DLMI Challenge report

Leopold Hebert- Stevens* LEOPOLD.HEBERT-STEVENS@STUDENT-CS.FR
Adrien Loizeau* ADRIEN.LOIZEAU@STUDENT-CS.FR

Editors: Under Review for MIDL 2023

Abstract

Radiation therapy plays a crucial role in cancer treatment, making accurate radiation dose prediction essential for designing effective and safe treatment plans. In this study, we present a deep learning model based on a modified UNet architecture to predict radiation doses using patient CT scans, segmentation masks of targeted organs, and maximum possible irradiation. Our model leverages BasicBlock modules in the encoder and decoder, which consist of convolutional layers with batch normalization and ReLU activation. We address overfitting through dropout, weight regularization, early stopping, and learning rate scheduling. Furthermore, we employ data augmentation and normalization techniques to improve the model's generalization ability. A pre-trained ResNet50 model was tested but did not yield significant improvements in accuracy, convergence, or speed. Finally our proposed model demonstrates promising results in predicting radiation doses for improved radiation therapy treatment plans and patient outcomes.

Keywords: Radiation therapy, Cancer treatment, Deep learning, UNet, ResNet50, Medical imaging, CT scans

1. Introduction

Radiation therapy is a widely used treatment for cancer that involves targeting cancerous tissues with gamma rays while preserving surrounding healthy tissues. The design of safe and effective radiation treatment plans requires simulation of the radiation dose prior to treatment. Currently, designing treatment plans can be slow and rely heavily on approximations. In this context, the development of deep learning models that accurately predict radiation doses using patient data such as CT scans and segmentation masks of organs has gained significant attention.

In this paper, we present our approach for training a deep learning model to predict radiation dose using a patient's CT, segmentation masks of the targeted organs, and maximum possible irradiation. Our approach was developed as part of a competition in which participants were tasked with accurately predicting radiation dose to improve the design of radiation therapy treatment plans and ultimately improve patient outcomes.

The accurate prediction of radiation dose is crucial for the design of safe and effective radiation therapy treatment plans. By developing a deep learning model that can predict radiation dose using patient CT scans and segmentation masks, we hope to improve the precision of radiation therapy treatment plans and ultimately improve patient outcomes.

* Contributed equally

2. Architecture and methodological components

2.1. Dataset

The dataset used in this competition was obtained from the Open-KBP challenge and was modified for our class into a 2D dataset. The dataset includes structural masks of the 10 organs involved in the treatment, a binary mask of where irradiation is allowed, ground-truth radiation doses, and CT scans of patients. Therefore the goal was to predict the ground-truth radiation doses based on the following inputs: a CT-scan, a binary mask of where irradiation is allowed, and 10 structural masks of the organs involved in the treatment. Since all of these inputs had a dimension of 128 by 128, we concatenated them along the depth dimension to form a 12-layer input.

2.2. Choice of model: UNet

In this paper, we present a deep learning model based on the UNet architecture for predicting radiation dose in cancer treatment. The choice of the UNet architecture is motivated by its proven success in medical image segmentation tasks. The UNet architecture is well-suited for this task of predicting a 128 by 128 dose because it is specifically designed for image segmentation tasks. The UNet has a contracting path, which captures context and reduces the resolution of the image, and an expansive path, which upsamples the image and recovers the spatial information. This allows the model to learn rich representations of the image at different scales, making it particularly effective for tasks where spatial information is important, such as in medical imaging.

Other models that could have been used include other convolutional neural network (CNN) architectures like VGG, ResNet, and DenseNet. However, these models are less efficient for this task because they are not specifically designed for image segmentation. VGG, for example, has a very deep architecture with many layers, making it computationally expensive and potentially prone to overfitting on small datasets like the one used in this competition. ResNet and DenseNet are also designed for image classification tasks, and while they may perform well on some segmentation tasks, they are not optimized for the task of predicting radiation doses based on CT scans and segmentation masks.

2.2.1. Our model

Our model is a modified version of the U-Net architecture, which has been shown to be effective in various medical image segmentation tasks. The U-Net architecture consists of an encoder and decoder, with skip connections between them. The encoder is a series of convolutional layers that downsample the image, while the decoder is a series of convolutional layers that upsample the image. The skip connections concatenate the output of the corresponding encoder and decoder layers to help preserve spatial information lost during downsampling.

Our modified version of the U-Net architecture uses a series of BasicBlock modules for the encoder and decoder. The BasicBlock module consists of two convolutional layers with batch normalization and ReLU activation. Each convolutional layer has a kernel size of 3 and a stride of 1, and the padding is set to maintain the same spatial dimensions. Dropout is applied to the output of each ReLU activation layer to prevent overfitting.

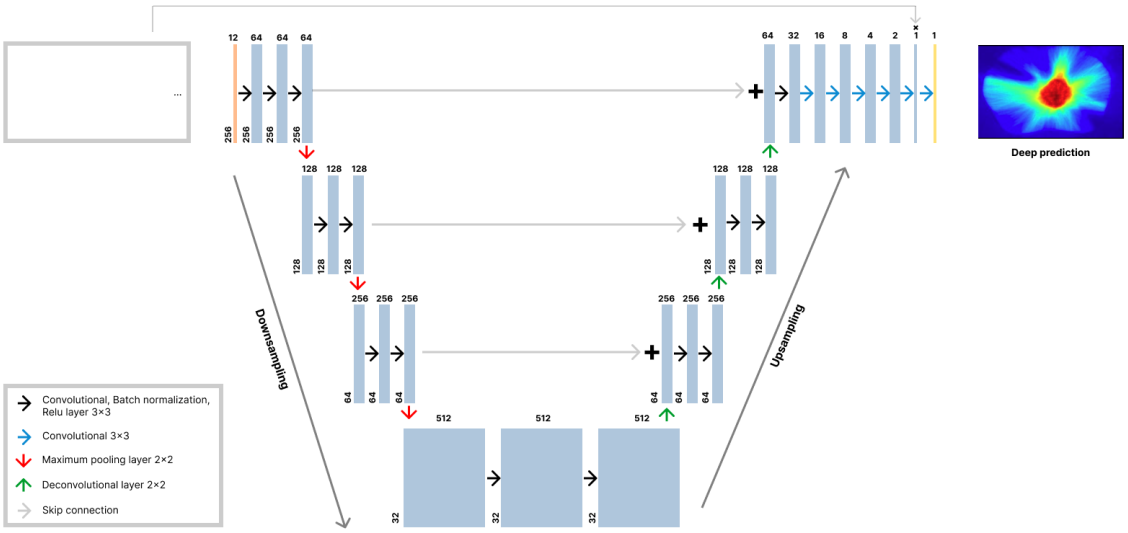

Figure 1: Schematic diagram of the U-net based dose prediction deep neural network (Dp-Net) architecture used for radiation therapy dose distribution prediction

The encoder part of our model consists of four BasicBlock modules followed by max pooling layers to downsample the feature maps. The decoder part of our model consists of four transpose convolution layers to upsample the feature maps also followed by four BasicBlock modules.

The final layers of our model consist of a series of convolutional layers that gradually reduce the number of channels, followed by a final convolutional layer that outputs a single-channel map of the predicted radiation dose. To ensure that the predicted dose is only calculated for areas where radiation is allowed, we multiply the output of the final layer by the binary mask of the region of interest.

3. Model tuning and comparison

3.1. Ablation study

In order to evaluate the impact of different components of our model, we conducted an ablation study.

First, we investigated the effect of increasing the number of layers in the bottom block of our model, from 512 to 1024. While this change led to a slight improvement in accuracy, it also increased the training time, and we found that the added complexity did not have a significant impact on the overall performance of the model maybe because a smaller model may be better able to capture the essential features of the data without capturing noise, and may generalize better to new data.

After experimenting with different learning rates, we found that the best performance was achieved with a learning rate of $5 * 10^{-4}$. It is worth noting that using a smaller

learning rate resulted in slower convergence and did not improve the final accuracy. On the other hand, using a larger learning rate caused the optimization algorithm to overshoot the optimal solution, leading to unstable training and poor results.

Additionally, we tested different batch sizes and found that a batch size of 32 was both faster and as accurate as a batch size of 16, 8 or less. This finding is consistent with previous research suggesting that larger batch sizes can lead to more stable and efficient training.

During our experiments, we found that ReLU activation function worked well for our model. However, we also explored other activation functions such as Leaky ReLU or ELU but we did not observe any improvements in performance with these activation functions. Therefore, we concluded that ReLU was the most effective activation function for our model.

Finally, we also tested different optimization algorithms to see if they had an impact on the performance of our model. Even though RMSprop and SGD are commonly used in deep learning, we found that the Adam optimizer provided the best results for our segmentation task as it is combinaison of the others.

3.2. Pre-processing

In addition to the UNet architecture and training strategies described earlier, we employed several data preprocessing techniques that improved our model's performance. One of the most important techniques was data augmentation, which helped increase the size and diversity of our training dataset. We used the Albumentations library [1] to perform the following image transformations:

- Gaussian noise and blurring are commonly used augmentations that help to smooth out images and make them more robust to noise.
- Elastic transformation is a deformation that stretches and compresses different parts of the image, making it more robust to geometric distortions.
- Rotation and vertical flipping are useful for training the model to recognize objects in different orientations.
- Grid distortion is a deformation that applies random displacements to the image grid, which helps the model to learn to handle more complex transformations.

These augmentations helped to generate a more varied set of images for our model to train on, which improved its generalization ability.

Moreover, to ensure that our model could learn effectively from the input data, we applied normalization to our CT scan images. CT scan images often have pixel values that range from 0 to 4095, which can make it difficult for the model to effectively learn from the input data without normalization. By normalizing the pixel values to have a mean of zero and standard deviation of one, we made it easier for the model to process the input data and mitigated any potential bias towards certain pixel values.

3.3. Overfitting issue

After tuning the model hyperparameters, the training loss (MAE) decreased significantly while the validation loss remained stagnant. For instance, after 30 epochs, the training loss

reached around 0.25, while the validation loss remained at approximately 0.5, indicating a severe overfitting problem. In response, we implemented several strategies to address this problem, including dropout, weight regularization, early stopping, and learning rate scheduling.

- Dropout: In the basic blocks of the model, we implemented a dropout of p=0.1 after each ReLU function. By randomly drops out some of the activations during training, which can help prevent overfitting by forcing the model to learn more robust representations.
- Weight regularization: We added l2 regularization with a coefficient of $5 * 10^{-4}$. This technique adds a penalty term to the loss function that discourages the model from assigning too much importance to any one feature. This can help prevent overfitting by encouraging the model to learn simpler, more generalizable representations.
- Early stopping: We added early stopping, which involves monitoring the validation loss during training and stopping the training process when the validation loss stops improving. This can help prevent overfitting by ensuring that the model is not trained for too long and does not become overly specialized to the training data.
- Learning rate scheduler: We added a step scheduler with a step size of 15 and a factor of 0.5. Every 15 epochs, the learning rate is reduced by a factor of 0.5 to help the model converge more smoothly and avoid getting stuck in local minima.

3.4. Pretrained model

Due to the complexity of the model and the data augmentation techniques used, each epoch of our UNet took around 2 to 3 minutes to compute on Google Colab GPUs. To reduce the training time, we experimented with using a pre-trained model as the encoder part of our UNet. Pre-trained models can potentially be faster to train because they have already learned general features from large datasets, which can be applied to other tasks. This means that they may require less training data and computation time to achieve good performance compared to our UNet model trained from scratch. We used a pre-trained ResNet50 model [2], which was trained on the RadImageNet dataset containing over 100,000 radiology images from different modalities and diagnoses, as the encoder part of our UNet. By using this pre-trained model, we aimed to take advantage of the features learned from radiology images to improve the performance of our model. However, despite freezing and fine-tuning the pre-trained model, we did not observe any improvement in accuracy, convergence, or speed compared to our original model.

One technique that could have been effective was to utilize transfer learning by using the pre-trained ResNet50 model for the CT scan part and a separate encoder for the masks. These two encoders could then be combined before the decoder. This approach could have leveraged the pre-existing knowledge of the ResNet50 model trained on radiology images while allowing the basic encoder trained from scratch to learn on the masks, improving performance on a new task. However, due to the high complexity this tasks represents it was not implemented.

References

[1] Alexander Buslaev et al. "Albumentations: Fast and Flexible Image Augmentations". In: *Information* 11.2 (2020). ISSN: 2078-2489. DOI: 10.3390/info11020125. URL: https://www.mdpi.com/2078-2489/11/2/125.

[2] Xueyan Mei et al. "RadImageNet: An Open Radiologic Deep Learning Research Dataset for Effective Transfer Learning". In: *Radiology: Artificial Intelligence* 0.ja (0), e210315. DOI: 10.1148/ryai.210315. eprint: https://doi.org/10.1148/ryai.210315. URL: https://doi.org/10.1148/ryai.210315.