Single Shot Lightweight Model For The Detection of Lesions And The Prediction of COVID-19 From Chest CT Scans

Aram Ter-Sarkisov

***Abstract*—We introduce a lightweight model derived from Mask R-CNN that segments lesions and predicts COVID-19 from chest CT scans in a single shot. The model requires a small dataset to train, and is evaluated on a large set of images to achieve a 42.45% average precision on the segmentation test split, and 93.00% COVID-19 sensitivity and F1-score of 96.76% on the classification test split across 3 classes: COVID-19, Common Pneumonia and Negative. We introduce an augmented Region of Interest layer that disentangles lesion detection functionality for segmentation and classification problems. Efficiency of the solution is confirmed by comparing it to a suite of the state-of-the-art models across both problems. Full source code, models and pretrained weights are available on https://github.com/AlexTS1980/COVID-Single-Shot-Model.**

***Index Terms*—COVID-19, Instance Segmentation, Object Detection, Regions of Interest, Image Classification**

I. Introduction

Examination of chest CT scans is one of the most popular and accurate ways of predicting COVID-19, alongside x-ray radiography (CXR) and real-time polymerise chain reaction (RT-PCR): it is faster than RT-PCR and more accurate than CXR. Depending on the stage of the virus, it can have higher sensitivity to COVID-19 than RT-PCR too, see [11], [1]. Since the onset of the COVID-19, a large number of convolutional neural networks (convnets) and other deep learning (DL) models for the detection of COVID-19 from chest CT scans and segmentation of lesions was introduced. Classifiers typically use a feature extractor like ResNet with a problem-specific logit output, e.g. COVID-19, Common Pneumonia (CP), like in [4], [7], [11], [2]. Most segmentation models use a model like U-Net with an encoder-decoder architecture to predict lesion masks at a pixel level, e.g. [3], [20], or fuse them with deep features for simultaneous lesions segmentation and image classification, e.g. in [18].

Recently introduced COVID-CT-Mask-Net, see [15], fuses advanced object detection and segmentation Mask R-CNN and Faster R-CNN models ([5], [12]) with image classification that exploits Mask R-CNN's functionality for predicting objects' bounding boxes, classes and masks from the regions of interest (RoIs). Batch of these predictions is converted into a ranked vector of features that the image classification layer in COVID-CT-Mask-Net can learn. In [16], [17] a number of modifications were presented, including truncated lightweight versions of the base model that achieve a higher accuracy of COVID-19 prediction and lesion segmentation. The main drawback of this approach is that it is split into two stages: first, Mask R-CNN is trained on the segmentation dataset, then, a classification model is initialized from its weights to train on the classification dataset.

In this paper we present a solution that fuses lesion segmentation in chest CT scans and prediction of the class of the input image (COVID-19, Common Pneumonia, Control) in a single shot (single shot model, SSM). The solution relies on Mask R-CNN's RoI functionality and consists of four main stages: backbone (ResNet feature extractor + Feature Pyramid Network, FPN), Region Proposal Network (RPN), Region of Interest (RoI) and an image classifier.

In our new solution we introduce the following novelties:

1) We disentangle instance (regional) detection functionality for segmentation and classification problems by augmenting RoI layer with a parallel classification branch, which has the same architecture as the segmentation branch. At training time, this disentangles the learning at instance (segmentation) and image (classification) levels,
2) We fuse lesion segmentation and COVID-19 prediction from chest CT scans in a single shot using the augmented RoI layer and achieve high precision across both problems outperforming a number of benchmark open-source solutions applied to our problem,
3) All backbone feature extractors are lightweight (truncated) and have less than 14M parameters; as a result, training and evaluation are very fast. On a CPU, processing of a single 512×512 chest CT scan slice takes between 3.81 - 7.08s, which includes the full segmentation output and the image class prediction.

Full solution is available in a Github repository: https://github.com/AlexTS1980/COVID-Single-Shot-Model. To the best of our knowledge, this is the first paper that presents a fusion of lesion instance segmentation and COVID-19 classification in a single shot. The rest of the paper is structured as following: in Section II we discuss datasets for both problems, in Section III we present the methodology, in Section IV we discuss experimental setup and results, in Section V we explain methodological limitations of our approach, Section

Aram Ter-Sarkisov is with the
Department of Computer Science
City, University of London
London, United Kingdom
e-mail: alex.ter-sarkisov@city.ac.uk

VI concludes.

II. Data

We require two separate sets of the training and evaluation data: segmentation data and classification data. Both of these sets are taken from CNCB-NCOV [19], http://ncov-ai.big.ac.cn/download resource. Segmentation data (750 images labelled at pixel level) is split randomly into 650 training and validation and 100 test images. Masks for the lesion classes, Ground Glass Opactity (GGO) and Consolidation (C) are merged into a single positive lesion class. Clean lung masks are merged with the background. Therefore, we have a total of 1+1 classes (background and lesions). All 750 images (scan slices) were taken from COVID-19-positive patients, but some of the slices are negative (no lesions present). They are discarded in the training stage, and labelled as a single negative (no lesions) observation at test stage.

For the 3-class (COVID-19, CP, Normal) classification data we use the COVIDx-CT [4] test and validation splits in full, and the training sample of 3000 images (1000/class) from the train split in [17], see Table I. The splits are consistent across classes and patients. This means that negative slices taken from the positive (COVID-19 or CP) patients were removed from the data altogether, and only those with lesions were kept, and every patient was randomly assigned only to one of the splits, see [4] and [16] for the details. Table I reports the segmentation/classification data properties. The ratio of the test to training split for the classification problem is very high, 7.06. This is one of the key advantages of SSM: it generalizes very well to the unseen data while using only a small portion of the training dataset.

III. Methodology

Mask R-CNN is one of the state-of-the-art models that detects and segments instances of objects in images using Region Proposal (RPN) and Region of Interest (RoI) layers. This is different to semantic segmentation that predicts classes at a pixel level, such as FCN [10] and UNet [14]. Unlike semantic segmentation models, Mask R-CNN detects separate objects, defined by their classes, bounding boxes and masks, and is therefore efficient at handling problems like partial occlusion or distinguishing between adjacent objects of the same class. At the same time, Mask R-CNN does not make global predictions, i.e. classification of the whole input image. Some previous solutions, e.g. [15], [16] fuse Mask R-CNN instance predictions with the global prediction for COVID-19 data, albeit in two stage (first segmentation, then classification).

The model we present in this study fuses segmentation and classification functionality for training and testing in a single shot. For comparison, we present two main approaches:

1) Pretraining Mask R-CNN on segmentation data, followed by the joint training on the segmentation and classification data,
2) Training the model from scratch in a single shot on segmentation and classification data through disentanglement of lesion detection for segmentation and classification problems.

Training time for all pretrained models in Table I includes the pretraining time. Each ResNet feature extractor is followed by a single FPN layer (see [8]), connected only to the last ResNet block (for simplicity we reduce this term to block).

A. Region of Interest layer

We briefly discuss RoI layer in Mask R-CNN, which is at the core of both fusion of segmentation and classification functionality and disentanglement of RoI detection in different types of images. Its segmentation branch is fully inherited from Mask R-CNN, see Figure 2 (light gray background).

First, RoI layer accepts β_1 raw box candidates from RPN and image-level features from FPN. At training time, RoIDetectBatch samples a batch of β_2 candidates to compare it to the gt labels. RoIAlign maps these candidates to FPN features to extract a batch of regional features using RoIAlign algorithm. This batch of RoIs with dimensionality $\beta_2 \times C \times H \times W$ (β_2 is the batch size, C is the number of channles/feature maps, H and W are the height and width of each feature map) is filtered through the fully connected BoxHead layer to predict refined boxes and classes for objects in the input image in BoxClass Prediction layer. For the segmentation step, RoIAlign(Mask) and MaskHead convolution layer have the same functionality as RoIAlign and BoxHead for masks, and MaskPrediction layer predicts mask for each object.

At test time, all raw candidates from RPN are filtered through RoIAlign, BoxClass, RoIAlign(Mask) and MaskHead to output a set of at most β_3 object predictions (class confidence scores + boxes + masks) with class confidence scores exceeding a preset threshold.

In [15] RoIBatchSelection method was added to the RoI layer, in order to extend its regional outputs to image classification. This functionality is explained in greater details in Sections III-B and III-C.

B. Fusion of Segmentation and Classification Functionality (base RoI layer)

In this setup, Mask R-CNN is pretrained on the segmentation data in Table I. Next, it is converted into SSM in Figure 1 by augmenting it with an image classifier **S**, Figure 3. Weights from Mask R-CNN are copied into the SSM, which then trains using both segmentation and classification data from Table I interchangeably. This means, that one iteration consists of two stages: first, the model samples an observation and trains on the segmentation data, then it repeats this step for the classification problem. The first stage is the same as in Mask R-CNN.

At classification training stage, following [15],

TABLE I: Comparison of models' sizes and datasets. Values are for the segmentation/classification data. One asterisk: two ResNet blocks (first and second). Two asterisks: three ResNet blocks (first, second and third). Superscript of 2 are models with the augmented RoI layer.

Model (Backbone)	#Total parameters	Training	Validation	Test	Ratio Test to Train split	Training Time(min)
ResNet18+FPN*	4.51M	650/3K	650/20.6K	100/21.1K	0.15/7.06	355
ResNet18+FPN**	6.64M					359
ResNet34+FPN*	5.17M					370
ResNet34+FPN**	12.04M					380
ResNet18+FPN*,2	6.13M					341
ResNet18+FPN**,2	8.27M					348
ResNet34+FPN*,2	6.80M					361
ResNet34+FPN**,2	13.66M					371

RoIBatchSelection method accepts β_1 predictions from BoxClass prediction layer, extracts top B predictions thereof (ranked by class confidence), and outputs batchified object predictions with dimensions $B \times 5$ (4 box coordinates + confidence score) with the following properties:

1) The batch contains a set of box coordinates (x, y, height, width),
2) For each box, the batch contains a normalized (softmax) class confidence score,
3) Predictions (boxes+scores) are ranked in the decreasing order of their class confidence scores.

Image classification layer **S** (see Figure 1) is expected to learn this distribution, which applies certain restrictions on the weights in RoI layer. Clearly, early in the training, these object predictions are inaccurate, slowing down the overall training of the model. As explained earlier, to address this deficiency, we first pretrain the model only on the segmentation data.

From the point of view of the model's architecture, there is only one RoI branch (gray area in Figure 2), i.e. all RoI weights are shared between both problems, and the only architectural difference between Mask R-CNN and SSM is still the image classifier **S**. In the segmentation stage, all SSM weights (i.e., backbone, RPN, RoI) except **S** are updated. In the classification stage, only backbone and **S** weights are updated.

Also, the main functional difference between segmentation and classification stages are still RoIDetectBatch and RoIBatchSelection batch functionalities (see Figure 2). As explained in Section III-A, in the segmentation training stage, RoIDetectBatch samples β_2 RoIs from RPN candidates (box coordinates) for loss computation. It is not used in the segmentation evaluation and classification stages. Also, as discussed earlier, RoIBatchSelection is only used in the classification to construct the batch size B of predictions for the image classification.

For the purpose of image classification it is important to point out that 'objects' in the RoiBatchSelection are not necessarily lesions. Obviously, there are none of them in Negative images, i.e. the whole Negative image is background, and the lesion confidence scores of 'objects' extracted from it will be very low, most likely well below any acceptance threshold. To maintain the size of the batch, we set acceptance threshold to -0.01, that guarantees acceptance of all 'objects'. For a more detailed discussion of RoIBatchSelection see [15]. The main drawback of this approach is that RoI layer fuses the learning of instance data (labelled at object level) and image data (labelled at image level) in a single branch. As a result, the model's accuracy on both problems erodes, because features for the classification problem distort regional detection functionality. The opposite is also true. Also, there are 5 loss functions for instance-level data, and one loss function for image-level data, which also biases training progress in favor of instance segmentation, see Section III-D and Appendix A. In Section III-C we explain how we resolve this problem.

C. Augmented RoI layer

In this approach, we disentangle instance-level detection for segmentation and classification problems through the transformation of the RoI layer.

First, we observe, that images in the classification dataset contain the same class-specific regional features (lesions, clean lungs, background), as the segmentation dataset, because they follow the same distribution, the difference is only in labels. Therefore, detection of such regions of interest will improve the accuracy of image-level prediction. To achieve this, we need an architecture that is capable of extracting such information from the classification dataset without training using instance-level labels. At the same time, RPN and RoI layers cannot be trained by the image-level labels either, because it distorts their local detection functionality and accuracy, as explained in Section III-B.

To satisfy these requirements, we augment RoI layer with a second branch for the classification data, that runs parallel to the segmentation branch, see light beige background in RoI in Figure 2, and it possesses the following properties:

1) Its parametrized architecture is identical to the detection part of the segmentation branch, which consists of two fully connected layers in BoxHead and a single one in BoxClass layer. BoxHead (image) and BoxClass (image) layers have the same number of weights and the same dimensionality as in the segmentation branch. The key idea of the new classification branch, is that

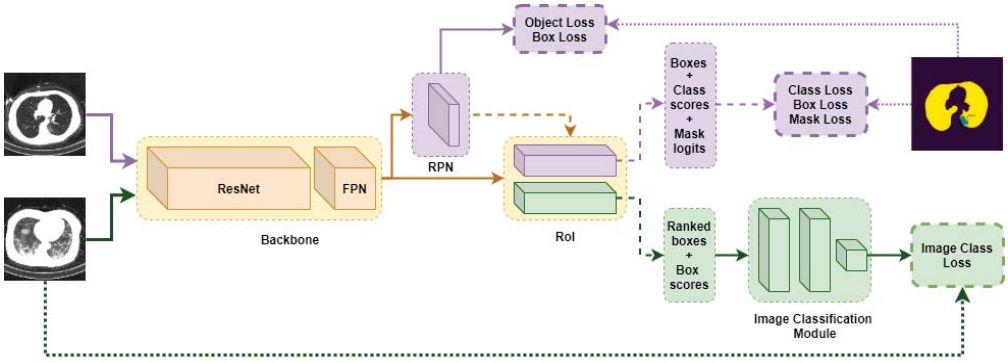

Fig. 1: Architecture of the Single Shot Model with the augmented RoI layer. Normal arrows: data and features, broken arrows: batches, dotted arrows: labels. Light beige background, blocks and arrows: both segmentation and classification. Green background, blocks and arrows: only classification. Purple background, blocks and arrows: only segmentation. CT scan source: CNCB-NCOV. Best viewed in color.

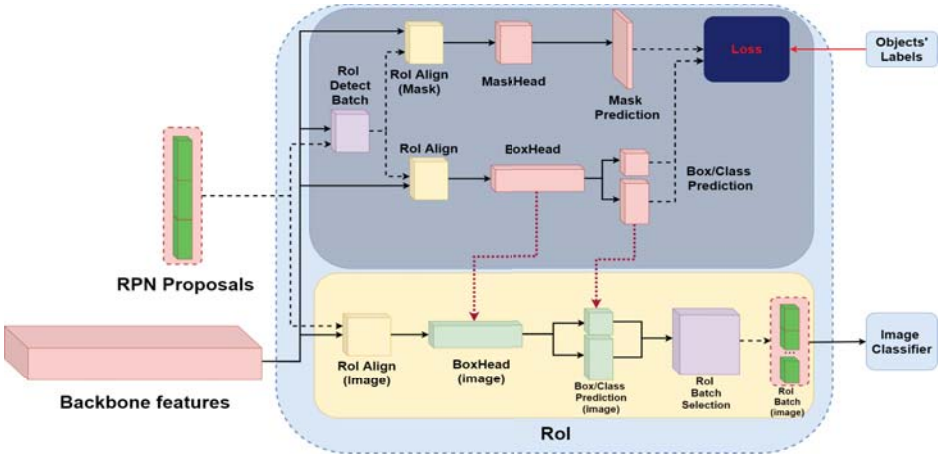

Fig. 2: RoI layer with two parallel branches for the disentanglement of RoI detection for segmentation and classification problems. Gray background: segmentation branch (training stage), light beige background: classification branch. Pink blocks: layers with trainable weights, light green blocks: layers with non-trainable weights, yellow blocks: RoIAlign, purple blocks: batches (RoIDetectBatch for segmentation and RoIBatchSelection for classification), bright green blocks: RPN batch and RoI image batch. Normal black arrows: features, normal red arrow: labels, broken black arrows: batches/samples, dotted maroon arrows: weight copy from the segmentation into the classification branch. Best viewed in color.

none of the weights in the BoxHead (image) and BoxClass (image) layers are trainable at any stage. Instead, we copy BoxHead and BoxClass weights from the segmentation branch into the classification branch, which is possible due to the identical architecture of the two, see Figure 2. As a result, classification branch inherits instance detection functionality, but applies it to the classification data and problem.

2) RoIAlign (image) has the same functionality as RoIAlign. It accepts all β_1 RPN proposals, crops and resizes corresponding areas in the FPN output to a fixed dimensionality and creates a batch of regions of interest with dimensions $\beta_1 \times C \times H \times W$. Features in the next two layers, BoxHead (image) and BoxClass (image) are extracted from each region of interest.

3) RoIBatchSelection functionality is the same as in [15]. It accepts β_1 predictions from BoxClass (image), extracts a ranked batch with dimensions $B \times 5$ and passes it to the image classification layer $\mathbf{S}$.

4) Disentanglement of these functionalities in RoI does not negatively affect the model's capacity to solve both problems. At segmentation training stage, classification branch weights are frozen. At classification training stage, its weights are also frozen, hence its regional functionality does not erode due to the the image-level loss. At the same time, it is capable of an accurate detection of the regions of interest in all CT scans, regardless of the problem at hand.

5) Careful empirical investigation confirmed that the approach does not suffer from the vanishing or exploding

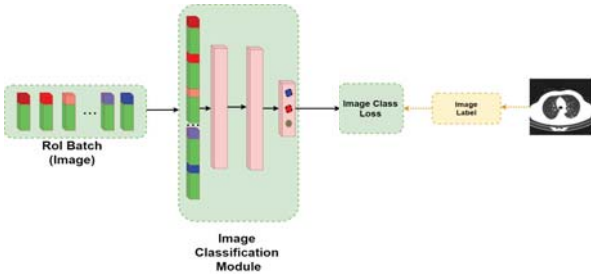

Fig. 3: RoI Batch to Feature Vector and image classification module **S**. Black arrows: features, dotted arrow: image label. Each bounding box (green blocks) has a softmax confidence score, that varies from ≈ 1 (red) to ≈ 0 (blue). CT scan source: CNCB-NCOV. Best viewed in color.

TABLE II: Precision results on the segmentation data. AP_1: AP@0.5IoU, AP_2: AP@0.75IoU, mAP: AP@[0.5:0.95]IoU, main MS COCO criterion. Bold+italicized: best, bold:second-best, italicized: third-best. One asterisk: two ResNet blocks (first and second). Two asterisks: three ResNet blocks (first, second and third). Superscript of 2 are models with the augmented RoI layer. Bold+italics: best, bold: second-best, italics: third-best.

Model	AP_1	AP_2	mAP
ResNet18+FPN*	0.4585	0.3175	0.3516
ResNet18+FPN**	0.4253	0.3222	0.3415
ResNet34+FPN*	0.5635	*0.3942*	*0.3993*
ResNet34+FPN**	0.5243	0.2984	0.3565
ResNet18+FPN^{*,2}	0.5095	0.3927	0.3923
ResNet18+FPN^{**,2}	**0.5799**	0.3828	**0.4245**
ResNet34+FPN^{*,2}	***0.6291***	***0.4648***	***0.4535***
ResNet34+FPN^{**,2}	0.5152	0.3381	0.3579
Mask R-CNN (head only)	0.5110	0.3010	0.2980
Mask R-CNN (full)	*0.5650*	**0.4130**	0.3520

gradients, heavy overfitting or large overheads, both in terms of the number of parameters and training and evaluation time.

In Section IV we compare results achieved by methods discussed in Sections III-B and III-C.

D. Loss Functions

The model computes and backpropagates L_{Total}, Equation 1, a linear combination of L_{SEG}, segmentation loss in Equation 2 and L_{CLS}, classification loss in Equation 3.

$$L_{Total} = L_{SEG} + L_{CLS} \tag{1}$$

In the segmentation training stage, RPN solves an object vs background binary problem, using overlaps of predictions extracted from the set of anchors (boxes with predefined dimensions) and gt objects to determine positives and background (box coordinates and objectness), Equation 4a. RoI solves a multiclass problem (box coordinates, classes and masks) for each prediction, Equation 4b and 4c. Therefore, Equation 2, loss of the segmentation branch, is the same as in [5], [12]. Loss function for the image classification problem, binary cross-entropy, are Equations 3 and 5.

A detailed discussion of the loss function computation is presented in Appendix A.

IV. Experimental Setup And Results

We test empirically the following three hypotheses in order to determine the best model overall:

1) Reducing ResNet18 and ResNet34 depth from the full architecture with 4 blocks to 3 and 2 with a single FPN layer does not degrade performance (see [17] on the matter of model truncation for this problem),
2) Compare the frameworks introduced in Section III: RoI layer with a single segmentation branch with pretraining vs augmented RoI layer without pretraining to determine the one that achieves better accuracy on both problems,
3) SSM achieves stronger results in both problems compared to OS baseline models.

For the experimental setup we selected two feature extractors for SSM: ResNet18 and ResNet34, because in [17] it was shown that smaller models achieve the classification accuracy close or better than that of the larger models like ResNet50 with just a fraction of the model's size. It was also shown in [17] that truncating models, i.e. deleting either the last or the last two blocks, in fact, improves the predictive quality of the model compared to the full model (see [6] for the explanation of ResNet residual architecture and Torchvision implementation, https://pytorch.org/docs/stable/torchvision/models.html).

We also considered three rules for updating model weights at classification training stage:

1) Layer **S** only,
2) Layer **S** + full backbone,
3) Layer **S** + batch normalization layers in the backbone.

Rule 1 requires the least number of weights updates, and rule 3 is in line with the highest COVID-19 sensitivity in [15]. Nevertheless, with rule 2, we achieved top results across all architectures and problems in this study, so we discarded the outputs obtained with rules 1 and 3.

In total, we trained 8 different variants of the model: 4 base models with a single RoI segmentation branch (with pretraining) and 4 augmented models with two parallel RoI branches (trained from scratch). In the first approach, segmentation model was pretrained for 50 epochs with Adam optimizer, learning rate of $1e-5$ and weight decay coefficient of $1e-3$. Important Mask R-CNN hyperparameters such as non-maximum suppression (NMS) thresholds and RoI classification confidence threshold were the same relevant as in [17]. All SSMs with either architecture were trained in a similar way, with an Adam optimizer, learning rate of $1e-5$ and weight decay coefficient of $1e-3$.

We do not compare our results to publicly available COVID-19 solutions due to a long list of methodological differences (see Section V for their discussion). Instead, we trained and evaluated several state-of-the art models on our

TABLE III: Class sensitivity, overall accuracy and F1-score results on COVIDx-CT test data for 3 classes (21192 images). One asterisk: two ResNet blocks (first and second). Two asterisks: three ResNet blocks (first, second and third). Superscript of 2 are models with the augmented RoI layer. Bold+italics: best, bold: second-best, italics: third-best.

Model	COVID-19	CP	Normal	Overall	F1 score
ResNet18+FPN*	90.20%	89.52%	89.34%	89.58%	0.8969
ResNet18+FPN**	83.00%	89.55%	**98.62%**	92.25%	0.9220
ResNet34+FPN*	88.70%	83.35%	94.54%	89.43%	0.8941
ResNet34+FPN**	87.13%	**96.75%**	88.04%	91.23%	0.9138
ResNet18+FPN*,2	***93.16%***	95.68%	96.18%	*95.38%*	*0.9542*
ResNet18+FPN**,2	**93.00%**	*96.53%*	***98.64%***	***96.75%***	***0.9676***
ResNet34+FPN*,2	89.62%	89.99%	96.76%	92.93%	0.9293
ResNet34+FPN**,2	91.44%	95.33%	92.48%	93.26%	0.9333
ResNet18	*92.59%*	96.25%	92.03%	93.58%	0.9361
ResNeXt50	91.94%	88.45%	84.21%	87.25%	0.8731
ResNeXt101	91.58%	92.13%	94.02%	92.87%	0.9286
DenseNet169	89.37%	***96.78%***	*98.12%*	**95.81%**	**0.9586**

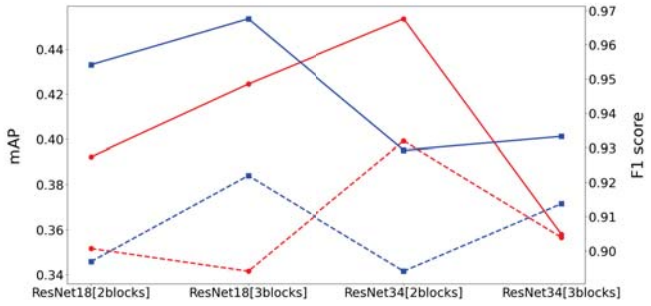

Fig. 4: Segmentation (red) and Classification (blue) accuracy for base model (broken line) and augmented model (normal line) across all architectures.

data: Mask R-CNN, ResNet18, ResNeXt50, ResNeXt101, DenseNet169 using the same hyperparameters as in SSM.

For the segmentation accuracy, we used MS COCO criteria [9]: AP@0.5 intersect over union (IoU) threshold (AP_1 in Table II), AP@0.75 IoU threshold (AP_2 in Table II) and AP averaged across 10 IoU thresholds, AP@[0.5:0.95]IoU with a step of 0.05 (mAP in Table II). For the classification accuracy in Table III, we used well-known metrics: class sensitivity, overall accuracy and class-adjusted F_1 score. Details of both test splits is presented in Table I.

Although at present there is no unified metric that balances detection and classification results, from the results presented in Table II and III we can infer several things. First, for the classification problem, adding the third block gave a stable improvement in F1 score and overall accuracy for each architecture. At the same time, performance on the segmentation data deteriorated. Second, switching from ResNet18 to ResNet34 with the same number of blocks overall improves mAP, but causes deterioration of the model's performance on the classification data. This means that neither depth, nor the feature extractor architecture or the number of parameters are good predictors of the model's accuracy.

Most importantly though, the results in Tables II and III, visualized in Figure 4, demonstrate the strength of the models with the augmented RoI layer across all architectures and problems. For the segmentation problem, augmented ResNet34 with 2 blocks reports the highest precision across all critera, and augmented ResNet18 with three blocks the second-highest for AP@50% IoU and mAP. For the classification problem, augmented ResNet18 with three blocks achieves top results for the Normal class, overall accuracy and F1 score, second-best COVID-19 sensitivity and third-best CP sensitivity. Augmented ResNet18 with 2 blocks achieves top COVID-19 sensitivity.

For mAP, augmented ResNet34 with two blocks beats the best base model by 0.054, for AP2 by 0.07, and for AP1 by 0.065. For COVID-19 sensitivity, augmented ResNet18 with two blocks beats the best base model, ResNet18 with 2 blocks, by 2.95%, for Normal class augmented ResNet18 with 3 blocks beats base ResNet18 with three blocks by 0.02%. Measured by the overall accuracy, it beats base ResNet18 with 3 blocks by 4.50% and measured by F1 score it also beats the same model by 4.56%. The only criteria, for which all models with the augmented RoI underperform the best base, ResNet34 with 2 blocks model is CP sensitivity, by 0.22%. Also, for this criterion, top benchmark model, DenseNet169, achieved top result beating base ResNet34 with 2 blocks further by 0.03%.

V. Methodological and Experimental Milestones and Limitations

At present, COVID-19 benchmark dataset like MS COCO or Pascal VOC, on which different models could be compared, does not exist. This is the consequence of a rapid spread of the infection, and the difference in the methods of collection and processing of CT data across OS studies. On top of that, to the best of our knowledge, none of the OS models generalizes to other datasets and real-life applications in radiology departments with good enough accuracy.

This problem is thoroughly investigated in [13], which presents a systematic review of 62 models for CXR and

CT data reported in publicly available studies. According to this study, due to a long list of methodological flaws and overfitting, no model can be used for real-life implementation. Another observation in this context, is that publications rarely address these limitations. Following the recommendations in [13], in our study, we attempted to overcome some of them:

1) Full source code, trained models and protocols are available on a Github repository, hence, our results can be easily verified,
2) We provide the full list of hyperparameters we used to train and evaluate our models, both in the paper and on Github,
3) In our study we worked with a large OS dataset, that includes both classification and segmentation data,
4) The ratio of test to train splits for classification is one of the highest available in the literature, all images were resized to 512×512 pixels, we did not use the weights from models pretrained on large benchmark datasets, such as ImageNet or MS COCO,
5) All methodology is clearly explained in sufficient details,
6) The models were compared to a suite of the state-of-the-art OS models for both problems.

Despite our best efforts, the presented SSM still suffers from the data bias, and in its current form may not be implementable in a real-life setting, without additional finetuning. This is the main limitation of our approach so far, and it is our best intention to address it in the follow-up studies. Nevertheless, methodology and results achieved in this study imply that the model has a strong potential for generalization, For example, the ratio of test to training splits in classification data, and the ability of the model to solve two problems in a single shot.

Also, in our experiments, we could not establish a single backbone architecture that could achieve the highest precision and accuracy for both problems. Augmented ResNet34 with 2 blocks achieved top results on the segmentation task, but could not outperform a number of other architectures on classification problem. At the same time, augmented ResNet18 with 3 blocks achieved the highest F1 score, overall accuracy and Control sensitivity and second-highest COVID-19 sensitivity and third-highest CP sensitivity, and two strong results on the segmentation data. Augmented ResNet18 with 2 blocks achieved top COVID-19 sensitivity. Also, we established that key backbone hyperparameters: number of weights, depth, architecture have no effect on the accuracy for either problem.

VI. Conclusions

In this paper we presented a fast and accurate lightweight single shot model model that fuses lesion instance segmentation and chest CT scans classification (COVID-19, Common Pneumonia, Control). Conceptual novelties include the disentanglement of lesion instance detection functionality for segmentation and classification tasks through the augmentation of the RoI layer with a separate classification branch. To the best of our knowledge, this is the first solution in the COVID-19 deep learning community that fuses lesion instance segmentation and COVID-19 classification a single shot. Our experiments show with confidence that this innovation strongly improves the model's performance across both tasks and all architectures.

So far we achieved strong results by only exploring the localization (bounding box) and the class strength (confidence score) of the objects (either lesions or background). One of the key differences between COVID-19 and CP is the configuration of lesions and features: diffuse distribution, attenuation, crazy-paving patterns, etc, that rectangular bounding boxes cannot capture. In our future research we intend to exploit the differences in these configurations for COVID-19 prediction by using masks (mask features) that capture the objects' shapes more accurately than bounding boxes. Full source code, model interfaces and pretrained weights are available on
https://github.com/AlexTS1980/COVID-Single-Shot-Model.

References

[1] T. Ai, Z. Yang, H. Hou, C. Zhan, C. Chen, W. Lv, Q. Tao, Z. Sun, and L. Xia, "Correlation of chest ct and rt-pcr testing in coronavirus disease 2019 (covid-19) in china: a report of 1014 cases," *Radiology*, p. 200642, 2020.

[2] C. Butt, J. Gill, D. Chun, and B. A. Babu, "Deep learning system to screen coronavirus disease 2019 pneumonia," *Applied Intelligence*, pp. 1–7, 2020.

[3] D.-P. Fan, T. Zhou, G.-P. Ji, Y. Zhou, G. Chen, H. Fu, J. Shen, and L. Shao, "Inf-net: Automatic covid-19 lung infection segmentation from ct images," *IEEE Transactions on Medical Imaging*, 2020.

[4] H. Gunraj, L. Wang, and A. Wong, "Covidnet-ct: A tailored deep convolutional neural network design for detection of covid-19 cases from chest ct images," *arXiv preprint arXiv:2009.05383*, 2020.

[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[6] 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.

[7] L. Li, L. Qin, Z. Xu, Y. Yin, X. Wang, B. Kong, J. Bai, Y. Lu, Z. Fang, Q. Song *et al.*, "Artificial intelligence distinguishes covid-19 from community acquired pneumonia on chest ct," *Radiology*, 2020.

[8] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[9] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[10] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[11] M. Polsinelli, L. Cinque, and G. Placidi, "A light cnn for detecting covid-19 from ct scans of the chest," *arXiv preprint arXiv:2004.12837*, 2020.

[12] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[13] M. Roberts, D. Driggs, M. Thorpe, J. Gilbey, M. Yeung, S. Ursprung, A. I. Aviles-Rivero, C. Etmann, C. McCague, L. Beer *et al.*, "Common pitfalls and recommendations for using machine learning to detect and prognosticate for covid-19 using chest radiographs and ct scans," *Nature Machine Intelligence*, vol. 3, no. 3, pp. 199–217, 2021.

[14] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[15] A. Ter-Sarkisov, "COVID-CT-Mask-Net: Prediction of COVID-19 from CT Scans Using Regional Features," *medRxiv*, 2020. [Online]. Available: https://github.com/AlexTS1980/COVID-CT-Mask-Net

[16] ——, "Detection and segmentation of lesion areas in chest CT scans for the prediction of COVID-19," *medRxiv*, 2020.

[17] ——, "Lightweight model for the prediction of COVID-19 through the Detection and Segmentation of lesions in chest ct scans," *medRxiv*, 2020.

[18] Y.-H. Wu, S.-H. Gao, J. Mei, J. Xu, D.-P. Fan, C.-W. Zhao, and M.-M. Cheng, "Jcs: An explainable covid-19 diagnosis system by joint classification and segmentation," *arXiv preprint arXiv:2004.07054*, 2020.

[19] K. Zhang, X. Liu, J. Shen, Z. Li, Y. Sang, X. Wu, Y. Zha, W. Liang, C. Wang, K. Wang *et al.*, "Clinically applicable ai system for accurate diagnosis, quantitative measurements, and prognosis of covid-19 pneumonia using computed tomography," *Cell*, 2020.

[20] L. Zhou, Z. Li, J. Zhou, H. Li, Y. Chen, Y. Huang, D. Xie, L. Zhao, M. Fan, S. Hashmi, F. Abdelkareem, R. Eiada, X. Xiao, L. Li, Z. Qiu, and X. Gao, "A rapid, accurate and machine-agnostic segmentation and quantification method for ct-based covid-19 diagnosis," *IEEE Transactions on Medical Imaging*, vol. 39, no. 8, pp. 2638–2652, 2020.

APPENDIX A
EQUATIONS

$$L_{SEG} = L_{RPN} + L_{RoI} + L_{Mask} \tag{2}$$

$$L_{CLS} = -\log\sigma(x^{C^{**}}) - \sum_{j\in C^{*}} \log(1-\sigma(x_j^{C^{*}})) \tag{3}$$

$$L_{RPN} = \sum_{j=1}^{n_{RPN}^{pos}} |\mathbf{x_j^{box}} - \mathbf{t^{box}}| + \sum_{j=1}^{n_{RPN}} q_j^{C}(x) \tag{4a}$$

$$L_{RoI} = \sum_{j=1}^{n_{RoI}^{pos}} L_1\mathrm{smooth}(|\mathbf{x_j^{box,C^{**}}} - \mathbf{t^{box}}|) - \sum_{j=1}^{n_{RoI}} \log\mathrm{softmax}(x_j^{C^{**}}) \tag{4b}$$

$$L_{Mask} = -\sum_{j=1}^{n_{RoI}^{pos}} \mathbf{m_j}\log\sigma(\mathbf{x_j^{C^{**}}}) \tag{4c}$$

$$q_j^{C}(x) = -\log(1-\sigma(x_j^{C^{*}})) - \log\sigma(x_j^{C^{**}}) \tag{5}$$

Here $n_{RPN/RoI}^{pos}$ is the number of positive predictions in the RPN/RoI samples, $n_{RPN/RoI}$ is the total number of predictions in the RPN/RoI samples, C refers to all classes, C^{*} are the incorrect classes, C^{**} is the correct class, σ is the sigmoid function. Since RPN and RoI train by constructing batches from each image, all sums in Equations 4a and 4b are over candidates sampled into the batch. Positive (non-background) predictions are determined by their overlaps with the gt boxes.

1) RPN box coordinates: $\mathbf{x_j^{box}}$ is the box coordinates predictions, the absolute difference is computed only for the box coordinates of the correct positive class, $\mathbf{t^{box}}$ are box coordinates for the corresponding gt object,
2) RPN classes: $q_j^{C}(x)$, Equation 5, is binary cross-entropy loss for both background and object, x is the class logit score,
3) RoI box coordinates: $L_1\mathrm{smooth}$ is a variation of absolute distance function, the difference is computed only for the box coordinates of the correct positive class, $\mathbf{x_j^{box,C^{**}}}$ and the corresponding gt box,
4) RoI classes: cross-entropy loss, with $x_j^{C^{**}}$ is the logit score for the correct class in each RoI prediction, including the background,
5) RoI masks: pixelwise binary cross-entropy between gt binary mask for the correct class, $\mathbf{m}_j$ and the logits for the correct class, $\mathbf{x_j^{C^{**}}}$