COBRA: Cpu-Only aBdominal oRgan segmentAtion
A small, fast & accurate 3D-CNN

Edward G. A. Henderson*
edward.henderson@postgrad.manchester.ac.uk

Dónal M. McSweeney*
donal.mcsweeney@postgrad.manchester.ac.uk

Andrew Green
andrew.green-2@manchester.ac.uk

Division of Cancer Sciences, The University of Manchester
Radiotherapy Related Research, The Christie NHS Foundation Trust
Manchester, UK

*authors contributed equally

Abstract

Abdominal organ segmentation is a difficult and time-consuming task. To reduce the burden on clinical experts, fully-automated methods are highly desirable. Current approaches are dominated by Convolutional Neural Networks (CNNs) however the computational requirements and the need for large data sets limit their application in practice.

By implementing a small and efficient custom 3D CNN, compiling the trained model and optimizing the computational graph: our approach produces high accuracy segmentations (Dice Similarity Coefficient (%): Liver: 97.3 ± 1.3*, Kidneys:* 94.8 ± 3.6*, Spleen:* 96.4 ± 3.0*, Pancreas:* 80.9 ± 10.1*) at a rate of 1.6 seconds per image.*

Crucially, we are able to perform segmentation inference solely on CPU (no GPU required), thereby facilitating easy and widespread deployment of the model without specialist hardware.

1. Introduction

Volumetric image segmentation is a time-consuming and often complicated task that requires medical expertise to produce high-quality, reliable delineations. A number of automated approaches have been proposed to free experts from this tedious task and consequently decrease annotation cost. However, variability in acquisition protocols, patient anatomy and the presence of pathologies make this a difficult task to automate. Current state-of-the-art approaches require large and diverse data sets to generalise to unseen examples. The complexity of the resulting models and the size of the CT volumes incur large computational costs and limit applications to centres with adequate computational resources.

We aim to improve accessibility to these methods by removing the need for expensive, specialist hardware (GPUs) and instead produce models that can perform quick inference entirely on CPU. To reduce the computational cost of image processing, high-resolution 3D CT scans are all downsampled to the same size. Our model architecture is inspired by the 3D U-Net presented in [1] with a few notable modifications to reduce model size and the number of FLOPs (discussed in section 2). Finally, we use the Open Neural Network Exchange (ONNX) [2] to compile the trained model, at which point we apply further optimisation to the computational graph.

As a result, we are able to deploy a small (1.7 Mb) and fast model (1.6 seconds/image on CPU) that produces high-quality 3D segmentations of abdominal organs.

2. Method

Our solution for this challenge is based on a single CNN model which performs segmentation inference on an entire downsampled CT scan. We developed a custom 3D CNN for this challenge which is illustrated in Figure 1.

2.1. Preprocessing

When training, we downsample all CT scans and gold standard segmentations to a standardised resolution of $96 \times 192 \times 192$. The CT scans are downsampled using 3^{rd} order spline interpolation, whereas nearest-neighbour downsampling is used for the corresponding gold standard segmentations. No cropping is applied prior to training. These decisions were made to promote segmentation scale invariance within our model.

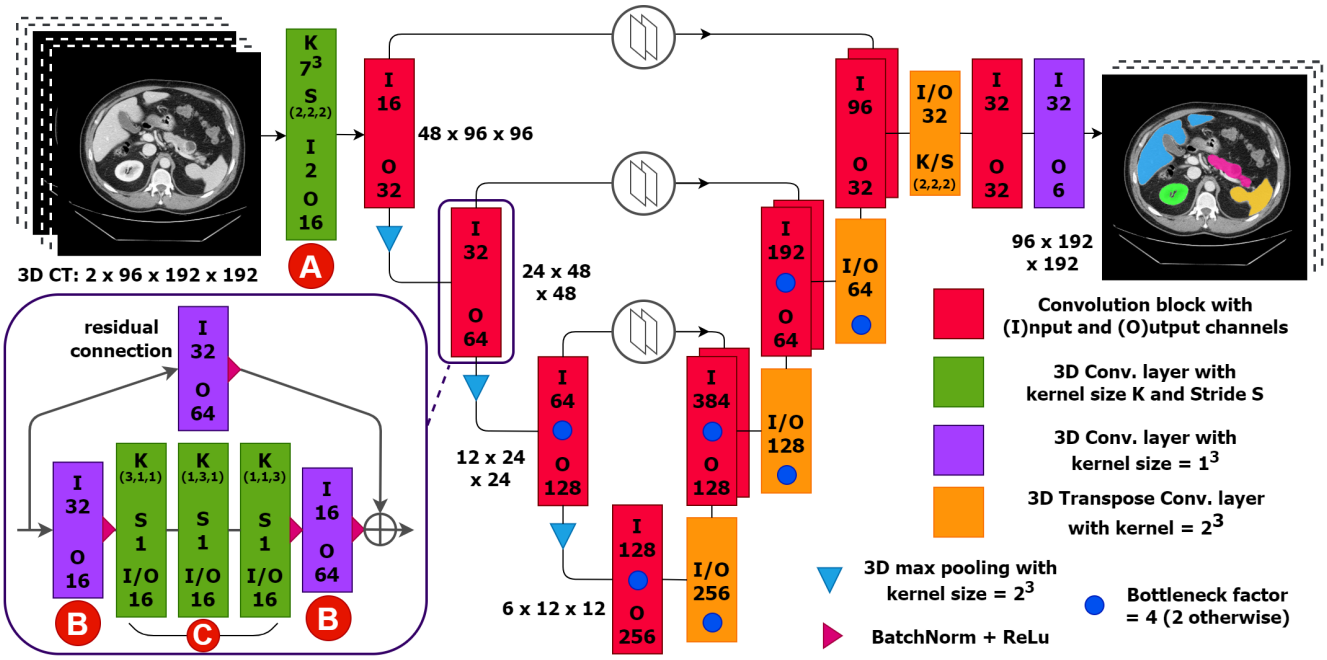

Figure 1. A diagram of our custom 3D CNN architecture. Our network is influenced by a standard 3D UNet [1] with added ResNet-like residual connections [3]. In order to reduce the model size and computational load we introduce three components: a YOLO-inspired [4] 7x7x7 input convolution with stride 2 to quickly reduce the size of the input image whilst preserving a large field of view (see A); bottleneck structures to reduce the number of kernels required (B); and asymmetric factorisation of convolution layers (C).

The liver, kidneys, spleen and pancreas are all soft tissue structures. Therefore, we chose to normalise the CT scans prior to training using windowing (grey-level mapping). By using windowing, we can enhance the contrast of these soft tissue structures whilst also mapping the voxel intensities onto the range $[0, 1]$ to improve learning stability. The CT scans are normalised in two separate contrast channels which are concatenated (see the input in Figure 1). This improved our segmentation performance. In channel one, we use a contrast setting which is used to view general soft tissue structures in the abdominal area (W400, L50). In the second channel we apply a tighter window (W100, L60) in an attempt to increase the contrast of the pancreas.

To aid in the determination of organ boundaries, we split the background label (0) in two: "air" and "body"; this is done by thresholding at -200 HU and applying binary closing and hole filling operations to the resulting mask. Air retains the original background label 0, and body becomes label 1, with all other labels shifted accordingly.

2.2. Proposed Method

In Figure 1 we show our custom 3D CNN designed for this challenge. The architecture is inspired by the classic 3D UNet [1] with added residual connections [3]. Residual connections are now a standard addition to most modern 3D UNet CNNs and improve gradient propagation in the training process.

Three key features of our model, which reduce the model size (number of parameters) and computational complexity (number of FLOPs), are introduced below.

First, a YOLO-inspired [4] input layer is added, consisting of a 7x7x7 convolution with stride of 2. In Figure 1 we mark this particular layer 'A'. This input layer quickly reduces the size of the volume being processed by the network, while preserving a wide field-of-view. By reducing the size of the CT scan being processed by the network, segmentation inference is accelerated.

To compress our model, we implemented bottleneck structures as used by He et al. for their deeper ResNet architectures [3]. Bottlenecks are introduced to reduce and then restore the number of kernels in convolutional layers in certain portions of the model. By compressing the number kernels of convolutional layers deep in the UNet architecture, we force our model to learn better feature representations whilst also reducing its size and computational complexity.

Bottlenecks are constructed by sandwiching existing convolutional layers with 1x1x1 convolutions which first reduce and then restore the number of kernels. An example bottleneck structure is labelled 'B' Figure 1. In our model we implement bottlenecks around all the traditional encoder and decoder double convolution layers (shown in red in Figure 1). In addition, all 3D transpose convolutions are similarly bottlenecked. For each bottleneck, we define a "bottleneck factor" which determines the multiplicative reduction

Table 1. Data splits of FLARE2021.

Data Split	Center	Phase	# Num.
Training (361 cases)	The National Institutes of Health Clinical Center	portal venous phase	80
	Memorial Sloan Kettering Cancer Center	portal venous phase	281
Validation (50 cases)	Memorial Sloan Kettering Cancer Center	portal venous phase	5
	University of Minnesota	late arterial phase	25
	7 Medical Centers	various phases	20
Testing (100 cases)	Memorial Sloan Kettering Cancer Center	portal venous phase	5
	University of Minnesota	late arterial phase	25
	7 Medical Centers	various phases	20
	Nanjing University	various phases	50

in kernels required. By default, we use a bottleneck factor = 2, except in blocks marked with a small blue dot in figure 1 where we use a bottleneck factor = 4. This higher level of kernel compression is performed in the widest layers of the CNN to further reduce model size and accelerate inference.

To further compress the size of the model and reduce the computational complexity we applied 3D asymmetric factorisation to the convolutional layers of our model. Factorisation of convolutional kernels was introduced by Szegedy et al. for their Inception v2 architecture [5] and has been more recently applied to 3D data by Yang et al. for action recognition in consecutive video frames [6]. Asymmetric factorisation simulates a large 3D kernel using a series of 1D kernels. For example, we factorise all 3x3x3 convolution kernels into a series of layers with 3x1x1, 1x3x1 and 1x1x3 kernels. A factorised 3x3x3 kernel reduces the number of parameters by up to a third compared to a standard layer. Additionally, the number of FLOPs required to perform factorised layers is significantly lower. For our final model, we applied asymmetric factorisation to all convolutional layers with 3x3x3 kernels and the YOLO-inspired 7x7x7 convolutional layer. Please refer to the section marked 'C' in Figure 1 which shows the layers resulting from asymmetric factorisation.

As a result, our custom 3D CNN contains just 436,982 parameters and requires 48 GFLOPs to segment a $96 \times 192 \times 192$ CT volume.

2.3. Post-processing

Our CNN model outputs raw segmentation predictions at a resolution of $96 \times 192 \times 192$. We apply *argmax* to obtain the hard predictions before upsampling the segmentations to the original CT scan dimensions using nearest-neighbour upsampling.

3. Dataset and Evaluation Metrics

3.1. Dataset

- The dataset used for FLARE2021 is adapted from MSD [7] (Liver [8], Spleen, Pancreas), NIH Pancreas [9, 10, 11], KiTS [12, 13], and Nanjing University under the license permission. For more detail information of the dataset, please refer to the challenge website and [14].

- Details of training / validation / testing splits:
  The total number of cases is 511. An approximate 70%/10%/20% train/validation/testing split is employed resulting in 361 training cases, 50 validation cases, and 100 testing cases. Detailed information is presented in Table 1.

At this stage we only have access to the training subset of images and their corresponding gold standard segmentations. We present a 5-fold cross validation of our model for this subset in anticipation of evaluation of our model with the validation and testing subsets.

3.2. Evaluation Metrics

- Dice Similarity Coefficient (DSC)
- Normalized Surface Distance (NSD) with tolerance of 1mm
- Running time: $\sim$ 1.6 seconds per CT scan
- Maximum used GPU memory - 0 Mb

4. Implementation Details

4.1. Environments and requirements

The environments and requirements of our method are shown in Table 2.

4.2. Training protocols

The training protocols we used to train our custom 3D CNN are shown in Table 3. We conducted a five-fold cross validation using the "Training" subset outlined in section 3.1 containing 361 CT scans. For the cross-validation, each set of training, validation and test folds contained 252, 36 and 73 images respectively.

Table 2. Environments and requirements.

Windows/Ubuntu version	Ubuntu 20.04.2 LTS
CPU	AMD Ryzen 9 3950X 16-Core Processor
RAM	64GB
GPU	NVidia GeForce RTX 3090
CUDA version	11.1
Programming language	Python 3.8
Deep learning framework	Pytorch (Torch 1.8.1, torchvision 0.9.1)
Specification of dependencies	SimpleITK (2.02), onnx (1.9.0), onnxruntime (1.8.1), numba (0.53.1), numpy (1.21.1)
(Optional) code is publicly available at	https://github.com/rrr-uom-projects/FLARE21 Docker image: afgreen/flare21:latest (on dockerhub)

Table 3. Training protocols.

Data augmentation methods	Shifting (± 4 voxels, $\sim$ 10mm), rotations (in-plane: $\pm 10°$), scaling (80% - 120%)
Initialization of the network	"He" normal initialization [15]
Patch sampling strategy	None - Full CT downsampled and used as input
Batch size	4
Patch size	$96 \times 192 \times 192$
Maximum epochs	1000
Optimizer	Adam
Initial learning rate	0.001
Learning rate decay schedule	Reduce LR on plateau with patience=75 epochs
Stopping criteria, and optimal model selection criteria	Early stopping when the validation loss does not improve for 175 epochs. Optimal model is chosen based on best validation loss.
Loss function	Weighted multi-class soft Dice
Training time	10-15 hours

4.3. Testing protocols

In the testing phase, inference is performed on the entire CT volume by downsampling the input image to a size of $96 \times 192 \times 192$ with 3rd order spline interpolation. A Gaussian kernel was applied in-plane to prevent aliasing artefacts when downsampling. As in the training phase, we do not crop images prior to inference.

We use ONNX [2] to compile the trained model thereby decreasing model size - enabling inference on CPU - and increasing inference speed. At this point, a number of graph level transformations are applied to improve model performance further. These include: constant folding, redundant node elimination and node fusion. This led to a small but noticeable improvement in inference speed.

We experimented with dynamic and static quantization. Though both reduced model size (1.7Mb $\rightarrow$ 614 Kb), the former increased inference time by an order of magnitude while the latter dramatically reduced model performance. Therefore, we chose not to apply weight quantization. Quantization-aware training may be an alternative for future work.

Following inference, *argmax* is applied to model predictions to convert them to multi-class masks then, they are upsampled to the original scan dimensions with nearest neighbour interpolation.

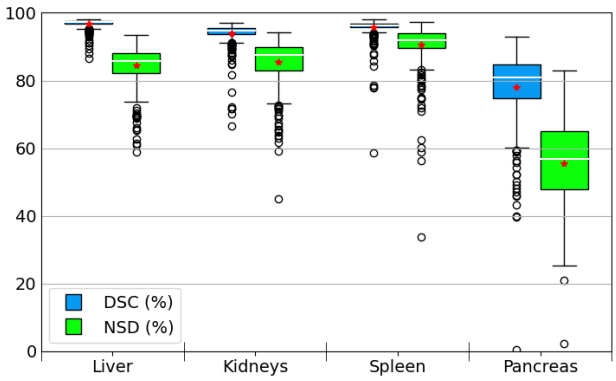

Figure 2. Boxplots of the organ segmentation results (DSC and NSD) of the 5-fold cross validation. The white line shoes the median and red star the mean.

5. Results

5.1. Quantitative results for 5-fold cross validation.

The provided results are based on the 5-fold cross-validation results and validation cases. Table 4 illustrates the results of 5-fold cross-validation. Figure 2 is the corresponding boxplot of organ segmentation performance. While high DSC and NSD scores are obtained for the liver, kidney and spleen, accuracy is lower for the pancreas - highlighting the difficulty of segmenting this organ.

Fold 2 was our highest performing model in terms of DSC and NSD across the four segmented organs, so we selected this model to submit for evaluation with the validation and test sets.

5.2. Quantitative results on validation set.

Table 5 illustrates the results on validation cases. Comparison between Table 4 and Table 5 illustrates better DSC and NSD performance is obtained for the 5-fold cross validation than the validation set.

Table 4. Quantitative results of 5-fold cross validation in terms of DSC and NSD. We show the median and standard deviation of both measures for every fold individually and for all training folds combined (361 images).

Training	Liver		Kidney		Spleen		Pancreas	
	DSC (%)	NSD (%)	DSC (%)	NSD (%)	DSC (%)	NSD (%)	DSC (%)	NSD (%)
Fold 1	97.3± 0.9	85.7± 4.5	94.8± 2.2	87.1± 5.5	96.4± 2.0	91.7± 5.3	81.5± 9.0	55.5± 11.9
Fold 2	97.3± 2.1	85.8± 7.0	94.9± 4.9	88.3± 8.2	96.5± 3.1	92.0± 6.5	81.2± 7.3	59.9± 10.2
Fold 3	97.1± 1.5	84.8± 5.9	94.6± 4.3	86.3± 7.0	96.2± 2.7	91.9± 5.3	80.1± 13.2	54.5± 12.6
Fold 4	97.1± 0.8	85.6± 5.1	94.8± 3.6	87.0± 5.8	96.3± 1.3	92.1± 4.5	81.0± 9.3	55.9± 13.7
Fold 5	97.4± 1.0	86.7± 4.9	94.9± 1.5	88.4± 5.4	96.3± 4.7	92.3± 8.7	80.0± 10.4	57.6± 12.3
Median	97.3± 1.3	85.9± 5.6	94.8± 3.6	87.5± 6.5	96.4± 3.0	92.0± 6.3	80.9± 10.1	56.8± 12.3

Table 5. Quantitative results on validation set.

Organ	DSC (%)	NSD (%)
Liver	90.6±13.5	62.7±18.2
Kidney	68.1±29.5	54.8±25.9
Spleen	83.8±21.6	66.1±21.3
Pancreas	53.9±26.6	38.2±21.7

5.3. Qualitative results

In Figure 3 we present two examples of segmentations from the training subset predicted by our model. In the top row we show an axial slice of a patient where our model performs well. From left to right we show the CT scan alone, the gold-standard segmentations and the organ segmentation predictions made by our model. In this example, our model segments the liver (*blue*), kidneys (*green*), pancreas (*pink*) and spleen (*yellow*) with high accuracy compared to the gold standard. Our model struggles in resolving the internal structure of the right kidney, where the gold standard segmentation has avoided the calyx. In the bottom row, we show an example where our model struggles to produce an accurate segmentation for the kidneys (*green*). In this example, a large tumour (circled in red) has infiltrated the right kidney. In the gold standard this tumour is included in the kidney segmentation, however, our model fails to classify this structure as part of the kidney.

In figure 4 we show three examples from the validation set where our model does a great job at segmenting the organs. These examples show the degree of anatomical variation commonly observed for these abdominal organs. The segmentation predictions made by our model (right column) closely reflect the gold standard (central column).

In figure 5 we show three challenging examples from the validation subset. In each of these cases, disease is present which abnormally alters the shape, size or appearance of one or more of the organs. In row a), the left kidney (green) is abnormally large, possibly infiltrated by a tumour, and is has been failed to be segmented by our model. In row b), the pancreas (pink) is atypically enlarged and shaped. Our CNN is unable to segment the pancreas in this case. Finally, in row c) the liver (blue) is darker than usual. This is often observed in fatty liver cases. Our model is unable to segment this liver, even though the liver is still easily visible. These three cases are uncommon and it is likely that our model was not exposed to many similar examples in the training phase.

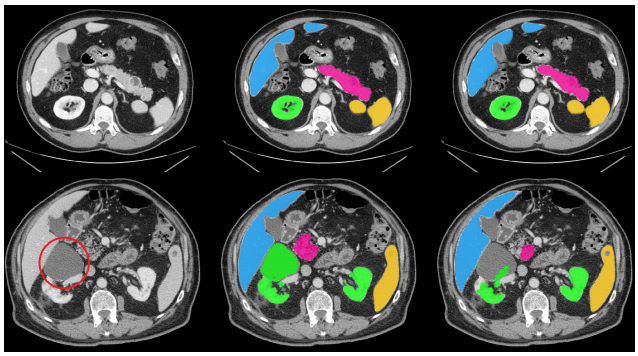

Figure 3. Two sets of axial slices of segmentation outputs from our model. Both patients shown here are from the training subset of images. In the first column is the CT image alone, the second column shows the gold standard segmentations and the third shows our models prediction. In the top row we show an example where our model performs well, accurately segmenting the liver (blue), kidneys (green), pancreas (pink) and spleen (yellow) with good accuracy. In the bottom row we show an example where our model struggles to segment the right kidney. In this example a tumour is present in the right kidney which our model is unable to recognise.

6. Discussion and Conclusion

We have developed a method for 3D segmentation of organs in abdominal CTs which is capable of producing high quality liver, kidney, spleen and pancreas segmentations in 1.6 seconds using only the CPU.

Medical image segmentation with CNNs is now very common and a well-studied field. However, many methods which produce accurate segmentations very quickly on a GPU will suffer significant slowdown when operating on a CPU. For example, Panda et al. [16] developed a 3D auto-segmentation method which produces high-quality segmentations of the pancreas (DSC = 0.91), but takes 4 minutes to perform inference for a single image on CPU.

We have performed a 5-fold cross-validation using the

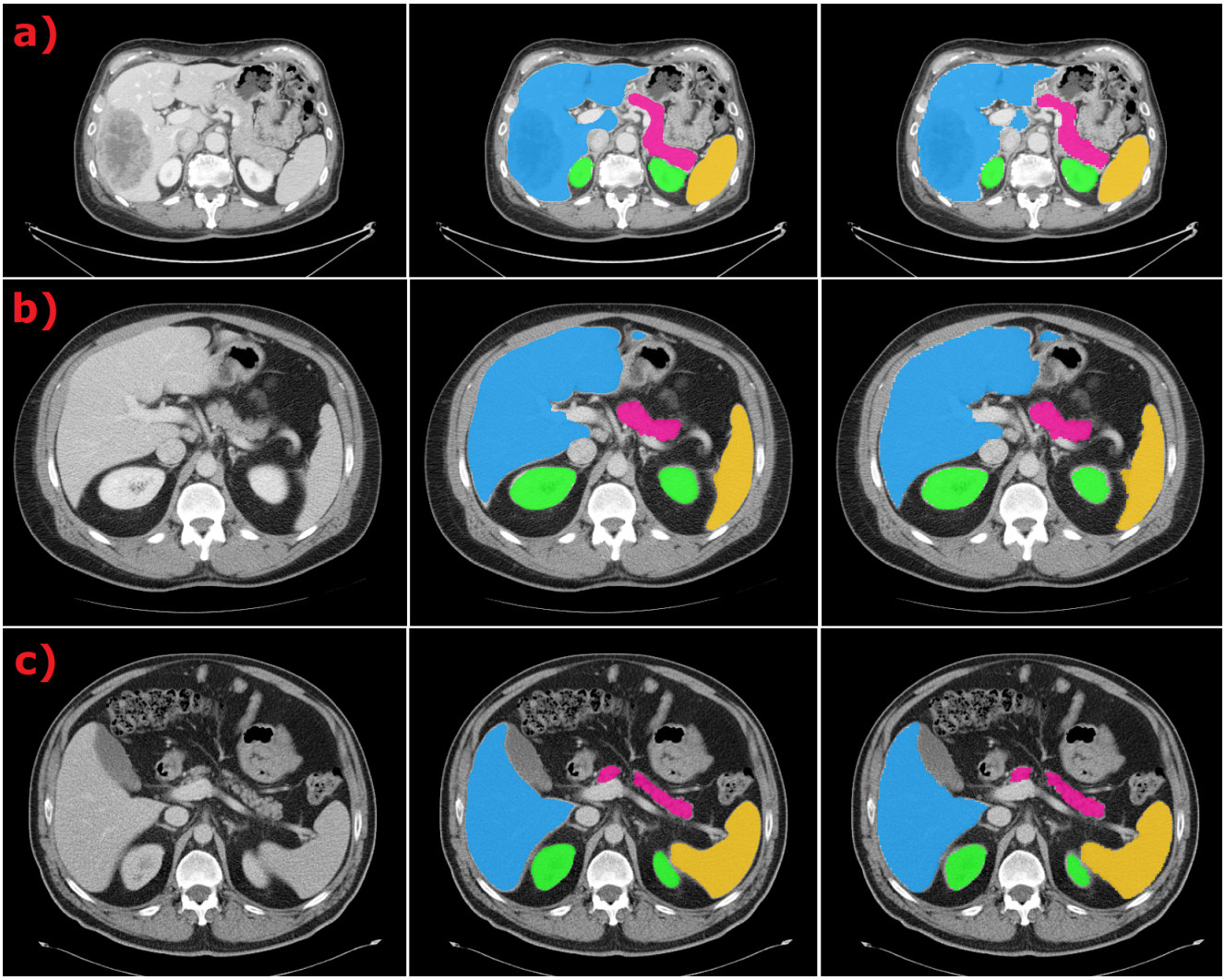

Figure 4. Three examples from the validation set where our model performs very well. As in figure 3, the first column shows the CT, the second the golden standard and the third shows our model's prediction. Our model can segment the abdominal organs in a diverse range of shapes and sizes reflecting common anatomical variation.

training subset of images (361 images). Our model performance was consistent across all five folds. However, when evaluated on the validation set, our models segmentation performance was substantially lower in all organs. This phenomenon may caused by over-fitting the model on the training set. However, we believe the significant deterioration in segmentation performance is due to the distribution of the contrast phases of images between the data splits. Our model was trained only with portal venous phase images, whereas at least half of the validation set is made up of images from an earlier contrast stage (late arterial).

Since we are unable to use external data for this challenge, in future we plan to include additional data augmentation to simulate different contrast phases in the training stage. In addition, we would like to ensure that any future training datasets contains examples of fatty livers and other diseases to ensure the model is robust to cases similar to the ones shown in figure 5.

One of the CTs in the training subset (*train_270_0000.nii.gz*) was missing many slices of the image, including large portions containing the liver and spleen. As a result, we excluded this image from the calculation of DSC and NSD.

Automated segmentation of medical images with CNNs is now the state-of-the-art [17], and will increasingly release clinicians from the time-intensive task of manually annotated structures. Our model performs fast and accurate 3D segmentation on the CPU, which enables much wider deployment of such models within clinics and research groups since it does not require the use of specialist hardware (GPUs).

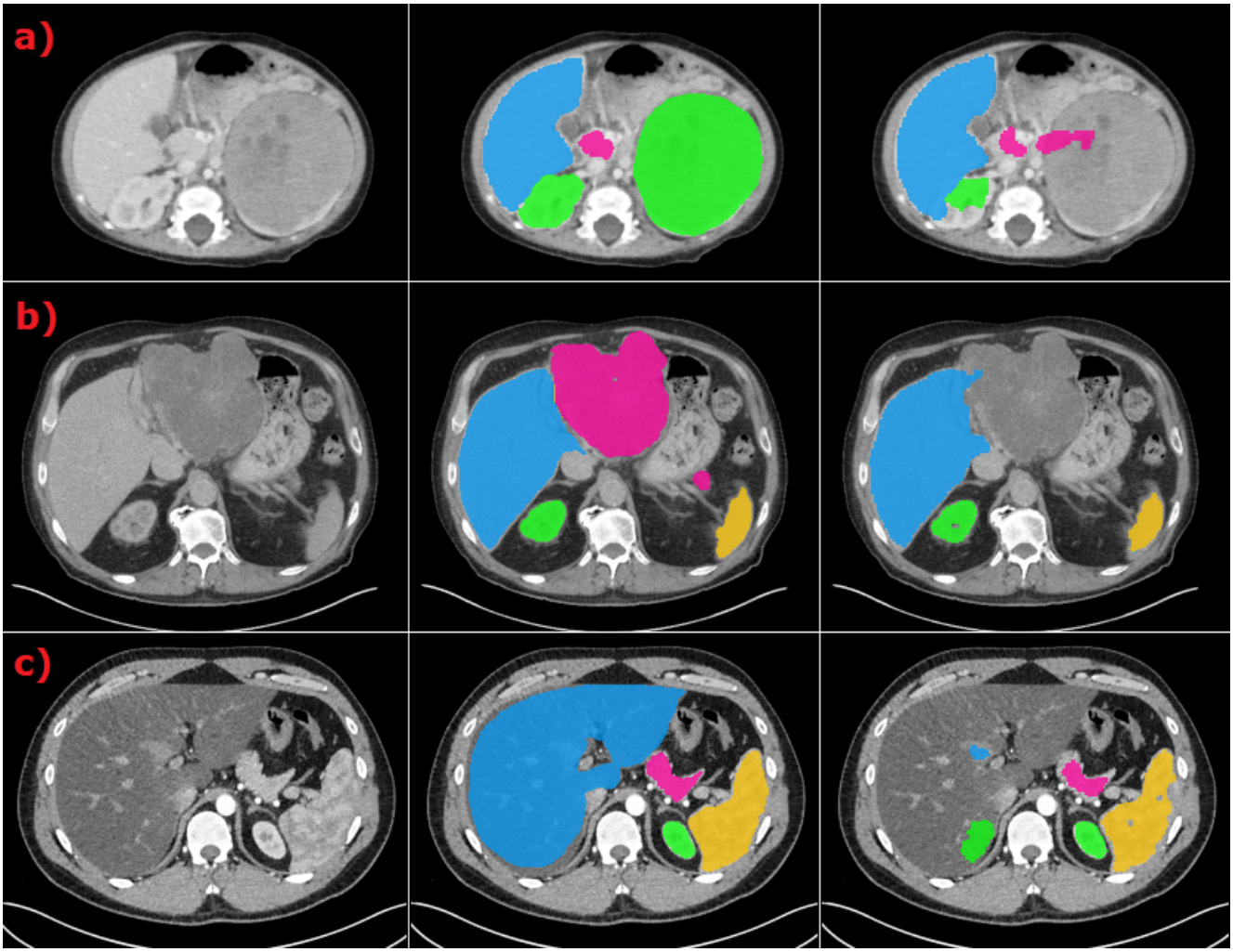

Figure 5. Three challenging examples drawn from the validation set where there is significant anatomical deformation caused by disease. As in figure 3, the first column shows the CT, the second the golden standard and the third shows our model's prediction. Our CNN is unable to segment the left kidney in case a) which is much larger than normal, likely infiltrated by a tumour. Similarly, our model is unable to segment the pancreas of case b) which is larger than standard and oddly shaped. The liver of case c) is darker than normal, often indicative of fatty liver disease. Due to it's darker appearance, our model is unable to segment this liver.

Acknowledgement

The authors of this paper declare that the segmentation method they implemented for participation in the FLARE challenge has not used any pre-trained models nor additional datasets other than those provided by the organizers.

References

[1] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2016, pp. 424–432. 1, 2

[2] J. Bai, F. Lu, K. Zhang *et al.*, "Onnx: Open neural network exchange," https://github.com/onnx/onnx, 2019. 1, 4

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. 2

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788. 2

[5] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem. IEEE Computer Society, dec 2016, pp. 2818–2826. [Online]. Available: https://arxiv.org/abs/1512.00567v3 3

[6] H. Yang, C. Yuan, B. Li, Y. Du, J. Xing, W. Hu, and S. J. Maybank, "Asymmetric 3D Convolutional Neural Networks for action recognition," *Pattern Recognition*, vol. 85, pp. 1–12, 2019. [Online]. Available: https://doi.org/10.1016/j.patcog.2018.07.028 3

[7] A. L. Simpson, M. Antonelli, S. Bakas, M. Bilello, K. Farahani, B. Van Ginneken, A. Kopp-Schneider, B. A. Landman, G. Litjens, B. Menze *et al.*, "A large annotated medical image dataset for the development and evaluation of segmentation algorithms," *arXiv preprint arXiv:1902.09063*, 2019. 3

[8] P. Bilic, P. F. Christ, E. Vorontsov, G. Chlebus, H. Chen, Q. Dou, C.-W. Fu, X. Han, P.-A. Heng, J. Hesser *et al.*, "The liver tumor segmentation benchmark (lits)," *arXiv preprint arXiv:1901.04056*, 2019. 3

[9] H. Roth, A. Farag, E. Turkbey, L. Lu, J. Liu, and R. Summers, "Data from pancreas-ct. the cancer imaging archive (2016)." 3

[10] H. R. Roth, L. Lu, A. Farag, H.-C. Shin, J. Liu, E. B. Turkbey, and R. M. Summers, "Deeporgan: Multi-level deep convolutional networks for automated pancreas segmentation," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2015, pp. 556–564. 3

[11] K. Clark, B. Vendt, K. Smith, J. Freymann, J. Kirby, P. Koppel, S. Moore, S. Phillips, D. Maffitt, M. Pringle *et al.*, "The cancer imaging archive (tcia): maintaining and operating a public information repository," *Journal of digital imaging*, vol. 26, no. 6, pp. 1045–1057, 2013. 3

[12] N. Heller, F. Isensee, K. H. Maier-Hein, X. Hou, C. Xie, F. Li, Y. Nan, G. Mu, Z. Lin, M. Han *et al.*, "The state of the art in kidney and kidney tumor segmentation in contrast-enhanced ct imaging: Results of the kits19 challenge," *Medical Image Analysis*, vol. 67, p. 101821, 2021. 3

[13] N. Heller, S. McSweeney, M. T. Peterson, S. Peterson, J. Rickman, B. Stai, R. Tejpaul, M. Oestreich, P. Blake, J. Rosenberg *et al.*, "An international challenge to use artificial intelligence to define the state-of-the-art in kidney and kidney tumor segmentation in ct imaging." *American Society of Clinical Oncology*, vol. 38, no. 6, pp. 626–626, 2020. 3

[14] J. Ma, Y. Zhang, S. Gu, C. Zhu, C. Ge, Y. Zhang, X. An, C. Wang, Q. Wang, X. Liu, S. Cao, Q. Zhang, S. Liu, Y. Wang, Y. Li, J. He, and X. Yang, "Abdomenct-1k: Is abdominal organ segmentation a solved problem?" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 3

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2015 Inter, 2015, pp. 1026–1034. 4

[16] A. Panda, P. Korfiatis, G. Suman, S. Garg, E. Polley, D. Singh, S. Chari, and A. Goenka, "Two-stage deep learning model for fully automated pancreas segmentation on computed tomography: Comparison with intra-reader and inter-reader reliability at full and reduced radiation dose on an external dataset," *Medical Physics*, vol. 48, no. 5, pp. 2468–2481, May 2021. 5

[17] C. E. Cardenas, J. Yang, B. M. Anderson, L. E. Court, and K. B. Brock, "Advances in Auto-Segmentation," *Seminars in Radiation Oncology*, vol. 29, no. 3, pp. 185–197, 2019. 6