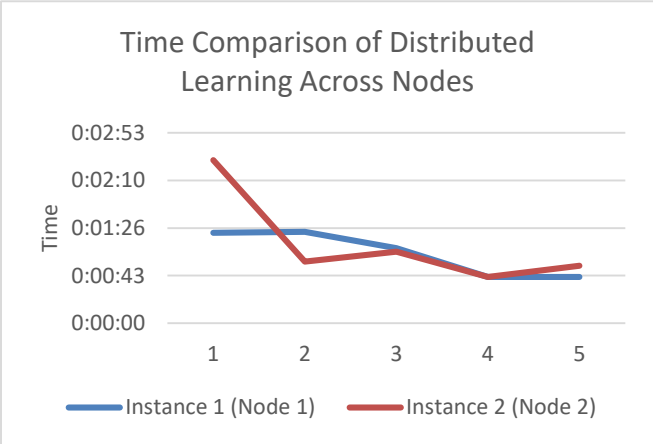

Figure 5: Time comparison of distributed learning across two nodes on a small proportion of the dataset.

In our final experiment, we conducted a 50-epoch training process with a batch size of 64. Data parallelization on the same node yielded an approximately 2.2 times speedup, while distributed learning over multiple nodes surprisingly achieved a 3.15 times speedup. This underscores that as the task size increases, greater benefits are derived from parallelization. The superior performance of distributed learning over multiple nodes may be attributed to the more effective implementation of the NCCL communication backend compared to the DataParallel library.

Configuration	Time
Single node, single GPU	14:37:06
Data Parallel, single node, 4 GPUs	6:43:44
Distributed Learning, 4 nodes, 1 GPU per node	4: 38:05

Table 1: Time comparison of training model for 50 epochs using three approach of single GPU, data parallel, and distributed learning

6 Results

6.1 Evaluation Metric

We used the dice coefficient metric to determine the overlap between the predicted segmentation mask and the ground truth mask.

$$\text{Dice Score} = \frac{2*|Y \cap Y'|}{|Y \cup Y'|} \quad (1)$$

Where Y is the ground truth and Y' is the model prediction.

6.2 Results on test set

We present the results in Table 2, which are based on the test set comprising 49 cases.

	Kidney	Tumor	cyst
Dice Score	0.937	0.788	0.907

Table 2: Dice scores for the kidney, tumor, and cyst segmentation of our model

7 Discussion and Conclusion

In this study, we used a 2D U-Net model to segment the kidney and tumors from CT images. Regarding the simplicity of 2D segmentation, our result is promising when compared with top performers in KiTS competition. However, there is a limitation to focusing on 2D slices, as it is harder to recognize the shape and edges of a tumor. Generally, the segmentation of the kidney is very promising, but the segmentation of tumors demonstrated insufficient accuracy, highlighting the need for further improvement.

In the following work, we did parallelization over multi-GPUs, where we achieved 3.15 times speedup through distributed learning over multiple nodes with the NCCL backend. Our study demonstrates that, particularly with large jobs and batch sizes, parallelization can significantly enhance the performance of deep learning models.

ACKNOWLEDGMENTS

This research was enabled in part by support provided by Dr. Ethan McDonald and the Digital Research Alliance of Canada (alliancecan.ca).

REFERENCES

[1] Ghislaine Scelo and Tricia L Larose. Epidemiology and risk factors for kidney cancer. Journal of Clinical Oncology, 36(36):3574, 2018.
[2] Pandey, J., & Syed, W. (2020). Renal Cancer.
[3] https://kits21.kits-challenge.org
[4] Prakash, N., Manconi, A., & Loew, S. (2020). Mapping landslides on EO data: Performance of deep learning models vs. traditional machine learning models. Remote Sensing, 12(3), 346.
[5] Lee, S. U., Chung, S. Y., & Park, R. H. (1990). A comparative performance study of several global thresholding techniques for segmentation. Computer Vision, Graphics, and Image Processing, 52(2), 171-190.
[6] Kohler, R. (1981). A segmentation system based on thresholding. Computer Graphics and Image Processing, 15(4), 319-338.
[7] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18 (pp. 234-241). Springer International Publishing.
[8] 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.
[9] Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., ... & He, K. (2017). Accurate, large minibatch sgd: Training imagenet in 1 hour. arXiv preprint arXiv:1706.02677.
[10] Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. T. P. (2016). On large-batch training for deep learning: Generalization gap and sharp minima. arXiv preprint arXiv:1609.04836.
[11] Mei, X., Liu, Z., Robson, P. M., Marinelli, B., Huang, M., Doshi, A., ... & Yang, Y. (2022). RadImageNet: an open radiologic deep learning research dataset for effective transfer learning. Radiology: Artificial Intelligence, 4(5), e210315.