smartMRI framework for segmentation of MR images using multiple deep learning methods

Cristina-Petruta Manoila[1], Alexe Ciurea[2], Felix Albu[3]
Affiliation 1: University of Warwick, Department of Computer Science, United Kingdom, cristina.manoila@warwick.ac.uk
Affiliation 2: Politehnica University of Bucharest, Romania, alexe.ciurea@gmail.com
Affiliation 3: Valahia University of Targoviste, Romania, felix.albu@valahia.ro

***Abstract*—MRI analysis frameworks are practical tools for accelerating the analysis by clinical researchers. However, over time, the focus has shifted to creating rigid frameworks. This paper presents a new framework that allows researchers to run different deep learning models based on predefined parameters suitable for automatically delineating the region of interest from magnetic resonance (MR) images of the knee joint. In addition, we present different deep learning methods for the automated segmentation of knee bones trained using data from the SKI10 challenge, concentrating on a convolutional neural network (CNN), which has proven promising potential in musculoskeletal imaging applications. We also propose a novel method that weighs the average of the surrounding pixels when the image is downsampled within a CNN.**

Keywords—MRI; framework; segmentation; knee joint; deep learning; Gaussian pooling.

I. INTRODUCTION

Segmentation of the knee joint is essential to identify degenerative joint diseases such as osteoarthritis (OA). However, most magnetic resonance imaging (MRI) studies focus only on an end-to-end deep learning framework using a single network. For instance, Kam et al. [1] developed a novel end-to-end framework to detect noise components in resting-state functional MRI, a technique for functional brain studies. They used only one database without analyzing other deep learning (DL) techniques. In addition, there is no flexibility in the use of different parameters or CNNs. Like Kam, Lee et al. [2] developed a new segmentation method for knee joints, but using this time two CNNs, (i) a bone cartilage complex, where they incorporated all four classes into two classes (tibia and tibia cartilage in one class, and femur and femur cartilage in the second class), and (ii) bone segmentation of the femur and tibia only. Their approach does not cover the analysis of multiple CNNs or the use of different parameters.

Under these conditions, we developed a novel framework for magnetic resonance (MR) images called smartMRI, inspired by our previous work [3] on the classification of EEG signals. This approach allows clinical researchers to swiftly develop their own CNNs and test different parameters. In this paper, we present the entire framework, such as the ability to use (i) any training and validation parameters and (ii) any deep learning methods used to identify the best performance of our approaches. In Table I, we present a comparison of the existing Python MRI deep-learning frameworks and our proposal.

TABLE I. EXISTING FRAMEWORKS

Author	*Testing different methods*	*Flexibility*
Kam et al. [1]	No	No
Lee et al. [2]	No	No
Ciurea et al. [3]	Yes	No
This work	Yes	Yes

Furthermore, we propose a novel CNN, entitled Pseudo3D GU-Net (Gaussian U-Net), which gives more weight to the central pixel value so that we can control the blurring of the image in the pooling layer by retaining more spatial information from the three slices. In contrast to Vu et al. [4], who developed a Pseudo-3D using an odd number of slices (3, 5, 7, 9, 11, 13) to obtain a 2D feature map that was fed straight into a 2D network, we propose a Pseudo3D U-Net with a number of slices of 3 by employing a novel downsampling method using a Gaussian filter.

The remaining sections of this paper are Section II, which describes the MRI framework; Section III, which explains the novel Pseudo3D GU-Net approach; Section IV presents the results of different CNNs on knee joint segmentation; and Section V concludes our work and provides future directions.

II. PROPOSED FRAMEWORK

SmartMRI is a framework that automates training of multiple DL models using Python 3.7 [5].

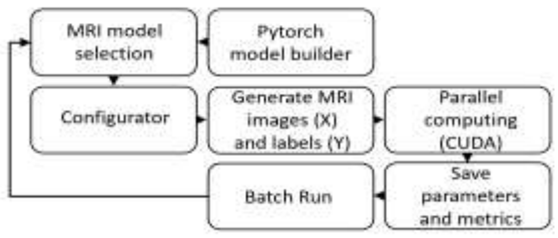
MRI model selection
Pytorch model builder
Configurator
Generate MRI images (X) and labels (Y)
Parallel computing (CUDA)
Batch Run
Save parameters and metrics

Fig. 1. Proposed MRI framework

The structure of this framework is shown in figure 1 as follows:

- *Pytorch model builder.* In this file, we build different models using the Pytorch library [6] so that we can take advantage of parallel computing on a graphical processing unit (GPU) to accelerate deep learning for computer vision. We employed the following CNNs for comparison: 2D U-Net [7] and Pseudo3D U-Net with maximum filters of 128; Pseudo3D Dense U-Nets with maximum filters of 512 and 868, respectively; and a novel Pseudo3D GU-Net with a maximum number of filters of 128 and 868, respectively.

- *MRI model selection* customizes different models from the Pytorch model builder file.

- *Configurator*. This block specifies the class selection for up to five classes (ground truth, tibia bone, femur bone, tibial cartilage, and femoral cartilage) and training and validation sets. The model names, loss functions, batch length, patience, validation methods (e.g., 80/20 split and k-fold), are configurable parameters in the training and validation settings.

- *Generate MRI images (X) and labels (Y).* This block normalizes the MRI images (X) between 0 and 1 and creates labels (Y) dependent on the selected number of classes.

- *Parallel computing (CUDA)* [8]. The model, along with the training and validation parameters, was loaded into the GPU. During training, the IDE console was updated with the training and validation parameter performance.

- *Save parameters and metrics.* The weights of the best epoch are saved when early stopping regularization is not enabled; otherwise, they are not saved to the disk.

-*Batch Run* sets the next configuration from the "Configurator" block and saves the training and validation settings along with the MRI metrics for later analysis by clinical researchers.

III. PSEUDO3D GU-NET

In this section, we introduce a novel deep learning approach that provides more weight to the current pixel position instead of calculating the maximum value in each patch of each feature map, and the results are downsampled. We used gaussian filtering [9], which is essential in image processing for reducing noise and blurring regions of an image and is defined by:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (1)$$

where x and y are the row and column dimensions, and σ the blur factor. To apply our GaussPool approach to the input image, we returned a kernel gauss of 3 × 3 with a blur factor of 1. First, our GaussPool function unfolds the image using the same kernel gauss size by generating the patches and finally multiplying them by the kernel gauss (figure 2).

The The Pseudo3D GU-Net is presented in figure 3, where we used three sequential cropped slices of 256 × 256 as input images, followed by repeated 3 × 3 convolutions, batch normalization, and a rectified linear unit (ReLu) activation function to replace all negative values in the feature map by zero, and all positive values retaining the same value, followed by the 3 × 3 Gaussian pooling presented above. At each downsampling step, we doubled the number of feature channels, except for the fourth and fifth steps, where we kept the same feature maps of 868 to avoid filling the GPU memory to its maximum capacity. We then upsampled the image by applying a 2D transposed convolution operator and concatenated the central slice from the encoder with the 2D upsampled image. The output of the network is a 2D image with three classes (ground truth, femur bone, and tibia bone).

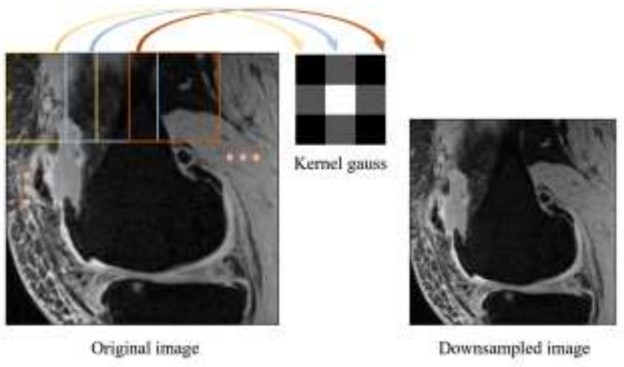

Fig. 2. Kernel gauss approach. (i) On the left side: original image, where the colored squares are patches in both directions (x,y). (ii) In the middle, the kernel Gaussian filter was multiplied by each patch. On the right: the downsampled image generated by (i) and (ii)

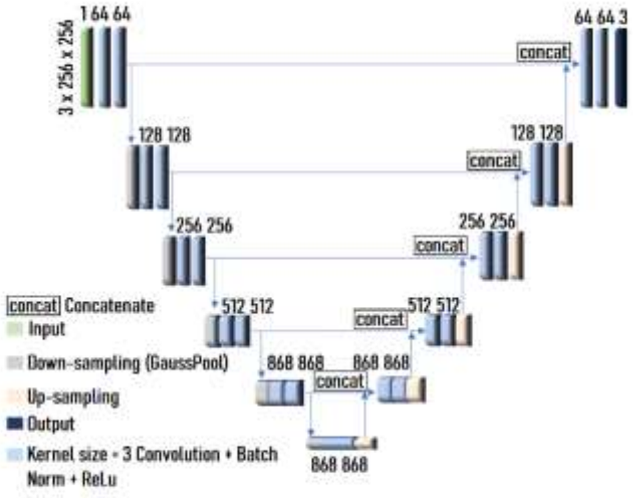

Fig. 3. The proposed method: Pseudo3D GU-Net

IV. RESULTS

In this section, we present the functionality of our framework by comparing different convolution neural networks using different feature maps to the network bottleneck.

A. Data set

The SKI10 data set is a publicly available MRI dataset consisting of 60 training and 40 validation MR images. The data set was mostly acquired at 1.5T, some at 3T and 1T with an in-plane resolution of 0.39 x 0.39 mm, approximately 300x356 matrix dimension (actual matrix varies with the size of the knee joint), a slice thickness of 1 mm, and approximately 100 image slices in the sagittal plane. The majority of images used a T1-weighted sequence, but some of them were also obtained with

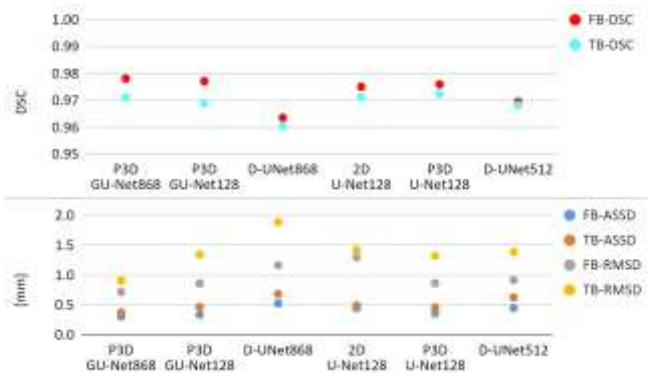

Fig. 4. Model comparison. Top: DSC average, higher is better. Bottom: Symmetric Surface Distances average, lower is better.

T2-weighting. Clinicians manually segmented the multi-class mask for the knee joint for each image data set with the following values: 0 for the background, 1 for the femur, 2 for the femoral cartilage, 3 for the tibia, and 4 for the tibial cartilage. However, in our paper, we analyzed three classes (background, tibia, and femur).

B. *Training set-up*

We trained our models using the sagittal plane and employed the Adam optimizer with a learning rate of 0.0001 using early stopping with patience of three epochs to prevent overfitting of the model. The maximum number of epochs was set to 100. The loss function mostly used was Focal Tversky Loss (FTL) [10] focusing on hard examples, such as small regions of interest (ROI) with the help of gamma, which was set to 0.7. Furthermore, we applied the softmax activation function at the output of the models.

C. *Evaluation metrics*

To evaluate our models, we split the data into 60 images for training and 40 images for validation for each patient, and employed several metrics used for the segmentation accuracy of the MRI volumes. First, the Dice similarity coefficient (DSC) represents the spatial overlap of the ground truth (GT) segmentation and predicted segmentation by the total size of the two objects. Second is the average symmetric surface distance (ASSD), where we compare the average closest distance between the outline of the predicted values and the outline of the GT values, and vice versa. Finally, we used the root mean square symmetric surface distance (RMSD) to consider the squared distances between the two sets of boundaries; a value of 0 corresponds to a perfect overlap between the two segmentations. ASSD and RMSD are expressed in millimeters, as shown in figure 4.

D. *CNNs employed*

The convolutional neural networks built in this work are as follows: (i) a 2D U-Net with maximum filters of 128 (2D U-Net128), (ii) a pseudo3D U-Net with maximum filters of 128 (P3D U-Net128); (iii) dense U-Net, where we use connections between layers to increase the number of feature channels with maximum filters of 512 and 868, respectively, and our approach uses a Pseudo3D Gaussian U-Net of 868 and 128 maximum filters (P3D GU-Net868 and P3D GU-Net128).

Fig. 5 shows the segmentation of the best networks shown in figure 4. We observed that P3D GU-Net128 and P3D U-Net128 could not accurately predict the femur bone (FB) and tibia bone (TB) marked with red and blue, respectively. On the other hand, P3D GU-Net868, with maximum feature maps of 868, better captures global image information and generates less ambiguity in terms of the localization of small ROIs.

In Table II, we compare our results with the current state-of-the-art, where we highlight the DSC average and mean standard deviation (STD) of the femur bone (FB) and tibia bone (TB) for 40 patients. In addition, we show the symmetric surface distance averages for ASSD, RMSD, and mean STD in Table III.

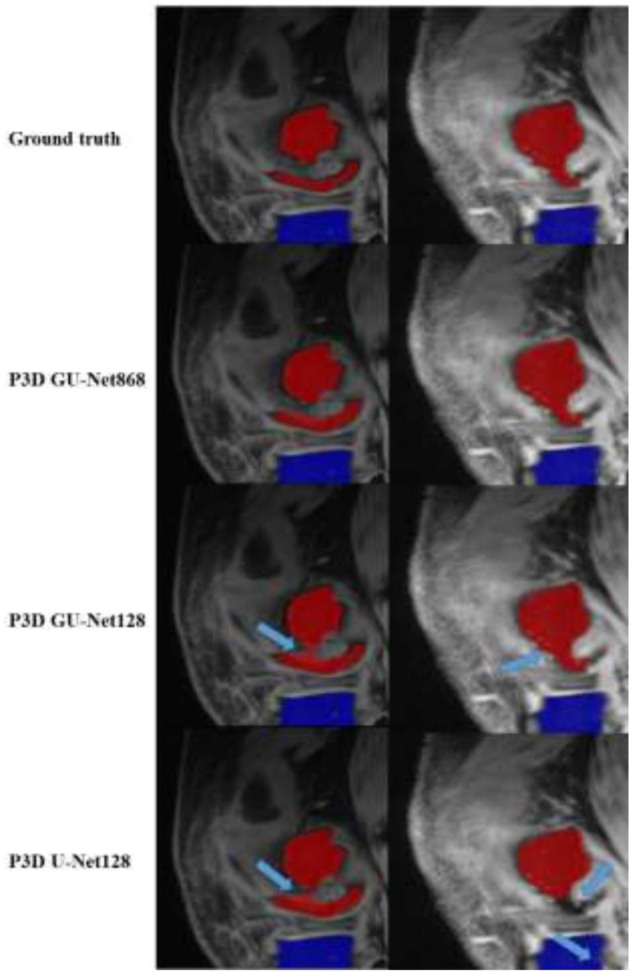

Fig. 5. CNNs segmentation results; FB and TB are delineated with red and blue, respectively.

TABLE II. DSC RESULTS AND COMPARISON

Methods 60 training/ 40 validation	***DSC (STD)***	
	FB	***TB***
Lee et al [2]	97.3 (1.9)	84.4 (4.1)
P3D U-Net128	97.5 (0.008)	97.1 (0.030)
P3D GU-Net128	97.6 (0.009)	96.8 (0.025)
P3D GU-Net868	**97.8 (0.009)**	**97.1 (0.025)**

TABLE III. SURFACE DISTANCE RESULTS AND COMPARISON

Methods* *60 training/ 40 validation	***Average***			
	FB-ASSD	***TB-ASSD***	***FB-RMSD***	***TB-RMSD***
Lee et al [2]	0.58 (0.45)	0.47 (0.32)	1.73 (0.85)	1.53 (1.05)
P3D U-Net128	0.35 (0.20)	0.46 (0.37)	0.87 (0.82)	1.32 (1.21)
P3D GU-Net128	0.34 (0.18)	0.47 (0.31)	0.85 (0.64)	1.33 (1.18)
P3D GU-Net868	**0.30 (0.16)**	**0.37 (0.30)**	**0.72 (0.58)**	**0.90 (1.01)**

E. Results Discussion

The presented convolutional neural networks showed promising results. As shown in figure 4, Pseudo3D GU-Net868 outperformed the other models. For this model, ASSD and RMSD have lower mean standard deviations, which means that data are clustered around the mean and not spread out (Table III). Furthermore, the average of the ASSD of 0.30(0.16) mm for FB and 0.37(0.30) mm for TB, respectively indicates that the average of all the distances from points of the boundary of the predicted region to the boundary of the ground truth (target) and vice versa are close to each other compared to other networks employed. Lower RMSD scores of 0.72(0.58) mm for FB and 0.90(1.01) mm for TB, respectively indicate how concentrated the data is around the line of best fit and the Pseudo3D GU-Net868 is the one that fits the best.

P3D GU-Net128 and P3D U-Net128 had lower performance than Pseudo3D GU-Net868 because of the lack of distinctive features, making it harder to recognize patterns in images. In contrast, P3D GU-Net868 performs better than the rest of the CNNs due to increasing details in the image and preserving more information from the neighbors.

V. CONCLUSIONS

We compared different CNNs employing our smartMRI framework to assess the performance of each network tested. In addition, we propose a novel deep learning method called Pseudo3D GU-Net, which outperforms current state-of-the-art methods, especially in terms of ASSD and RMSD metrics. In this approach, we replaced each image pixel with a weighted average of the neighbouring pixels by applying a Gaussian filter of 3 × 3. In addition, we feed a sub-volume of 3 sequential slices extracted from the whole volume to output a 2D image, retaining more detail of the image with minimal computational cost being more efficient than a fully 3D CNN. In terms of future work, we propose to add more parameters in the configurator and use a similar approach as GaussPool but using median pooling. Our models were tested on 3 classes to find the best approach and validate the proposed Pseudo3D GU-Net. We also plan to use different data sets and to train to detect the remaining 2 classes, femoral and tibial cartilages, to diagnose patients prone to knee joint diseases such as osteoarthritis. Furthermore, we plan to test our approach on unseen data together with clinicians and try the presented approach on other applications such as howling detection on spectrograms of hearing aids signals.

REFERENCES

[1] T. E. Kam *et al.*, "A Deep Learning Framework for Noise Component Detection from Resting-State Functional MRI," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2019, vol. 11766 LNCS. doi: 10.1007/978-3-030-32248-9_84.

[2] H. Lee, H. Hong, and J. Kim, "BCD-NET: A novel method for cartilage segmentation of knee MRI via deep segmentation networks with bone-cartilage-complex modeling," in *Proceedings - International Symposium on Biomedical Imaging*, 2018, vol. 2018-April. doi: 10.1109/ISBI.2018.8363866.

[3] A. Ciurea, C. P. Manoila, and B. Ionescu, "SmartEEG: An End-to-End Framework for the Analysis and Classification of EEG signals," in Proc. of IEEE International Conference on E-Health and Bioengineering - EHB, Nov. 2021. doi: 10.1109/EHB52898.2021.9657753.

[4] M. H. Vu, G. Grimbergen, T. Nyholm, and T. Löfstedt, "Evaluation of multislice inputs to convolutional neural networks for medical image segmentation," *Med Phys*, vol. 47, no. 12, 2020, doi: 10.1002/mp.14391.

[5] G. van Rossum *et al., Python 3 Reference Manual*, vol. 585, no. 7825. 2009.

[6] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019, vol. 32.

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015, vol. 9351. doi: 10.1007/978-3-319-24574-4_28.

[8] S. Cook, *CUDA Programming: A Developer's Guide to Parallel Computing with GPUs*, no. 1. 2013.

[9] W. M. Wells, "Efficient Synthesis of Gaussian Filters by Cascaded Uniform Filters," *IEEE Trans Pattern Anal Mach Intell*, vol. PAMI-8, no. 2, 1986, doi: 10.1109/TPAMI.1986.4767776.

[10] S. Jadon, "A survey of loss functions for semantic segmentation," in Proc. IEEE Conf. Comput. Intell. Bioinf. Comput. Biol. (CIBCB), Oct. 2020, pp. 1–7, doi: 10.1109/CIBCB48159.2020.9277638.