Journal of Imaging Informatics in Medicine (2024) 37:3208–3216
https://doi.org/10.1007/s10278-024-01138-2

MedYOLO: A Medical Image Object Detection Framework

Joseph Sobek[1] · Jose R. Medina Inojosa[2,3] · Betsy J. Medina Inojosa[2] · S. M. Rassoulinejad-Mousavi[1] · Gian Marco Conte[1] · Francisco Lopez-Jimenez[2] · Bradley J. Erickson[1]

Received: 18 December 2023 / Revised: 10 April 2024 / Accepted: 29 April 2024 / Published online: 6 June 2024

Abstract
Artificial intelligence-enhanced identification of organs, lesions, and other structures in medical imaging is typically done using convolutional neural networks (CNNs) designed to make voxel-accurate segmentations of the region of interest. However, the labels required to train these CNNs are time-consuming to generate and require attention from subject matter experts to ensure quality. For tasks where voxel-level precision is not required, object detection models offer a viable alternative that can reduce annotation effort. Despite this potential application, there are few options for general-purpose object detection frameworks available for 3-D medical imaging. We report on MedYOLO, a 3-D object detection framework using the one-shot detection method of the YOLO family of models and designed for use with medical imaging. We tested this model on four different datasets: BRaTS, LIDC, an abdominal organ Computed tomography (CT) dataset, and an ECG-gated heart CT dataset. We found our models achieve high performance on a diverse range of structures even without hyperparameter tuning, reaching mean average precision (mAP) at intersection over union (IoU) 0.5 of 0.861 on BRaTS, 0.715 on the abdominal CT dataset, and 0.995 on the heart CT dataset. However, the models struggle with some structures, failing to converge on LIDC resulting in a mAP@0.5 of 0.0.

Keywords Object detection · Medical imaging · Computed tomography · Magnetic resonance · Convolutional neural network · Deep learning

Background

The standard approach for object localization in 3-D medical imaging uses segmentation models to create voxel-by-voxel annotations for the objects of interest. While this approach enables models to have great accuracy, it has several downsides. Generating voxel-accurate annotations for medical imaging is a time-consuming process that often requires multiple experts to verify label quality. Due to variability between annotators, medically accurate segmentations of organs or lesions can have issues with indefinite structure boundaries, potentially excluding relevant information in nearby tissue. Even with high-quality labels, segmentation models can struggle to accurately label target structure boundaries and often require post-processing to fill missing internal volumes and remove spurious predicted objects. Altogether this makes segmentation models cost-prohibitive to train while potentially limiting the predictive power of downstream diagnostic or classification models.

Object detection models offer an alternative that avoids these issues for tasks where voxel-specific accuracy is not required. Past work in 3-D object detection for medical imaging has primarily focused on models specialized for lung nodule detection, with few open-source or general-purpose object detection frameworks available. nnDetection [1] is open-source and provides the current state of the art for general-purpose object detection in 3-D medical imaging. This framework is self-configuring, featuring automatic preprocessing, augmentation, and fivefold cross-validation. However, the high degree of automation in the nnDetection framework limits the ability of users to modify the framework or its data input pipeline. A variety of 2-D object detection models, such as the YOLO [2] family of

✉ Joseph Sobek
sobek.joseph@mayo.edu

[1] Department of Radiology, Mayo Clinic, Rochester, MN, USA

[2] Department of Cardiovascular Medicine, Mayo Clinic, Rochester, MN, USA

[3] Division of Epidemiology, Department of Quantitative Health Sciences, Mayo Clinic, Rochester, MN, USA

models, are also available and can provide high-quality bounding boxes with slice-by-slice accuracy. However, these 2-D models are generally intended for use with photographs and require cumbersome conversion processes for both their input, to resolve the incompatibility of 3-D input data with the model, and their output, to reconstruct the predictions for use in downstream, 3-D tasks. This conversion process also discards 3-D spatial information, which may be helpful when detecting complex structures.

We have developed MedYOLO, an open-source, general-purpose 3-D object detection framework for medical imaging based on the Ultralytics YOLOv5 [3] detection model with native compatibility with NIfTI imaging. This model uses a one-shot method for object detection which inputs whole images into the model, in contrast to the sliding window approach used by nnDetection [1], which inputs overlapping patches from the original image and combines patch-based predictions to generate the final predictions. Though less automatic than nnDetection, the MedYOLO framework is more straightforward to modify and shows high performance on many organs, including some which nnDetection appears to struggle with.

Methods

Data

We tested MedYOLO on four types of medical imaging. These included the following: (1) FLAIR magnetic resonance (MR) scans from the BRaTS 2021 Task 1 dataset (1000 training scans, 251 validation) [4–8] using the whole tumor segmentation mask to generate bounding boxes. (2) LIDC lung nodule Computed tomography (CT) dataset (689 training scans, 173 validation) [9–11] using two different sets of labels, one placing bounding boxes around individual nodules and the other using a single bounding box containing every nodule in each scan. These were used to test the model's detection of very small objects and diffuse structures respectively. (3) A proprietary abdominal organ segmentation CT dataset [12] (60 training scans, 15 validation). From this, bounding boxes were generated for the following organs: left kidney, right kidney, spleen, pancreas, diaphragm, bladder, uterus, prostate, aorta, spinal cord, stomach, and liver. Due to the limited number of training examples for this dataset, an additional preprocessing step was used to generate rotated training examples. Training scans were rotated in the axial view using five different base rotation angles for each example (0°, ± 8°, and ± 17°) with an additional random jitter of ± 3°. An identical rotation was applied to each image's segmentation mask, and then bounding box labels for the rotated example were generated. The original, unrotated examples were also included in the training set to mitigate any spurious signals from the interpolation algorithm used for the rotation. In total, this resulted in 360 training examples for the abdominal CT dataset. (4) A clinically indicated and randomly selected ECG-gated cardiac CT dataset (648 training scans, 163 validation) curated with data from Mayo Clinic and annotated by the authors using the RILContour application [13], for which we attempted to predict a bounding box around the heart and thoracic aorta. Each of these datasets was split at the patient level to prevent data leakage between the training and validation sets.

Metrics

To analyze MedYOLO's performance, we use 3-D analogues of the object detection metrics reported by YOLOv5 [3]. First is mean average precision (mAP). This defines an intersection over union (IoU) threshold, in this case, IoU 0.5, which predictions must meet or exceed to be considered correct. The model's precision-recall curves at the chosen IoU threshold are then used to calculate the model's average precision for each class, and, finally, the mean of these average precision values is taken over every class. Second is a 3-D version of the COCO metric [14], which averages the mAP from IoUs ranging between 0.5 and 0.95 with an interval of 0.05. The nnDetection framework, which we compare MedYOLO against, only reports mAP in IoU intervals of 0.1 up to an IoU of 0.9, so for it we report a slightly more relaxed version of the COCO metric averaging mAP for IoUs between 0.5 and 0.9 with an interval of 0.1.

Network Design

MedYOLO is modeled after the YOLOv5 framework but rewritten for use with 3-D medical imaging. It accepts NIfTI files as input and includes normalization functions for CT and MR scans. This framework was chosen because it was state of the art for 2-D object detection when MedYOLO development started, earlier versions of YOLO showed good performance on radiographs [15], and preliminary tests with YOLOv5 showed promising results on the cardiac CT dataset. While there have since been several innovations to the YOLO family of models, with iterative improvements currently reaching YOLOv9 [16], significant differences in the codebases and detection strategies obstruct the addition of their innovations to our framework. Like YOLOv5, MedYOLO is an anchor-based detection model created from a convolutional neural network (CNN) with a terminal detection layer that connects to intermediary layers earlier in the network. The primary difference within the CNN, compared to YOLOv5, is the replacement of 2-D neural network layers with their 3-D versions, as shown in the architectural diagram in Fig. 1.

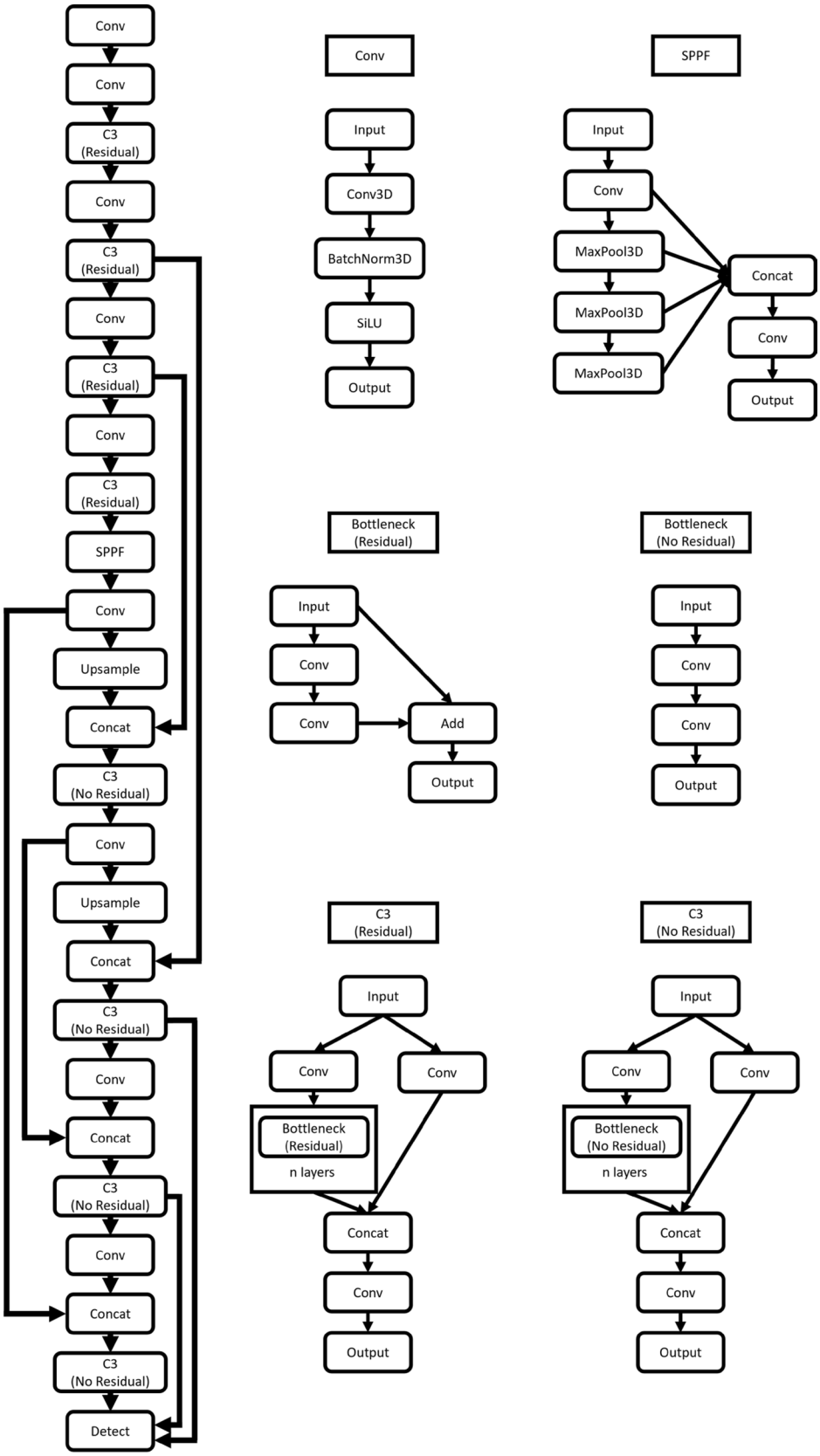

Fig. 1 Architectural diagram of MedYOLO and its component modules. Due to its complexity, the Detect module is easier understood through code and not displayed here

The width and number of MedYOLO's layers are configurable using yaml files, with small, medium, and large versions included in the repository. Several of YOLOv5's key library dependencies, such as the functions it calls from OpenCV, are incompatible with 3-D data, and removing these dependencies required significant changes to the data handling pipeline and the removal of some augmentation routines. A limited portion of the YOLOv5 codebase is compatible with the additional dimension, and this code was retained to improve the interpretability of the MedYOLO framework.

As in earlier versions of YOLO [2], MedYOLO calculates priors for its anchor boxes using k-means clustering on the training set labels. Our initial tests using the elbow method determined six anchor boxes, in contrast to the three used by YOLOv5, to be the appropriate number. This was left fixed for every dataset tested but is a configurable hyperparameter.

The MedYOLO CNN requires cubic inputs analogous to the square inputs used by 2-D YOLO models. Since medical imaging is typically anisotropic in at least one dimension, we use trilinear interpolation to reshape input data into cubes of a user-configurable size. To balance batch size against available GPU resources and avoid errors caused by undersized inputs, a side length of 350 pixels per side was chosen for most of our tests. This gives us a final feature map size of $11 \times 11 \times 11$ at the bottom of the CNN. Cubes with side length 512 were also tested for the LIDC dataset, which gave a final feature map size of $16 \times 16 \times 16$ (see Table 1 for the corresponding GPU footprints during training).

An example NIfTI scan, which for the tested datasets typically has a shape between $512 \times 512 \times 40$ and $512 \times 512 \times 100$, loaded into the pipeline is first converted into a PyTorch Tensor. This Tensor is then transposed from (X, Y, Z) to (Z, X, Y) such that it has a shape of $40 \times 512 \times 512$. The Tensor is then interpolated into a cube with a shape of $350 \times 350 \times 350$. Augmentation, as detailed in the next section, is then applied. Finally, a modality-appropriate normalization is applied to the Tensor. The framework allows users to apply their own normalization and preprocessing routines if desired.

Table 1 VRAM consumption during training for different model sizes at different input data scales

Model	Batch size	Cube length (voxels)	VRAM (GB)
MedYOLO-S	8	350	35
MedYOLO-S	2	512	26
MedYOLO-L	2	350	25

Training

The MedYOLO training process largely matches that of YOLOv5. We use nearly identical hyperparameters to those used by YOLOv5 for training from scratch, the only exception being hyperparameters to configure the added augmentation routines. Several of YOLOv5's augmentation routines needed to be removed, such as color value shifts and random perspective shifts, as they are inapplicable to grayscale data or rely on 2-D imaging libraries. Three forms of live augmentation were used on every dataset: random cutout augmentation, which masks blocks of random size within the image with random noise, random translation augmentation, and random zoom augmentation. We trained MedYOLO on each dataset for 1000 epochs using the Adam optimizer [17], with early stopping occurring after 200 epochs passed with no improvement.

MedYOLO uses a version of YOLOv5's composite loss function adapted for 3-D volumes. The bounding box loss component compares the IoU and the distance between the centers of the predicted and target bounding boxes. The objectness component, which trains the model to evaluate its predictions by comparing their IoUs to the model's confidence level, is calculated using binary cross-entropy. Finally, the classification loss uses binary cross-entropy on the predicted class. The repository includes focal loss options for the latter two components, but these were not used for the results reported in this paper.

As it is the current state of the art, has readily available code, and performs well without an extensive hyperparameter search, the nnDetection framework [1] was chosen for performance comparisons. It was trained on the same datasets with identical training and validation splits for comparison purposes. The only exception is the abdominal CT dataset. For this dataset, the patients were split into training and validation sets identically as they were for MedYOLO, but, because nnDetection does not track patient IDs during the cross-validation process and runs automatic augmentation routines which include rotation, we did not use the pregenerated rotated training examples with nnDetection. This avoids data leakage between cross-validation folds and interference with the automatic augmentation.

Results

In Table 2, we report the best validation metrics after training both MedYOLO and nnDetection. Where not otherwise specified, results from the small version of MedYOLO (MedYOLO-S) are reported. For the BRaTS dataset early stopping occurred at epoch 675, for the ECG gated cardiac dataset early stopping occurred at epoch 938, and for the abdominal dataset and LIDC the small model trained for

Table 2 mAP comparisons between MedYOLO and nnDetection. Due to MedYOLO's failure to converge on LIDC, we did not train nnDetection on the dataset

	MedYOLO mAP@0.5	MedYOLO mAP@0.5:0.95	nnDetection mAP@0.5	nnDetection mAP@0.5:0.9
BRaTS	0.861	0.431	0.836	0.488
BRaTS (MedYOLO-L)	0.0	0.0	0.836	0.488
Gated cardiac	0.995	0.852	0.0	0.0
Abdominal	0.683	0.203	.208	.064
Abdominal (MedYOLO-L)	0.715	0.246	.208	.064
LIDC	0.0	0.0	*	*

1000 epochs. We also trained the large version of MedYOLO (MedYOLO-L) on the abdominal dataset, for which early stopping occurred at epoch 968, and on BRaTS, on which it trained for 1000 epochs.

Our results show that MedYOLO matches or exceeds nnDetection's performance for most datasets. Both MedYOLO and nnDetection perform well on BRaTS, and example MedYOLO bounding boxes are shown in Fig. 2. In contrast, the Gated cardiac CT dataset shows significantly different behavior between the frameworks. While MedYOLO nearly completely captures the target volume, nnDetection shows poor mAP at higher IoUs. Further investigation, depicted in Fig. 3, shows predictions from the nnDetection framework are correctly localized but do not correctly span the target volume. This creates highly confident predictions with IoUs between 0.1 and 0.4, but, for this dataset, none with IoU above 0.5.

Table 3 provides a more detailed breakdown of MedYOLO's performance on the Abdominal dataset compared to nnDetection, with example validation predictions from MedYOLO-L shown in Fig. 4. While both MedYOLO models generally outperformed nnDetection in the abdomen, the latter performed significantly better at identifying the uterus, a class present in only 17 out of the 60 original abdominal training examples and 4 out of the 15 validation examples. MedYOLO-L provides general improvements to the performance of the small model on the Abdominal dataset at the cost of slower training. However, its performance on

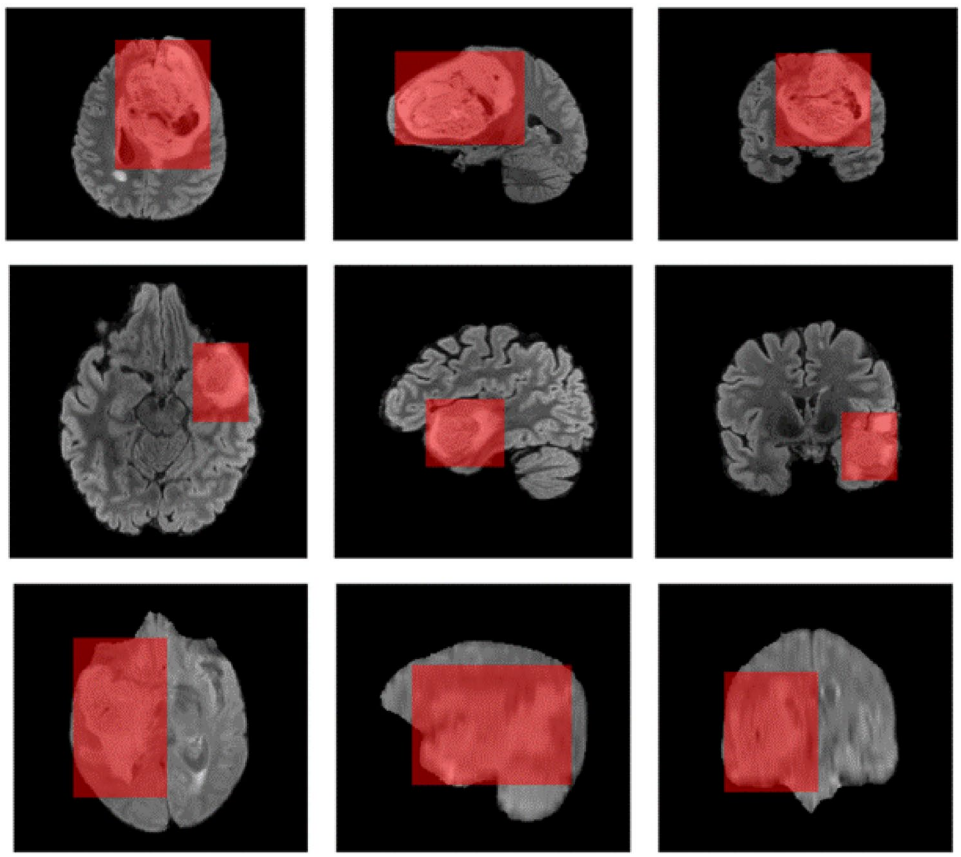

Fig. 2 Bounding box segmentations for BRaTS validation data using MedYOLO-S on three different scans

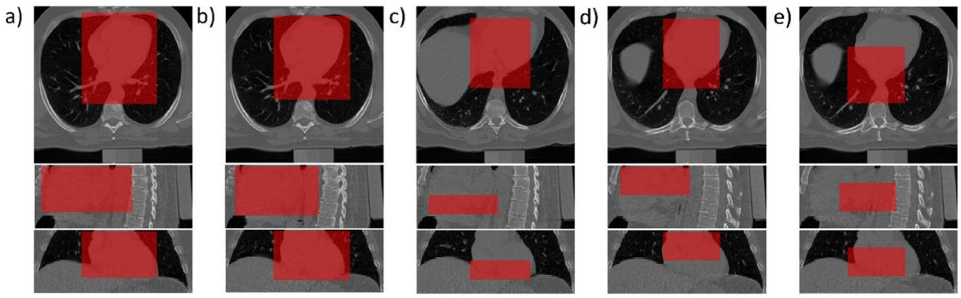

Fig. 3 Bounding box segmentations for the ECG-gated cardiovascular CT dataset. Segmentations correspond to the following: ground truth (**a**), MedYOLO prediction (**b**), nnDetection highest confidence prediction (**c**), nnDetection 2nd highest confidence prediction (**d**), nnDetection 3rd highest confidence prediction (**e**)

Table 3 Per-class average precision (AP) on the abdominal dataset for each model at IoU 0.5 and averaging IoUs between 0.5 and 0.9 or 0.95 for nnDetection and MedYOLO respectively

MO test set performance per class	nnDetection		MedYOLO-S		MedYOLO-L	
	AP@0.5	AP@0.5:0.9	AP@0.5	AP@0.5:0.95	AP@0.5	AP@0.5:0.95
Left kidney	.308	.118	.49	.115	.866	.254
Right kidney	.251	.061	.781	.199	.869	.394
Spleen	.124	.034	.836	.198	.698	.158
Pancreas	.09	.025	.868	.202	.85	.225
Diaphragm	0	0	.752	.338	.82	.431
Bladder	.725	.223	.647	.247	.702	.245
Uterus	.629	.202	.138	.019	.201	.073
Prostate	.277	.081	.75	.228	.897	.219
Aorta	.01	.004	.634	.12	.593	.166
Spinal cord	0	0	.567	.14	.423	.159
Stomach	.085	.026	.791	.2	.724	.197
Liver	0	0	.937	.435	.942	.431

the uterus remains substantially worse than nnDetection's performance.

Despite good results on a variety of different structures, MedYOLO failed on two of the tested tasks. The framework did not demonstrate any learning progress for the LIDC dataset in any tested training configuration and was unable to detect any lung nodules. This is likely due to the coarse final feature maps of MedYOLO's CNN limiting the model's capacity to detect very small structures, though the available datasets did not allow us to explore the lower bounds of detectable structures. While MedYOLO-S displayed similar performance to nnDetection on the BRaTS dataset, MedYOLO-L failed to converge on the dataset and was unable to identify tumors.

Discussion

MedYOLO shows promising results for use as an object detector for 3-D medical imaging, capable of achieving good results using a simple, one-shot detection method. On many structures, MedYOLO's method shows similar or better performance than the sliding window method used by nnDetection. However, during our analyses, the model was unable to detect lung nodules, which suggests the sliding window method offers advantages for some tasks. Achieving high mAP on a variety of structures without hyperparameter tuning or dataset-specific pre-processing suggests MedYOLO has some robustness against suboptimal hyperparameter choices. The framework's ability to quickly and accurately identify and localize structures makes it ideal for use in machine learning pipelines to localize data to relevant volumes before it is passed to downstream models.

Despite the impressive performance for some tasks, the current implementation has significant potential for improvement. One of the biggest avenues for improvement would be additional augmentation routines. In particular, the performance on BRaTS tumor detection, with MedYOLO-S finishing training early despite comparably modest results and MedYOLO-L failing to converge, suggests the model may be missing pertinent information. Further augmentation could extend the training process to achieve higher performance on this and other datasets. The obstacle we encounter is a relative lack of efficient and easily implementable augmentation routines for 3-D data, causing many standard live augmentation strategies to slow down training speeds significantly.

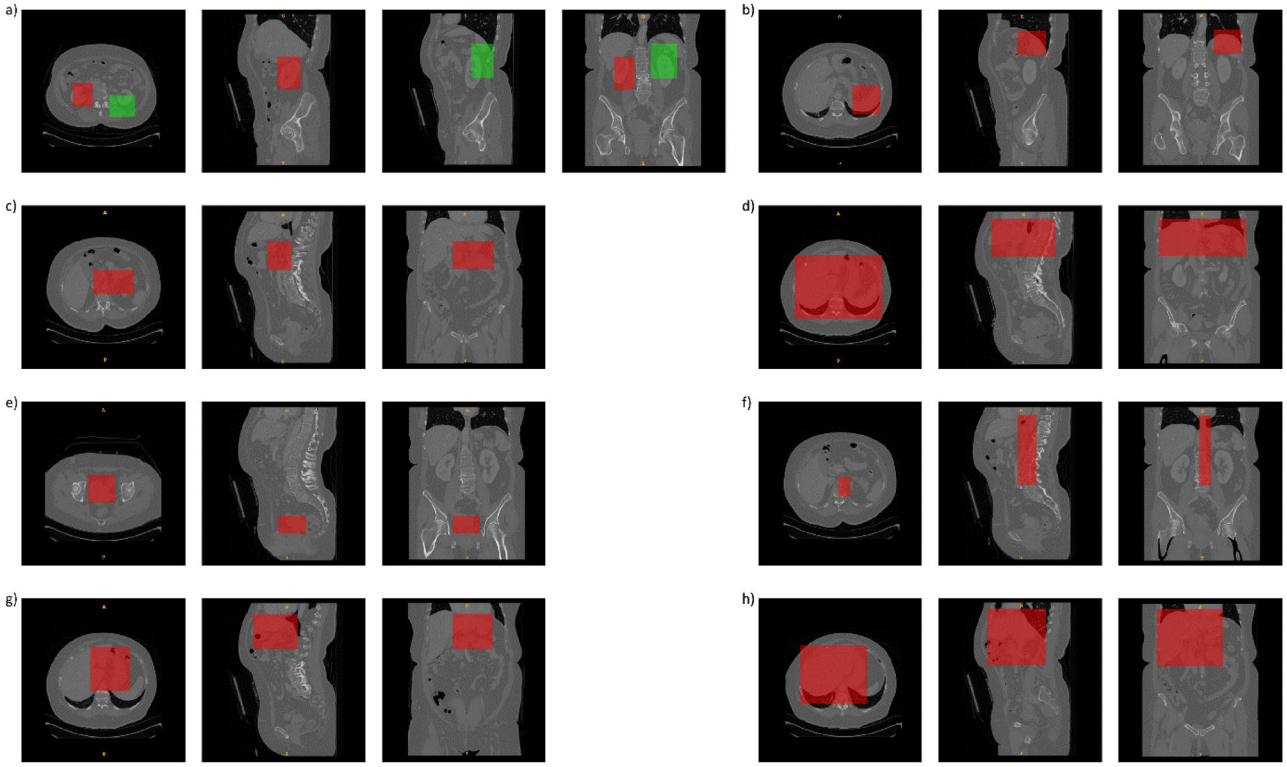

Fig. 4 Bounding box segmentations for several organs in the abdominal dataset using MedYOLO-L. **a** Left and right kidneys. **b** Spleen. **c** Pancreas. **d** Diaphragm. **e** Bladder. **f** Aorta. **g** Stomach. **h** Liver

Pre-generating augmented training data avoids this issue, but this increases storage requirements drastically due to the large memory footprint of medical imaging.

Another modification that may improve performance is switching the algorithm we use for reshaping data. Trilinear interpolation allows us to smoothly transform 3-D input data into a cubic shape but does not improve the information available in the input. A more sophisticated resampling method, such as super-resolution, could provide extra detail and add value to the slices created during the reshaping process. To present a more accurate and easily reproduced assessment of baseline results we limited ourselves to the simpler method.

The requirement to reshape input data into cubic volumes is probably the biggest weakness of our pipeline. Increasing the number of input slices, often by an order of magnitude, dramatically increases the computational resources demanded by our model. In addition, balancing batch memory constraints against the accuracy of batch statistics requires us to use smaller cubes and reduce the axial resolution of our input data. Gradient accumulation over several batches could help resolve this. We did not experiment with this because we achieved adequate performance without it on most of the datasets tested. While it may have been useful for the LIDC dataset, the lack of any training progress even when giving the model the full axial resolution of the images suggests this is unlikely. A related issue comes from medical imaging datasets in general, which often consist of images with a variable number of slices. Reshaping these datasets into a fixed, cubic size risks distorting input scans in unpredictable ways relative to one another. Signs of this appear during inference, as the model primarily fails to predict accurate bounding boxes for scans with unusually few slices, even for the ECG-gated cardiac dataset on which its validation metrics are extremely high.

Early during development, substantial effort was applied to remove the cubic shape requirement from the MedYOLO framework, in particular for the added depth dimension, but this is necessary for the following reasons. First, the fixed depth and number of pooling layers in the MedYOLO and YOLO architectures impose minimum resolution requirements on every dimension. The resolution chosen for most of the discussed datasets, $350 \times 350 \times 350$, is close to this minimum resolution. Second, YOLOv5 models require square input data to simplify anchor-based prediction. This becomes a requirement for cubic inputs when transforming these 2-D models into 3-D. To handle rectangular images, YOLOv5 resamples and zero-pads input into square shapes which reduces the distortion that occurs during resampling. This process is computationally cheap for 2-D data, but the

larger memory footprint and highly anisotropic shapes of medical imaging make it impractical to zero-pad 3-D input data into cubes, limiting us to resampling techniques.

Despite these limitations, MedYOLO demonstrates that one-shot, anchor-based approaches can achieve high performance on 3-D medical object detection tasks. However, future frameworks may perform better using this YOLO-like approach in a 2.5-D paradigm. This would enable new frameworks to maintain the native resolution of their input data without compromising batch size or introducing distortion from reshaping. The primary downside of a 2.5-D approach compared to a 3-D approach is that accurate bounding boxes require additional annotation effort, though still less than voxel-accurate segmentation requires.

Conclusion

We report the development of MedYOLO, an anchor-based, one-shot object detection framework written for use with 3-D medical imaging. MedYOLO matches or exceeds nnDetection's performance localizing a broad range of structures in both CT and MR modalities without hyperparameter optimization. Areas for future improvement to the pipeline include expanding the augmentation routines as efficient 3-D algorithms are developed and incorporating more sophisticated interpolation methods for resampling input data into the required cubic shapes.

Acknowledgements The authors acknowledge the National Cancer Institute and the Foundation for the National Institutes of Health, and their critical role in the creation of the free publicly available LIDC/IDRI Database used in this study.

Author Contribution Joseph Sobek coded the framework, converted the annotations, trained and evaluated the models, and prepared the manuscript. Jose Medina Inojosa and Betsy Medina Inojosa assisted in the acquisition of the cardiac CT dataset and helped edit the manuscript. Seyed Moein Rassoulinejad-Mousavi provided additional curation for the abdominal CT dataset. Gian Marco Conte assisted with acquiring and converting the BRaTS dataset and provided comments on the manuscript. Bradley Erickson and Francisco Lopez-Jimenez were the principal investigators and provided comments on the manuscript.

Funding This work was supported by Mayo Clinic internal funding.

Data Availability The BRaTS and LIDC data are available from TCIA. To protect patient privacy, the abdominal and cardiac datasets are not available to share.

Code Availability The code for MedYOLO is available at: https://github.com/JDSobek/MedYOLO

Declarations

Ethics Approval This retrospective study was reviewed and deemed exempt by Mayo Clinic's Institutional Review Board.

Consent to Participate The informed consent was waived by ethics groups.

Consent for Publication The informed consent was waived by ethics groups.

Competing Interests The authors declare no competing interests.

References

1. Baumgartner M., Jäger P.F., Isensee F., Maier-Hein K.H.: nnDetection: A Self-configuring Method for Medical Object Detection. Medical Image Computing and Computer Assisted Intervention – MICCAI 2021. https://doi.org/10.1007/978-3-030-87240-3_51, 2021
2. Redmon J, Farhadi A: YOLO9000: Better, Faster, Stronger. 2017 IEEE Conference on Computer Vision and Pattern Recognition. https://doi.org/10.48550/arXiv.1612.08242, 2016
3. Jocher G, et al.: ultralytics/yolov5: v7.0 - YOLOv5 SOTA Real-time Instance Segmentation. https://doi.org/10.5281/zenodo.3908559, 2022
4. U. Baid, et al.: The RSNA-ASNR-MICCAI BraTS 2021 Benchmark on Brain Tumor Segmentation and Radiogenomic Classification. https://doi.org/10.48550/arXiv.2107.02314, 2021
5. B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, et al.: The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS). IEEE Transactions on Medical Imaging 34:1993-2024, 2015
6. S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J.S. Kirby, et al.: Advancing The Cancer Genome Atlas glioma MRI collections with expert segmentation labels and radiomic features. Sci Data 4:170117, 2017
7. S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, et al.: Segmentation Labels and Radiomic Features for the Pre-operative Scans of the TCGA-GBM collection. The Cancer Imaging Archive. https://doi.org/10.7937/K9/TCIA.2017.KLXWJJ1Q, 2017
8. S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, et al.: Segmentation Labels and Radiomic Features for the Pre-operative Scans of the TCGA-LGG collection. The Cancer Imaging Archive. https://doi.org/10.7937/K9/TCIA.2017.GJQ7R0EF, 2017
9. Armato III, S. G., et al.: Data From LIDC-IDRI. The Cancer Imaging Archive. https://doi.org/10.7937/K9/TCIA.2015.LO9QL9SX, 2015
10. Armato III, S.G., et al.: The Lung Image Database Consortium (LIDC) and Image Database Resource Initiative (IDRI): A completed reference database of lung nodules on CT scans. Medical Physics 38:915—931, 2011
11. Clark, K., et al.: The Cancer Imaging Archive (TCIA): Maintaining and Operating a Public Information Repository. Journal of Digital Imaging 26:1045-1057, 2013
12. Weston, A.D., et al.: Complete abdomen and pelvis segmentation using U-net variant architecture. Med. Phys., 47:5609-5618, 2020
13. Philbrick, K.A., et al.: RIL-Contour: A Medical Imaging Dataset Annotation Tool for and with Deep Learning. Journal of Digital Imaging 32:574-581, 2019
14. Lin, T.-Y., et al.: Microsoft COCO: Common Objects in Context. https://doi.org/10.48550/arXiv.1405.0312, 2015
15. Rouzrokh, P. et al.: Deep Learning Artificial Intelligence Model for Assessment of Hip Dislocation Risk Following Primary Total Hip Arthroplasty From Postoperative Radiographs. https://doi.org/10.1016/j.arth.2021.02.028, 2021

16. Wang, C.-Y., et al.: YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. https://doi.org/10.48550/arXiv.2402.13616, 2024
17. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. https://doi.org/10.48550/arXiv.1412.6980, 2014