Cov3d: Detection of the presence and severity of COVID-19 from CT scans using 3D ResNets

Robert Turnbull[1]

[1]Melbourne Data Analytics Platform
University of Melbourne, Victoria 3052, Australia
robert.turnbull@unimelb.edu.au
https://mdap.unimelb.edu.au

July 5, 2022

Abstract

Deep learning has been used to assist in the analysis of medical imaging. One such use is the classification of Computed Tomography (CT) scans when detecting for COVID-19 in subjects. This paper presents Cov3d, a three dimensional convolutional neural network for detecting the presence and severity of COVID19 from chest CT scans. Trained on the COV19-CT-DB dataset with human expert annotations, it achieves a macro f1 score of 0.9476 on the validation set for the task of detecting the presence of COVID19. For the task of classifying the severity of COVID19, it achieves a macro f1 score of 0.7552. Both results improve on the baseline results of the 'AI-enabled Medical Image Analysis Workshop and Covid-19 Diagnosis Competition' (MIA-COV19D) in 2022.

Keywords: Deep Learning, Computer Vision, Medical Imaging, Computed Tomography (CT), 3D ResNet, COVID-19.

1 Introduction

To best care for patients, medical professionals need fast and accurate methods for detecting the presence and severity of COVID-19 in patients. Nucleic acid amplification tests (NAAT), such as real-time reverse transcription polymerase chain reaction (rRT-PCR), are recommended by the World Health Organisation (WHO) because they are highly specific and sensitive [21]. These kinds of tests have the disadvantage of taking a long time and not occurring at the point of need [16]. Medical imaging can be used to complement these diagnostic strategies [9]. Computed Tomography (CT) scans use x-rays to reconstruct cross-sectional images to produce a three dimensional representation of the internals of the body [17]. Thoracic radiologists have used chest CT scans to

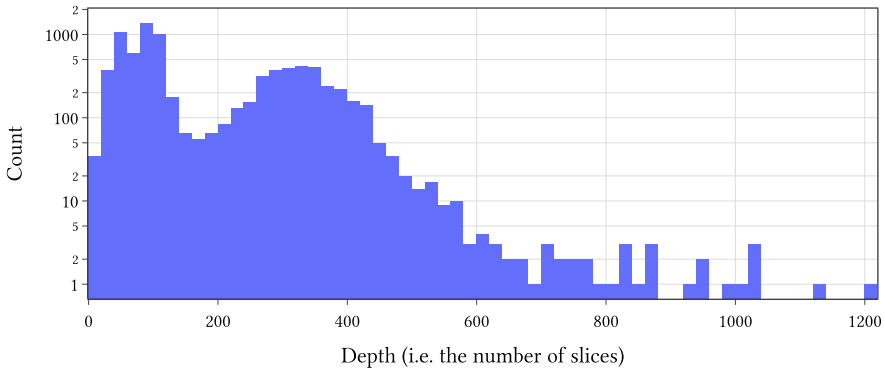

Figure 1: A histogram of the number of slices NB. The vertical axis uses a logarithmic scale.

correctly diagnose patients with COVID-19 including cases where rRT-PCR gives a negative result [22]. But this technique requires a human interpreter with sufficient knowledge and experience to provide reliable results. Advances in deep learning (DL) for computer vision offers the possibility of using artificial intelligence for the task of CT scan image analysis [12]. Early in the pandemic, attempts to use deep learning showed significant promise for accurate detection of COVID-19 [2, 11, 13]. To further this area of research, the 'AI-enabled Medical Image Analysis Workshop and Covid-19 Diagnosis Competition' (MIA-COV19D) was created as part of the International Conference on Computer Vision (ICCV) in 2021 [10]. This competition sought submissions to predict the presence of COVID-19 in a dataset of CT scan cross-sectional images. The winning submission produced a macro F1 score of 90.43 on the test partition of the dataset [5]. In 2022, the AI-enabled Medical Image Analysis Workshop issued a second competition with an enlarged dataset and new task which is to also predict the severity of COVID-19 in pa [9]. This article presents an approach to the tasks of this competition by using a three dimensional convolutional neural network called Cov3d.[1]

2 The COV19-CT-DB Database

The database (COV19-CT-DB) comprises more than 7,700 chest CT scans from more than 3,700 subjects and is annotated to record whether or not the subject had COVID-19 [9]. Each CT scan contains 2D slices perpendicular to the long axis of the subject. The number of slices (hereafter referred to as the 'depth') of the scan ranges from 1 to 1201. It has a bimodal distribution (see fig. 1)

[1]The code for this project has been released under the Apache-2.0 license and is publicly available at https://github.com/rbturnbull/cov3d.

Table 1: The number of CT scans in the partitions of the database

COVID-19	Training	Validation	Test
Positive	882	215	–
Negative	1,110	269	–
Total	2,292	484	5,281

Table 2: The number of CT scans with severity annotations in the partitions of the database.

Index	Severity	Training	Validation	Test
1	Mild	85	22	–
2	Moderate	62	10	–
3	Severe	85	22	–
4	Critical	26	5	–
	Total	258	106	265

with more then 58% of scans have fewer than 160 slices and almost 99% having under 500 slices. The slices are provided as sequentially numbered JPEG files and typically have a resolution of 512×512 pixels. It has been divided into training, validation and test partitions (table 1).

For a minority of the CT scans where COVID-19 was present, four experts have annotated the severity of disease in the patient with four categories: mild, moderate, severe and critical. These categories correspond to greater degrees of pulmonary parenchymal involvement. These annotations are indicated in CSV files that accompany the database. The number of scans with these annotations for the three partitions in the database are given in table 2.

3 Methods

To analyze this database, this paper discusses Cov3d, a classifier of CT scan images using three dimensional convolutional neural networks. Cov3d uses the deep learning framework PyTorch [14]. Cov3d also uses the fastai library which adds higher level components [6]. It also is built using the FastApp framework for packaging deep learning models built with fastai and wrapping them in a command-line interface.[2]

3.1 Preprocessing

As seen in fig. 1, the number of slices in each CT scan is significantly varied. To regularize these so that multiple scans could be processed in batches and

[2]FastApp is available as an alpha release at: https://github.com/rbturnbull/fastapp/.

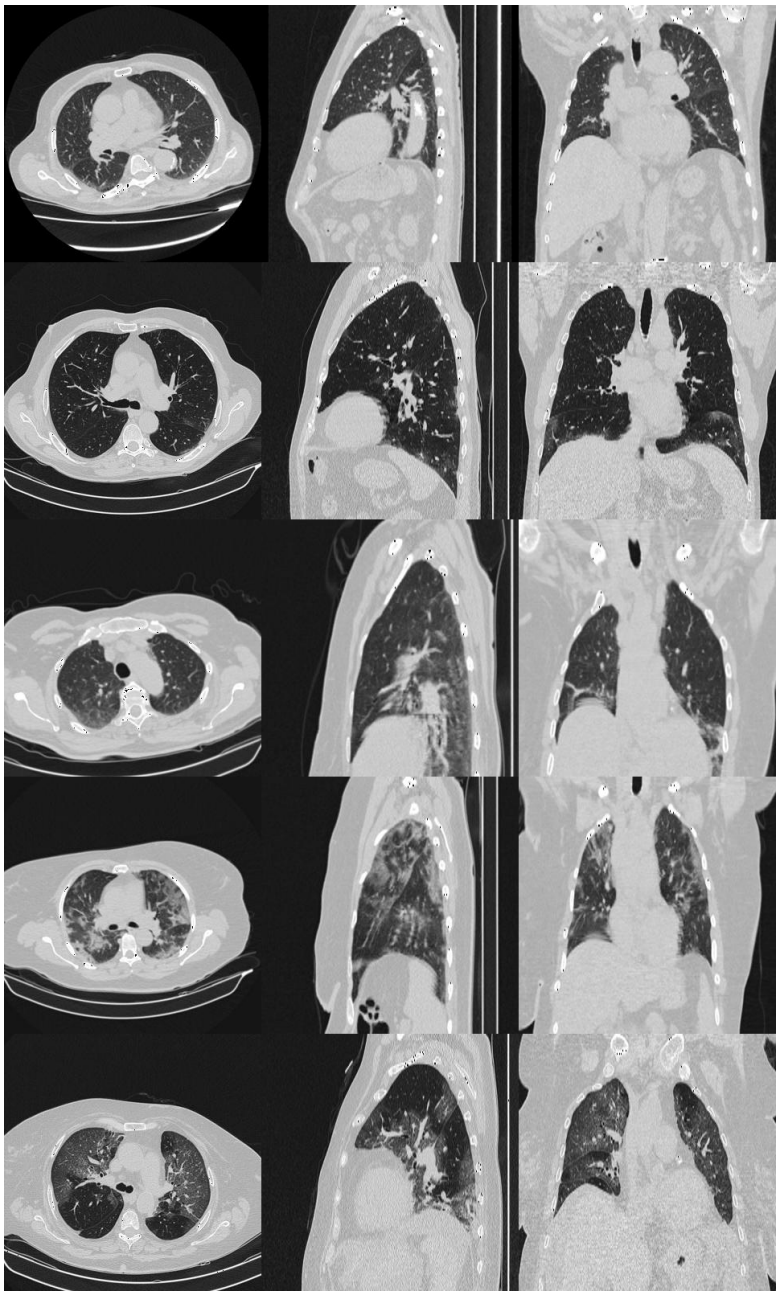

Figure 2: Cross-sections from the five CT scans in the axial (left), sagittal (middle), and coronal (right) planes. The rows correspond to subjects without COVID19 (top) and then having COVID19 with severity categorizes as (in order): mild, moderate, severe and critical (bottom).

to reduce the size of the inputs to fit within the memory of the hardware, each scan went through a preprocessing step. First the 2D slices were reduced from a resolution of 512^2 pixels to 128^2, 256^2 or 320^2 pixels through bicubic interpolation. Then the slices were interpolated using 1D cubic interpolation along the axis perpendicular to the cross sections so that the depth was exactly half of the width/height of the processed scan. This gives three resolution sizes: small (64×128×128), medium (128×256×256) and large (160×320×320). These processed 3D images are saved as PyTorch tensor files and are loaded from disk during the training cycle. Only a single channel is stored for each image.

3.2 Neural Network Model

ResNet models have been tremendously popular for computer vision tasks on 2D images [3]. The residual 'shortcut' connections between the layers address the problem of vanishing gradients which allows for models with a greater number of layers. Cov3d uses a neural network model analogous to a two dimensional ResNet-18 model but with 3D convolutional and pooling operations instead of their 2D equivalents. In particular, Cov3d uses the ResNet 3D 18 model included in the Torchvision library [20]. This model has been pre-trained on the Kinetics-400 dataset which classifies short video clips [7]. The time dimension in the pre-trained model was used for the depth dimension of the CT scans. Though the CT scans are quite different to video clip data in Kinetics-400, we can anticipate that the pre-trained network will have learned to identify shapes and patterns, particularly at the early layers of the network, which ideally will improve training through transfer learning (TL).

A number of modifications to the pre-trained model were made before use. Since the pre-trained model used 3 channel inputs, the weights for the initial convolutional layer were summed across the channels so that single channel input could be used. Dropout [4] was added after each of the four main layers of ResNet model. The final linear layer to predict the 400 categories of the Kinetics dataset was replaced with two linear layers with a ReLU activation function between then and dropout. The size of the penultimate layer was varied as a hyperparameter and the size of the final layer was dependent on the loss function discussed below.

3.3 Loss

For detection of the presence of COVID-19, the penultimate layer connects to a single logit x_i which models the log-odds that the subject has the disease, where i refers to the index of the input. This can be converted to a probability p_i using the sigmoid function:

$$p_i = \frac{1}{1 + e^{-x_i}} \tag{1}$$

The loss for task 1 (i.e. detection of the presence of COVID-19) is denoted

ℓ_{covid} and is given by the weighted binary cross entropy:

$$\ell_{covid}(p_i, y_i) = -w_i \left(y_i \log p_i + (1 - y_i) \log(1 - p_i)\right) \tag{2}$$

where w_i is the weight of the input item and y_i is the target probability. The weights are set to compensate for the different proportion of COVID-19 and non-COVID-19 scans in the training partition (see table 1).

Szegedy et al. introduced a regularization mechanism called label-smoothing which modifies the target probabilities to mitigate against the network becoming overconfident in the result [19]. This technique is employed here and thus the target probability for positive COVID19 scans is set to $1 - \epsilon_p$ and the target probability for negative COVID19 scans is ϵ_p. The hyperparameter ϵ_p was set to either 0.0 or 0.1.

Task 2 requires inferring the severity of COVID-19 in patients base on the four categories. The dataset for this task is quite small (table 2) which makes training a model challenging. We can expand this dataset by regarding the COVID19 negative scans as belonging to a fifth category. Furthermore, the COVID19 positive scans which have not been added in the dataset can be regarded as belonging to a superset of those four severity categories. This can be modelled by connecting the penultimate layer to a vector of five dimensions ($z_{i,c}$) corresponding to the four severity categories ($c = 1, 2, 3, 4$) and an additional category for being COVID19 negative ($c = 0$). The probability distribution over the categories ($s_{i,c}$) is given by the softmax function:

$$s_{i,c} = \frac{e^{z_{i,c}}}{\sum_{j=0}^{4} e^{z_{i,j}}} \tag{3}$$

The probability that the scan is merely COVID19 positive is given by summing over the four severity categories ($\sum_{c=1}^{4} s_{i,c}$). If the input is in one of the four annotated severity categories, or is classed as non-COVID19, then the loss is thus given with the cross entropy:

$$\ell_{severity}(s_{i,c}, y_i) = -w_i \sum_{c=0}^{4} y_{i,c} \log(s_{i,c}) \tag{4}$$

where w_i is the weight of the input item and y_i is the target probability for class c. If the input is annotated as COVID19 positive but without a severity category then the loss is given by:

$$\ell_{severity}(s_{i,c}, y_i) = -w_i \log(\sum_{c=1}^{4} s_{i,c}) \tag{5}$$

In this way, the second task to predict the severity can be trained using the entire database. Accuracy metrics for this task are restricted to the instances in the validation set which were annotated with the four severity categories.

As above, label smoothing was used as a regularization technique. However, unlike in Szegedy where the target probabilities were combined with a uniform

distribution [19], here the target probability for each class is reduced to $1 - \epsilon_s$ and the remaining probability of ϵ_s is divided between neighbouring categories whilst non-neighbouring categories remain with a target probability of zero. Non-COVID19 scans were considered to be neighbouring to 'mild' severity scans.

The two loss functions discussed above could be used independently to train in separate models or the can be combined as a linear combination to train a single model to perform both tasks simultaneously:

$$\ell_{combined} = (1 - \lambda)\ell_{covid} + \lambda\ell_{severity} \tag{6}$$

where $0.0 \leq \lambda \leq 1.0$.

3.4 Training Procedure

The models were trained for 30, 40 or 50 epochs through the training dataset. The batch size was limited to two because of memory constraints. The Adam optimization method was used for updating the training parameters [8]. The maximum learning rate was set to 10^{-4} and this and the momentum for the optimizer were scheduled according to the '1cycle' policy outlined by Smith [18]. Scikit-learn was used to calculate the macro f1 scores on the validation set every epoch [15]. The weights which yielded the highest macro f1 score on the validation dataset for the two tasks are saved for later inference.

3.5 Regularization and Data Augmentation

To mitigate against overfitting on the training dataset, there is the option to randomly reflect the input scans through the sagittal plane each training epoch. At inference, the reflection transformation is then applied to the input and the final probability predictions are taken from the mean. Weight decay of 10^{-5} was applied.

4 Results

The models were trained using NVIDIA Tesla V100-SXM2-32GB GPUs on the University of Melbourne's high-performance computing system Spartan. Results were logged using the 'Weights and Biases' platform for experiment tracking [1].

The results of experiments with different hyperparameter settings are shown in table 3. No one set of hyperparameters achieved the highest result in both tasks so two separate models are stored for inference on the two tasks. The highest macro f1 score for task 1 was 0.9476 which is significantly above the baseline of 0.77. This model used the highest resolution (160×320×320) and had a λ value of 0.1 which weighs the loss heavily to ℓ_{covid}. The highest macro f1 score for task 2 was 0.7552 which is above the baseline of 0.63.

The submission for the test set includes the four highest performing models for each task (marked [1]–[4]) and also a basic ensemble of the four which averages the probability predictions between each model.

Table 3: Results of experiments. The best result for each task is highlighted in bold. In descending order of task 1 macro f1.

ID	Depth	Width/ Height	Reflection	λ	Epochs	ϵ_p	ϵ_s	Task 1 macro f1	Task 2 macro f1
1	160	320	Yes	0.1	40	0.1	0.0	**0.9476** [1]	0.4764
2	160	320	Yes	0.1	50	0.1	0.0	0.9412 [2]	0.4417
3	160	320	No	0.1	30	0.1	0.1	0.9394 [3]	0.4508
4	160	320	Yes	0.0	40	0.0	0.1	0.9372 [4]	–
5	160	320	Yes	0.5	30	0.1	0.1	0.9372	0.5952
6	160	320	Yes	0.9	50	0.1	0.0	0.9371	0.6369
7	128	256	Yes	0.1	40	0.1	0.0	0.9351	0.4424
8	128	256	Yes	0.1	50	0.1	0.0	0.9328	0.4812
9	128	256	Yes	0.1	40	0.0	0.0	0.9311	0.388
10	160	320	Yes	0.1	30	0.1	0.1	0.9287	0.4153
11	160	320	Yes	0.9	40	0.1	0.0	0.9277	0.6623
12	64	128	Yes	0.1	40	0.1	0.1	0.9268	0.3988
13	128	256	No	0.5	30	0.1	0.1	0.9264	0.5909
14	128	256	Yes	0.9	40	0.1	0.0	0.9256	**0.7546** [1]
15	128	256	No	0.1	30	0.1	0.1	0.9249	0.4416
16	160	320	Yes	0.9	40	0.1	0.1	0.9237	0.6766 [3]
17	128	256	Yes	0.0	40	0.0	0.1	0.9226	–
18	64	128	Yes	0.2	30	0.1	0.0	0.9222	0.4745
19	64	128	Yes	0.1	50	0.1	0.0	0.9204	0.4725
20	64	128	Yes	0.0	40	0.1	0.0	0.9183	–
21	64	128	Yes	0.1	40	0.1	0.0	0.9183	0.4376
22	64	128	Yes	0.0	40	0.0	0.1	0.9183	–
23	128	256	Yes	0.1	30	0.1	0.1	0.9167	0.4354
24	64	128	No	0.1	30	0.1	0.0	0.9093	0.3873
25	64	128	Yes	0.9	40	0.1	0.0	0.9062	0.6754 [4]
26	64	128	No	0.5	30	0.1	0.0	0.8973	0.6143
27	128	256	Yes	1.0	40	0.0	0.0	–	0.6712
28	64	128	Yes	1.0	40	0.0	0.0	–	0.6811 [2]

[RESULTS TO FOLLOW ON THE TEST SET ONCE THE INFORMATION IS RELEASED]

5 Conclusion

Cov3d is a three dimensional model for detecting the presence and severity of COVID19 in subjects from chest CT scans. It is based on a 3D ResNet pretrained on video data. The model was trained using a customized loss function to simultaneously predict the dual tasks in the 'AI-enabled Medical Image Analysis Workshop and Covid-19 Diagnosis Competition'. The results for both tasks improve upon the baseline results for the challenge. Cov3d shows that deep learning can be used to interpret medical imaging such as CT scans and holds promise for complementing an array of other diagnostic methods to provide better care for patients.

6 Acknowledgements

This research was supported by The University of Melbourne's Research Computing Services and the Petascale Campus. Sean Crosby and Naren Chinnam were instrumental in arranging the computational resources necessary. David Turnbull gave feedback on an earlier form of this paper which clarified the expression.

References

[1] Biewald, L.: Experiment tracking with weights and biases (2020), `https://www.wandb.com/`, software available from wandb.com

[2] Harmon, S.A., Sanford, T.H., Xu, S., Turkbey, E.B., Roth, H., Xu, Z., Yang, D., Myronenko, A., Anderson, V., Amalou, A., Blain, M., Kassin, M., Long, D., Varble, N., Walker, S.M., Bagci, U., Ierardi, A.M., Stellato, E., Plensich, G.G., Franceschelli, G., Girlando, C., Irmici, G., Labella, D., Hammoud, D., Malayeri, A., Jones, E., Summers, R.M., Choyke, P.L., Xu, D., Flores, M., Tamura, K., Obinata, H., Mori, H., Patella, F., Cariati, M., Carrafiello, G., An, P., Wood, B.J., Turkbey, B.: Artificial intelligence for the detection of covid-19 pneumonia on chest ct using multinational datasets. Nature Communications **11**(1), 4080 (2020). https://doi.org/10.1038/s41467-020-17971-2, `https://doi.org/10.1038/s41467-020-17971-2`

[3] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90

[4] Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors (2012). https://doi.org/10.48550/ARXIV.1207.0580, `https://arxiv.org/abs/1207.0580`

[5] Hou, J., Xu, J., Feng, R., Zhang, Y., Shan, F., Shi, W.: Cmc-cov19d: Contrastive mixup classification for covid-19 diagnosis. In: 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW). pp. 454–461 (2021). https://doi.org/10.1109/ICCVW54120.2021.00055

[6] Howard, J., Gugger, S.: Fastai: A Layered API for Deep Learning. Information **11**(2) (2020). https://doi.org/10.3390/info11020108, `https://www.mdpi.com/2078-2489/11/2/108`

[7] Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., Suleyman, M., Zisserman, A.: The kinetics human action video dataset (2017). https://doi.org/10.48550/ARXIV.1705.06950, `https://arxiv.org/abs/1705.06950`

[8] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014). https://doi.org/10.48550/ARXIV.1412.6980, `https://arxiv.org/abs/1412.6980`

[9] Kollias, D., Arsenos, A., Kollias, S.: Ai-mia: Covid-19 detection & severity analysis through medical imaging. arXiv preprint arXiv:2206.04732 (2022)

[10] Kollias, D., Arsenos, A., Soukissian, L., Kollias, S.: Mia-cov19d: Covid-19 detection through 3-d chest ct image analysis. arXiv preprint arXiv:2106.07524 (2021)

[11] Kollias, D., Bouas, N., Vlaxos, Y., Brillakis, V., Seferis, M., Kollia, I., Sukissian, L., Wingate, J., Kollias, S.: Deep transparent prediction through latent representation analysis. arXiv preprint arXiv:2009.07044 (2020)

[12] Kollias, D., Tagaris, A., Stafylopatis, A., Kollias, S., Tagaris, G.: Deep neural architectures for prediction in healthcare. Complex & Intelligent Systems **4**(2), 119–131 (2018)

[13] Kollias, D., Vlaxos, Y., Seferis, M., Kollia, I., Sukissian, L., Wingate, J., Kollias, S.D.: Transparent adaptation in deep medical image diagnosis. In: TAILOR. p. 251–267 (2020)

[14] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`

[15] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)

[16] Peeling, R.W., Heymann, D.L., Teo, Y.Y., Garcia, P.J.: Diagnostics for covid-19: moving from pandemic response to control. Lancet (London, England) **399**, 757–768 (2022). https://doi.org/10.1016/S0140-6736(21)02346-1

[17] Seeram, E.: Computed tomography: A technical review. Radiologic Technology **89**, 279CT–302CT (2018)

[18] Smith, L.N.: A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay (2018). https://doi.org/10.48550/ARXIV.1803.09820, `https://arxiv.org/abs/1803.09820`

[19] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition

(CVPR). pp. 2818–2826. IEEE Computer Society, Los Alamitos, CA, USA (jun 2016). https://doi.org/10.1109/CVPR.2016.308, https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.308

[20] Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6450–6459 (2018). https://doi.org/10.1109/CVPR.2018.00675

[21] World Health Organization: Recommendations for national SARS-CoV-2 testing strategies and diagnostic capacities. https://www.who.int/publications/i/item/WHO-2019-nCoV-lab-testing-2021.1-eng, accessed: 2022-06-27

[22] Xie, X., Zhong, Z., Zhao, W., Zheng, C., Wang, F., Liu, J.: Chest CT for Typical Coronavirus Disease 2019 (COVID-19) Pneumonia: Relationship to Negative RT-PCR Testing. Radiology **296**(2), E41–E45 (2020). https://doi.org/10.1148/radiol.2020200343, https://doi.org/10.1148/radiol.2020200343, pMID: 32049601